

Copyright

by

Jesse David Thomason

2018

**The Dissertation Committee for Jesse David Thomason
certifies that this is the approved version of the following dissertation:**

**Continually Improving Grounded Natural Language
Understanding through Human-Robot Dialog**

Committee:

Raymond J. Mooney, Supervisor

Peter Stone

Scott Niekum

Stefanie Tellex

**Continually Improving Grounded Natural Language
Understanding through Human-Robot Dialog**

by

Jesse David Thomason

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

May 2018

Acknowledgments

I would like to thank my thesis advisor, Raymond J. Mooney, who has guided, collaborated with, and supported me during my graduate career, despite those two weeks we shared hotel rooms in Australia and that I have perfected mimicking his voice. I also thank Jivko Sinapov, who, as a postdoc at UT and then as a professor at Tufts, has been a valued collaborator, roboticist, mentor, and drinking partner. Thanks to the rest of my thesis committee: Peter Stone, Stefanie Tellex, and Scott Niekum, for their feedback and willingness to both digest this document and watch me compress its contents into an hour-long presentation at 9am. Thanks also to Stacy Miller, without whose administrative support I would no doubt have been kicked out on some technicality.

I would also like to thank my undergraduate advisors, Diane Litman and Jingtao Wang, for working with me. I became enamored with natural language processing and decided to pursue graduate education from the experiences I had in your labs. Your support was instrumental in my admission to UT, and thus in everything that has followed.

I have made close and valuable friends in the AI lab, a subset of which includes: Karl Pichotta, Piyush Khandelwal, Aishwarya Padmakumar, Yonatan Bisk (he sort of counts, right?), Amelia Harrison, Yinon Bentor, Dan Garrette, Elad Liebman, Stephen Roller, Wesley Tansey, Josiah Hanna, Subhashini Venugopalan, Rodolfo Corona, Nick Walker, Greg Durrett, Sam Barrett, Craig Corcoran, and Katie Genter. Thank you for the half-baked idea bouncing, your feedback on practice talks and paper drafts, and letting me prod you to leave the office (or the conference) to do anything else. Relatedly, this thesis is partially made possible by many members of the Learning Agents Research Group, especially those that like to get lunch and play racquetball.

Austin has been a beautiful and exciting place, and I'm grateful to my raucous rabble: Tyler Hunt, Victoria Hunt, Tyler Smith, Anisha Vichare, Tim Stamler,

John Kallaugher, Eric Holgate, Ishan Durugkar, Vance Miller, Julia Aponte, Victoria Stengel, Woody Austin, Lauren Marshall, and many of my aforementioned AI lab “friends.” I would have missed the best of Austin without you to stuff lunch holes, attend shows, have drinks, go hiking, make cocktails, and dance until last call with me.

Though physically separated, I’ve also had valuable ears from fellow graduate students elsewhere: Alex Ilitchev, Clai Morehead, Tyler Webb, and Elizabeth Kline. Thank you for listening to me complain constantly, offering words of encouragement, and sharing your own complaints. I think your experiences have each been more challenging than mine, and I believe in all of you. To my working friends across the country who so wisely chose not to pursue PhDs: James Michael Bridges, Austin Rowe, Drew Seel, Gabrielle Toborg, and Erin Millen, (and yes, Tyler, you too again) thank you for keeping up our yearly vacation. I hope we can do so for a long time yet.

Finally, I owe a great debt to two pillars of long-standing support. William Ryan Parker is the best friend, roommate, and server administrator a fledgling researcher could ask for, and belongs in many of the enumerations above. Kathy Wilson has always been present for me and encouraged my pursuits, even though as a kid I almost burned our house down that one time cooking literal grass from the yard in a skillet while she was sleeping. Plus all the snakes I brought home.

Continually Improving Grounded Natural Language

Understanding through Human-Robot Dialog

Jesse David Thomason, Ph.D.

The University of Texas at Austin, 2018

Supervisor: Raymond J. Mooney

As robots become ubiquitous in homes and workplaces such as hospitals and factories, they must be able to communicate with humans. Several kinds of knowledge are required to understand and respond to a human’s natural language commands and questions. If a person requests an assistant robot to *take me to Alice’s office*, the robot must know that Alice is a person who owns some unique office, and that *take me* means it should navigate there. Similarly, if a person requests *bring me the heavy, green mug*, the robot must have accurate mental models of the physical concepts *heavy*, *green*, and *mug*. To avoid forcing humans to use key phrases or words robots already know, this thesis focuses on helping robots understanding new language constructs through interactions with humans and with the world around them.

To understand a command in natural language, a robot must first convert that command to an internal representation that it can reason with. Semantic parsing is a method for performing this conversion, and the target representation is often semantic forms represented as predicate logic with lambda calculus. Traditional semantic parsing relies on hand-crafted resources from a human expert: an ontology of concepts, a lexicon connecting language to those concepts, and training examples of language with abstract meanings. One thrust of this thesis is to perform semantic parsing with sparse initial data. We use the conversations between a robot and human users to induce pairs of natural language utterances with the target semantic forms a robot discovers through its questions, reducing the annotation effort of creating training examples for parsing. We use this data to build more dialog-capable

robots in new domains with much less expert human effort (Thomason et al., 2015; Padmakumar et al., 2017).

Meanings of many language concepts are bound to the physical world. Understanding object properties and categories, such as *heavy*, *green*, and *mug* requires interacting with and perceiving the physical world. Embodied robots can use manipulation capabilities, such as pushing, picking up, and dropping objects to gather sensory data about them. This data can be used to understand non-visual concepts like *heavy* and *empty* (e.g. *get the empty carton of milk from the fridge*), and assist with concepts that have both visual and non-visual expression (e.g. *tall* things look big and also exert force sooner than *short* things when pressed down on). A second thrust of this thesis focuses on strategies for learning these concepts using multi-modal sensory information. We use human-in-the-loop learning to get labels between concept words and actual objects in the environment (Thomason et al., 2016, 2017). We also explore ways to tease out polysemy and synonymy in concept words (Thomason and Mooney, 2017) such as *light*, which can refer to a weight or a color, the latter sense being synonymous with *pale*. Additionally, pushing, picking up, and dropping objects to gather sensory information is prohibitively time-consuming, so we investigate strategies for using linguistic information and human input to expedite exploration when learning a new concept (Thomason et al., 2018).

Finally, we build an integrated agent with both parsing and perception capabilities that learns from conversations with users to improve both components over time. We demonstrate that parser learning from conversations (Thomason et al., 2015) can be combined with multi-modal perception (Thomason et al., 2016) using predicate-object labels gathered through opportunistic active learning (Thomason et al., 2017) during those conversations to improve performance for understanding natural language commands from humans. Human users also qualitatively rate this integrated learning agent as more usable after it has improved from conversation-based learning.

Table of Contents

Chapter 1	Introduction	1
Chapter 2	Background and Related Work.....	7
2.1	Instructing Robots in Natural Language	7
2.2	Learning Semantic Parsers.....	9
2.3	Language Grounding with Machine Perception	11
2.4	Language Grounding with Human-Robot Interaction	14
2.5	Polysemy and Synonymy in Language Understanding	16
Chapter 3	Learning to Interpret Natural Language Commands through Human-Robot Dialog	18
3.1	Motivation	18
3.2	Methods.....	22
3.3	Evaluation.....	25
3.3.1	Mechanical Turk Experiments	26
3.3.2	Segbot Experiments.....	29
3.4	Conclusion and Extensions	31
Chapter 4	Learning Multi-Modal Grounded Linguistic Semantics by Play- ing <i>I Spy</i>	33
4.1	Motivation	35
4.2	Methods.....	36
4.3	Evaluation.....	40
4.3.1	Setup.....	41
4.3.2	Quantitative Results.....	42
4.3.3	Qualitative Results	44
4.4	Conclusion.....	45
Chapter 5	Opportunistic Active Learning for Grounding Natural Language Descriptions	47
5.1	Motivation	48

5.2	Methods.....	50
5.2.1	Natural Language Grounding	51
5.2.2	Active Learning Dialog Policy.....	52
5.2.3	Robot Implementation	55
5.3	Evaluation.....	55
5.3.1	Experimental Results	57
5.3.2	Results	59
5.3.3	Discussion	60
5.4	Conclusion.....	60
Chapter 6 Multi-Modal Word Synset Induction		63
6.1	Motivation	63
6.2	Methods.....	66
6.2.1	Dataset	66
6.2.2	Synset Induction.....	68
6.3	Evaluation.....	71
6.3.1	Automated Evaluation	71
6.3.2	Human Evaluation	74
6.4	Conclusion.....	76
Chapter 7 Guiding Exploratory Behaviors for Multi-Modal Grounding of Linguistic Descriptions.....		80
7.1	Motivation	81
7.2	Methods.....	83
7.3	Evaluation.....	86
7.3.1	Learning object colors, weights, and contents.....	86
7.3.2	Learning words from everyday human users	91
7.4	Conclusion.....	97
Chapter 8 Jointly Improving Parsing and Perception for Natural Language Commands through Human-Robot Dialog.....		99
8.1	Motivation	101

8.2	Methods.....	103
8.2.1	Semantic Parser.....	105
8.2.2	Multi-modal Perception	112
8.2.3	Language Grounding	114
8.2.4	Dialog Policy	116
8.2.5	Learning from Conversations.....	125
8.3	Evaluation.....	128
8.3.1	Experiment Design	129
8.3.2	Mechanical Turk Evaluation	137
8.3.3	Physical Robot Implementation	152
8.4	Conclusion.....	154
Chapter 9	Future Work.....	156
9.1	Improving Language Understanding and Perception Modules	156
9.2	Human-Robot Dialog in Other Applications.....	162
Chapter 10	Conclusions	165
References	167

Chapter 1

Introduction

Humans use natural language to articulate their thoughts and intentions to other people. As robots become ubiquitous across diverse human environments, such as homes, offices, factory floors, and hospitals, the need for smooth human-robot communication grows. The language we use to discuss these spaces 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*). Natural language is the most natural channel for human-robot communication, and so robot natural language understanding needs to be robust to a wide-range of both human speakers and human environments.

Pre-programming robots with fixed language understanding components limits them, since environments and speakers can change. However, we can leverage the presence of these human interlocutors to gain additional learning signals for language understanding. Rather than force humans to use language robots around them can understand, robots in human environments can dynamically adapt—continually learning new language constructions and perceptual concepts as they are used in context. In this thesis, we discuss several directions of research in algorithms for continually improving embodied natural language understanding through human-robot conversation. The two main thrusts of this work are improving semantic, or abstract, understanding, and improving perceptual, or grounded, understanding.

Untrained human users providing natural language commands to robots expect world knowledge, perceptual knowledge, and semantic understanding from their robot interlocutors. Consider, for example, the command:

Move the light mug from Bob's office to the west, middle pod. (1.1)

Human utterances like the one above can be translated into semantic meanings. Given a semantic meaning, a robot can check against its world knowledge and per-

ception to resolve references to the real world and take actions appropriately in response. For example, one semantic interpretation of the above command is

$$\begin{aligned} &\text{move}(\text{the}(\lambda x.(\text{light}_{\text{weight}}(x) \wedge \text{mug}(x))), \\ &\quad \text{the}(\lambda y.(\text{office}(x) \wedge \text{owns}(\text{robert}, y))), \\ &\quad \text{the}(\lambda z.(\text{west}(z) \wedge \text{middle}(z) \wedge \text{pod}(z)))). \end{aligned} \tag{1.2}$$

The λ expressions used here are described in detail in Chapter 2, but for now, consider them variables that need to be solved for and refer to entities (objects, rooms, people, etc.) in the robot’s environment. Beyond translating the command into this semantic interpretation, *grounding* that interpretation against concepts in the world is a difficult problem that we explore over several chapters.

Translating human utterances to semantic meanings 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). For example, in (1.1) above, *Bob* refers to a person, `robert`, not an action verb, and *Bob’s office* is understood as a request for some space satisfying both being an `office` and belonging to `robert`. To learn a function between commands and semantic meanings, a *semantic parser* is often employed. Building a semantic parser can require expert annotation of a lexicon mapping between words and their meanings, an ontology of relevant concepts in the world, and training examples of natural language sentences paired with composed meanings. In our work, we use conversations robots have with humans to ameliorate this annotation effort, improving parser performance over time while starting out with sparse annotated resources (Chapter 3).

In order to converse about the environment they share with humans, robots must gather and maintain world knowledge through perception. Some world knowledge is ontological, such as the layout of a building, ownership relations between people and rooms, or assignments between patients and doctors in a hospital. This information can be created by humans and stored as a knowledge base accessible for language understanding. For example, in (1.1), the parse of *Bob’s office* can

be grounded against such a knowledge base to find the entity satisfying these constraints. Other world knowledge is perceptual, such as whether an object is a *mug*, where some movable objects were last seen, and whether an object can be picked up and moved somewhere else. This kind of dynamic knowledge changes over time, as new objects come in and out of the environment, and humans use previously unheard words to describe them. A service robot in a human environment needs both types of knowledge to understand and respond to human requests through dialog and actions.

Gathering correspondences between objects in the world and perceptual concepts, or predicates, applicable to those objects (e.g. *heavy*, *brown*, *mug*) is a time-consuming annotation effort if performed exhaustively. This effort is exacerbated by multi-modal representations of objects, such as their weight, relevant for the word *heavy*, since a human annotator has to physically interact with the object, removing the ability to crowd-source annotations online using photos alone. Rather than asking multiple humans to list every concept word applicable to an object in person, we explore the use of a human-robot conversation game, *I Spy*, to extract object-predicate relationships in a paradigm human users find interactive and enjoyable (Chapter 4).

In a deployed setting, especially a workplace, humans are less likely to spend time playing language games with their robot collaborators. Thus, necessary information like object-predicate relationships should be extracted from natural human-robot conversations. We explore the use of opportunistic active learning to ask questions about how concept words apply to objects that are *local* to the human and robot during an object retrieval task in which the human is requesting an object that is *remote*. If a human requests *an empty cup* and the robot knows its concept model for *empty* is poor, it can point to a local object whose emptiness it is unsure of and ask for the human’s opinion, improving the concept model with this additional labeled example. We find that humans are undeterred by both these and off-topic questions, in which the robot asks about irrelevant words as well in order to improve performance in future interactions with other users (Chapter 5).

The words used to describe object properties do not form a one-to-one map-

ping with underlying predicates. For example, the synonymous words *claret* and *purple* can reasonably refer to the same underlying predicate classifier. Additionally, the polysemous word *light* may refer to either a predicate for light coloration or a predicate for light weight. Robust robot perception must account for these ambiguous word senses, and will benefit from identifying synonymous senses (since it increases the amount of labeled training data for the underlying concept). In (1.1), for example, the system must find the correct sense of *light*, marked as a weight sense in the associated parse (1.2). We explore an unsupervised method for detecting word senses to address polysemy, then clustering those senses into word sense synonym sets to address synonymy. We compare representing concepts as simply their visual or linguistic embeddings, and demonstrate that using a multi-modal, visual plus linguistic representation leads to the most interpretable sense synonym sets (Chapter 6).

A major limitation of existing approaches to multi-modal language grounding is that a robot has to exhaustively explore training objects with a variety of actions when learning a new language concept. For example, to learn *squishy*, the robot may perform its whole suite of behaviors (e.g. looking, grasping, lifting, lowering, dropping, pushing, etc.) on objects labeled as *squishy* and *not squishy* to decide whether *squishy* applies. If the robot already knows *rigid*, however, it should be able to reason that since *squishy* is also a property related to the resistance the objects puts up to pressure, the behaviors that give discriminating information for *rigid* (pressing, grasping) should do so for *squishy* as well. Even without knowing similar words, it might be more productive to ask a human which behaviors are best for deciding *squishy* before performing all of them. We explore both strategies to build a prior over exploratory behaviors’ applicability to learn a new concept: examining that concept’s nearest neighbors in an embedding space and leveraging human annotations about behavior relevance. We demonstrate that both relevant behaviors for related concepts and human annotations lead to reductions in the exploration time required to achieve high recognition accuracy for new concepts (Chapter 7).

Together, the work in this thesis demonstrates individual steps towards an integrated approach to human-robot interaction— overviewed in Figure 1.1. These

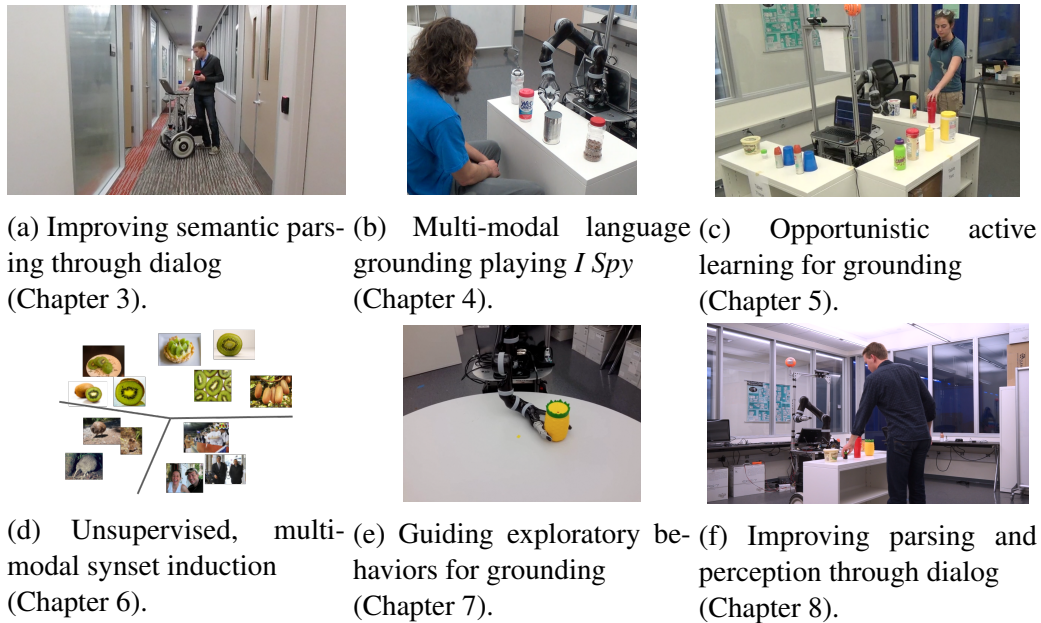


Figure 1.1: Works presented in this thesis.

steps assume little in-domain knowledge, enabling robots to acquire domain-specific vocabulary, syntax, and perceptual concepts on-the-fly from interaction with humans. Our integrated approach provides a stepping stone towards a lifelong learning agent for continually improving natural language understanding on a robotic platform from conversations with human users (Chapter 8).

In the remainder of this thesis, we first discuss the background of algorithms utilized for command understanding and dialog, as well as the substantial body of related work on semantic parsing for understanding human language commands, learning semantic parsers from corpora, language grounding from machine perception, language grounding involving human-robot interaction, and polysemy and synonymy in language understanding (Chapter 2). We continue with a discussion of work that uses weak supervision from human-robot dialog to generate training data for a semantic parser (Chapter 3). We then describe work that grounds predicates in multi-modal perception, allowing a robot to move beyond pre-written predicates (*office*, *possesses*) into perceptual concepts (*mug*, *heavy*) (Chapter 4). Addressing

label-gathering between objects and perceptual predicates, we summarize work on opportunistic active learning during human-robot conversations to improve predicate classifiers in an object retrieval task (Chapter 5). We then discuss work in noun phrase synonym sense set induction that utilizes multiple modalities of noun phrase context (Chapter 6), aimed at addressing the weak one-to-one assumption between surface forms of words and their underlying meanings. Aware of the physical and time costs involved in gathering multi-modal object representations through interactive behaviors, we also discuss work on using linguistic neighbors and human feedback to expedite object exploration when learning new predicates (Chapter 7). Combining semantic parser learning with multi-modal perception, we present a system that uses conversations to jointly improve language and perceptual understanding, bringing together the two main thrusts of this thesis (Chapter 8). We then discuss the future directions of this combined approach, as well as other next steps for the ideas explored (Chapter 9). Finally, we conclude by summarizing the insights and contributions provided by this thesis (Chapter 10).

Chapter 2

Background and Related Work

This thesis concerns the integration of semantic parser learning with robot perception for natural language understanding. We focus on methods that bypass the use of large corpora and substantial amounts of domain-relevant training data. This work took place largely within the Building-Wide Intelligence (BWI) project at the University of Texas at Austin (http://www.cs.utexas.edu/~larg/bwi_web/). We use Segway-based robots for embodied experiments, as described in Khandelwal et al. (2014, 2017).

In this chapter, we discuss existing work on instructing robots through natural language, a task our integrated system does with both semantic parsing and perception. We overview relevant work on learning semantic parsers, including work on inducing training data for semantic parses from conversations. We discuss language grounding as a task, grounding in machine perception, and grounding with additional signal from human-robot interactions. Finally, we overview natural language understanding tasks involving detecting polysemy and synonymy in language. We draw from and make contributions to each of these areas.

Instructing Robots in Natural Language

Instructing robots through natural language is essential for humans and robots to cooperate in shared environments. Researchers have focused on different aspects of human-robot communication, including using perception alongside semantic parsing for action understanding, and acquiring new actions from language descriptions in a perceivable environment.

Understanding the mutual environment is essential. Semantic parsing has been used as the understanding step in tasks like unconstrained natural language instruction, where a robot must navigate an unseen environment (Kollar et al., 2010; Matuszek et al., 2012b, 2013). Weak supervision can be used to improve such

parsers based on interactions with humans (Artzi and Zettlemoyer, 2013b). Simpler parsing approaches, such as transforming commands using semantic role labeling to form a meaning representation, require less training data at the cost of being less robust to language variation (Bastianelli et al., 2013). There have been focused efforts to understand human language commands with respect to a shared environment, such as the SemEval task of Dukes (2014). Work on semantic graphs connects environment referents probabilistically based on both sensor data and human language (Walter et al., 2013), while similar work additionally incorporates knowledge base information and conversation context (Mohan et al., 2013). One framework, based on generalized grounding graphs, acts to both understand human language requests about objects in the world and generate language requests regarding the shared environment (Tellex et al., 2014). Extensions of this framework can be used to memorize new semantic referents in a dialog, like “this is my snack” (Paul et al., 2017), or to reason about abstract sets and ordinalities (Paul et al., 2016). Other work uses vision techniques to build a knowledge base-like model of the shared environment. Then, language understanding is performed through semantic parsing followed by grounding against this vision-derived knowledge to resolve visual predicates (e.g. *red*) (Yang et al., 2014; Lu and Chen, 2015).

Recent work aims to translate human instructions directly to grounded behavior like route-following, skipping parsing in favor of sequence-to-sequence, end-to-end learning using neural methods (Mei et al., 2016). End-to-end neural methods are data hungry and fail to converge to useful solutions when training examples are sparse, such as in human-robot dialogs where gathering environment-specific interactions is costly. Past methods consider information jointly from the instructional utterance and the perceived environment to perform action understanding as a sequence (Misra et al., 2014) or hierarchy (Kuehne et al., 2014).

Going beyond action understanding, past work has also used semantic representations of utterances together with perception of objects in an environment to learn new manipulation behaviors from human instruction (She et al., 2014; Gemignani et al., 2015). Similarly focused, situated action learning for navigation maps human language instructions into programmatic behaviors that can be

used as modules for hierarchically-composed actions (Meriçli et al., 2014).

We present a robotic system that understands requests for actions in natural language. This understanding involves semantic parsing, but does not include action learning. Instead, we focus on methods robust to language variations and the use of perceptual predicates like *heavy mug* to describe objects in the real world, while executing pre-programmed actions (such as navigation, delivery, and object pick-and-place).

Learning Semantic Parsers

A semantic parser is a function from strings of words to a semantic meaning defined by some ontology (for example, Artzi and Zettlemoyer (2013a)). Formally, a parser $\mathcal{P} : \mathbb{P}(T) \times \mathcal{L}_{\mathcal{O}} \rightarrow S_{\mathcal{O}}$ takes in a sequence of word tokens $\vec{x} \in \mathbb{P}(T)$ in the power set of T , the set of all word tokens, and a lexicon $\mathcal{L}_{\mathcal{O}}$ for ontology \mathcal{O} and outputs a semantic parse $s \in S_{\mathcal{O}}$, the set of all semantic parses in ontology \mathcal{O} . An ontology \mathcal{O} defines a set of constants and predicates. Constants are things like items, places, and people. Predicates map from one or more constants to boolean true or false values (e.g. *red* is a predicate that tells whether an argument item constant is red in color). The lexicon \mathcal{L} is a data structure that contains information about how individual word tokens relate to that ontology, for example, that token *alice* refers to ontological constant `alice` and that possessive marker *'s* invokes predicate `owns` (see Figure 2.1). A semantic parse is a meaning representation in terms of ontological predicates and lambda expressions. A lambda expression allows meaning to abstract 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 some unique constant x that is both an `office` and belongs to `alice`, instantiating that constant as the argument for the `go` command.

Learning a semantic parser to perform these translations is non-trivial, but

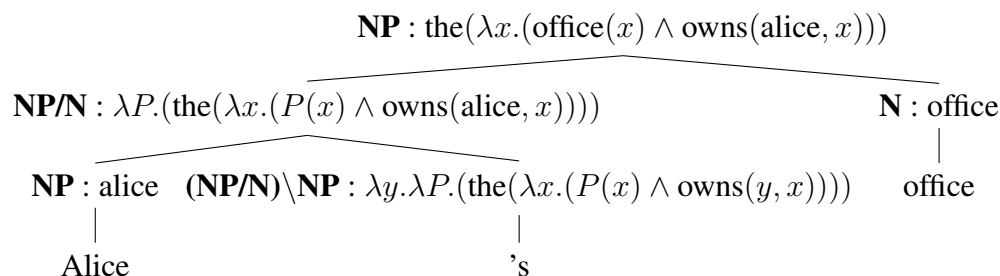


Figure 2.1: A CCG-driven λ -calculus parse of the expression *Alice’s office*.

using statistical parsing with compositional semantics makes it feasible. Many works, including our own, frame compositional semantics through combinatory categorial grammar (CCG) (Steedman and Baldridge, 2011), popular in both older and current deep syntactic parsing models (Lee et al., 2016; Lewis et al., 2016; Misra and Artzi, 2016). CCG adds a syntactic category to each token of the input sequence \vec{x} and then hierarchically combines those categories to form a syntax tree. These syntactic categories are typically a small set of basic ones, like N (noun) and NP (noun phrase), together with compositional entries, like PP/NP (a prepositional phrase formed after consuming a noun to the right). In a compositional semantics framework, the input lexicon \mathcal{L} contains entries mapping word tokens to both semantic meanings and CCG syntax categories. Semantic meanings can be composed according to the hierarchy established by a CCG syntactic parse tree to form a semantic parse that spans an entire token sequence. Figure 2.1 demonstrates this for *Alice’s office*.

Given data in the form of paired token sequences with their semantic parse trees, a statistical parser can be trained to produce those parses given novel sequences (Berant et al., 2013; Liang and Potts, 2015). However, annotating token sequences with entire parse trees is costly, and many works instead train with latent trees, requiring only the final desired semantic form (tree root) (Liang et al., 2011). Some works go beyond parser training and additionally incorporate ontology matching. This matching removes the restriction that the lexicon use predicates that exactly match the domain (Kwiatkowski et al., 2013). Others guide the semantic search with domain knowledge (Gardner and Krishnamurthy, 2017). Recent

methods perform this translation using sequence to sequence (Kočíský et al., 2016; Jia and Liang, 2016; Konstas et al., 2017) and sequence-to-tree (Dong and Lapata, 2016) deep models, but these are data-hungry and thus not feasible when we need to deploy in multiple domains (such as robots operating in an office versus a hospital).

The number of parameters to be learned in a semantic parser becomes large quickly even in a restricted domain, and annotating natural language with semantic forms is an expensive human effort. Consequently, some work in semantic parser learning has focused on overcoming data sparsity during training, for example by identifying incorrect parses using unskilled annotators and performing a kind of active learning which those incorrect parses are rectified by domain experts and added as additional data to tune a parser (Iyer et al., 2017). Other work directly queries human users to resolve ambiguities (He et al., 2016), or induces training examples automatically from existing conversational data (Artzi and Zettlemoyer, 2011).

In our work, we build on the idea of learning from existing conversations to gather parsing training examples from dialog (Chapter 3). We then use semantic parsing as an understanding step in a robotic system capable of grounding language referents in world knowledge as well as perceptual information (Chapter 8).

Language Grounding with Machine Perception

Commanding robots with language requires both semantic understanding and a subsequent grounding step where referents in the real world are connected to the language used to describe them. Mapping from a referring expression such as *the blue cup* to an object referent in the world is an example of the *symbol grounding problem* (Harnad, 1990). Symbol grounding involves connecting internal representations of information in a machine to real world data from its sensory perception. *Grounded language learning* bridges these symbols with natural language. Comparative studies have established that joint representations of language that consider some form of perception outperform text-only representations of word meaning (Silberer and Lapata, 2012). The grounded form of a semantic meaning is

called its *denotation*. Throughout this thesis, when we refer to grounding a semantic form, we mean finding its denotation in a given world environment.

Early work coupled vision with speech descriptions of objects to learn grounded semantics (Roy and Pentland, 2002). Recently, most work has continued grounding language in visual information. For grounding referring expressions in an environment, many approaches learn perceptual classifiers for words given some pairing of human descriptions and labeled scenes (A. Lazaridou and Baroni., 2014; Sun et al., 2013). Some approaches additionally incorporate language models into the learning phase (FitzGerald et al., 2013; Krishnamurthy and Kollar, 2013; Zitnick and Parikh, 2013; Matuszek et al., 2012a). Still other researchers have translated images into a distribution over possible descriptions, attempting to ground images in text, then doing query similarity in the resulting textual space to ground text in images (Guadarrama et al., 2015).

Recent work bypasses any explicit modeling of language predicates in favor of neural methods, such as localizing an object in a given image given a target query in natural language (Hu et al., 2016), or learning language concepts directly from sentence descriptions of images (Mao et al., 2015). A related task is to resolve ambiguities like prepositional phrase attachment in natural language by using associated images to gather additional information (Christie et al., 2016). A dataset for understanding compositional language using robot vision similarly uses neural methods to avoid directly modeling predicates and relations, instead learning a pipeline from commands to manipulations end-to-end, with mixed success (Bisk et al., 2016).

There has been some work on combining language with sensory modalities other than vision, such as audio (Kiela and Clark, 2015) or haptic (Chu et al., 2013; Gao et al., 2016b) signals. Additionally, researchers have explored the use of haptic and proprioceptive feedback from a robot arm to automatically learn to order objects by weight, height, and width (Sinapov et al., 2016). In other fields, such as psychology, multiple sensory modalities for understanding words have been studied and categorized (Lynott and Connell, 2009). In general, humans tend to use senses beyond vision to understand how words connect to properties of their environments.

Recent work uses co-training between two modalities— RGB and depth information— to automatically learn grounded object attributes using deep networks in an unsupervised fashion, bootstrapped from a small number of supervised examples (Cheng et al., 2016). Other approaches work with knowledge base structures directly, learning to map streams of text references of world states to knowledge base entries describing those states (Liang et al., 2009). A less supervised approach first induces possible attributes by clustering using feature-based representations of objects, colors, and faces, then ties the discovered clusters to language labels post-hoc (Alomari et al., 2017a). A recent survey examines this broad space of multi-modal representations of objects and concepts in machine learning (Baltrušaitis et al., 2017).

A major limitation of these multi-modal approaches is that they require a robot to perform exhaustive object exploration, i.e., a robot must explore each object with some fixed number of exploratory actions during which it records non-visual sensory data (or this data must come from some other external corpus). One possible way to address this problem is to estimate the relevance of each behavior for the task of learning a novel predicate or category. Recent work has used word embeddings to predict unseen verb causality information from seen verbs (Gao et al., 2016a), and unseen noun affordances from seen nouns (Fulda et al., 2017). These embeddings can be jointly formed from textual and visual input for more robust composition (Silberer et al., 2017). We later show that such embeddings can be used to guide behaviors during language grounding (Chapter 7).

In robotics, language can function as a signal for navigation and question-answering tasks, but many existing lines of research do not use human-robot dialog as part of this process. For navigation, simple strategies can learn to ground nouns and prepositions describing entities along a path to be followed in a physical environment (Barrett et al., 2015). Other work distinguishes between affordances discovered during exploration (useful for grounding object noun categories) and visual information (useful for learning attributes) (Yürüten et al., 2013). Recent work introduces a task paradigm around this idea, embodied question answering (Das et al., 2017a). In that work, a simulated agent is given an unambiguous question in tem-

plated language and continuous RGB camera input and must produce a sequence of driving actions to eventually generate an answer to the question from one among several hundred possibilities. Though that work does not incorporate dialog, similar neural approaches have been used to train conversational robot-robot conversational agents, and may be applicable for human-robot dialog (Das et al., 2017b). Similar thrusts utilize a visually-grounded navigation dataset based on real-world environments (Anderson et al., 2017), and a dataset of hundreds of simulated 3D home layouts (Brodeur et al., 2017).

In our work, we introduce multi-modal perception for a robotic system using vision together with haptics, audio, and proprioception that uses conversations with humans as a learning signal (Chapter 4), and investigate strategies to reduce the exploration time required to learn new perceptual concepts (Chapter 7).

Language Grounding with Human-Robot Interaction

Machine perception is necessary for human-robot interaction. It can also be improved by that interaction. A number of researchers have focused on solving the symbol grounding problem for situated robots by leveraging their interactions with the humans they are working to understand.

One line of existing work focuses on gathering data from human demonstration and speech to learn language grounding. Using unscripted human descriptions of objects together with their deictic hand gestures, researchers train a grounding system for identifying referent objects (Matuszek et al., 2014). Similar work used only speech from humans describing objects to achieve one-shot learning of object attributes and names (Perera and Allen, 2013). Other researchers have focused on learning unary properties of objects (*red*) together with relational (*taller*) and differentiating (*differ by weight*) properties of objects by exploring them with a robotic arm, provided properties and relational labels as human supervision after that exploration (Sinapov et al., 2014b). These kinds of relations can also be orthogonally learned from text data (Forbes and Choi, 2017), and there is potential to combine these approaches outside the scope of this thesis.

Closer to our work, some researchers gather data for perceptual grounding using interaction with a human interlocutor. This combination of dialog and perception affords new opportunities for smart interactions, such as the robot asking questions targeting weaknesses in its understanding (Thomason et al., 2016). Early work on learning to ground object attributes and names using dialog framed the data gathering phase as a *20 Questions*-style game where a robot tried to guess a target object by asking narrowing questions (e.g. *is it red?*) (Vogel et al., 2010). Other research focused on acquiring perceptual understandings through a command-, rather than game-based environment (Dindo and Zambuto, 2010). Researchers have carried this idea to more complete systems with both perceptual grounding and action learning capability for identifying and manipulating objects, where the agent can request more information about uncertain concepts (Mohan et al., 2012). Agents can also be taught spatial language through a curriculum-learning paradigm, where increasingly difficult spatial relational language is introduced (Spranger and Steels, 2015). Similar to other work that learns from demonstration and description offline, Kollar et al. (2013) studies the joint acquisition of perceptual classifiers and language understanding in an interactive setting. Focused efforts have begun studying one-shot object attribute learning (Krause et al., 2014), including actively exploring an environment with a physical robot and assigning previously unseen attributes to objects in an online fashion (Tucker et al., 2017). Referring expressions and actions can also be disambiguated by considering perceptual grounding information, using the real-world to rectify errors at the parsing level (Bastianelli et al., 2016).

Other work aims to address perceptual mismatch between humans and robots, since our sensory systems differ, and delineations present in the one may be undetectable in the other (Liu et al., 2014; Liu and Chai, 2015). In the vein of game-based data gathering, researchers have framed learning attribute classifiers for objects as an *I Spy* game in which a human describes a target object among several options to a robot and confirms when the correct one is identified (Parde et al., 2015). In another game-like approach, users offered commands and selected from a set of world-states that would result from different system understandings of those commands (Wang et al., 2017). Other object identification work has focused on

integrating language with gesture, bypassing perception in favor of language co-occurrences with particular objects (Whitney et al., 2016), or using language and gesture to decide when to ask clarification questions (Whitney et al., 2017).

In our work, we bootstrap a perception system using an interactive *I Spy* game (Chapter 4), and in conversations during an object identification task (Chapter 5). We then introduce continual learning to the perceptual component of a grounded dialog agent (Chapter 8).

Polysemy and Synonymy in Language Understanding

Semantic understanding in language is complicated by words that have multiple, distinct meanings, and by sets of words with the same underlying meaning. A sense inventory for words, such as WordNet (Fellbaum, 1998), structures word meaning into senses which can be taken on by one or more words. Words that refer to the same sense are called synonymous. A word which refers to multiple meanings across its different senses is polysemous. Sets of word senses are called *synsets*.

Word sense induction (WSI) is the task of determining whether an individual word type, such as *bat*, is polysemous, and what its underlying senses are. Given a set of word senses and a word in context, the task of *word sense disambiguation* (WSD) is to choose which sense the word expresses in the given context (Navigli, 2009). WSD has applications across language understanding, since the multiple meanings of a word can be arbitrarily different and unrelated.

Traditional work on WSI discovers senses for a word by clustering the textual contexts in which it occurs (Yarowsky, 1995; Pedersen and Bruce, 1997; Schutze, 1998; Bordag, 2006; Navigli, 2009; Manandhar et al., 2010; Reisinger and Mooney, 2010; Di Marco and Navigli, 2013). The multiple meanings for *bat* can be recognized as two clusters: one a set of contexts with words like *cave* and *flew*; and another with words like *baseball* and *strike*. Other notions of context can be used to discover word senses, such as images the word is used to describe. Some previous work has recognized the value of perceptual word senses for tasks such as

image-segment labeling (Barnard and Johnson, 2005) and image retrieval from text queries (Lucchi and Weston, 2012).

Past work has used visual information to disambiguate word senses, but assumes the senses of each word are known in advance (Barnard and Johnson, 2005). Using both textual and visual information to perform WSI has been done as well, but on datasets where every input word is known in advance to be polysemous (Loff et al., 2006; Saenko and Darrell, 2008).

Some work performs co-clustering in separate textual and visual spaces, treating textual clusters as word senses and visual clusters as iconographic senses (viewpoint changes, color differences, etc.) that offer a finer-grained distinction than word senses (Chen et al., 2015). Other work attempts to handle the polysemy of attributes like *red* automatically by considering the compositions they participate in, e.g. *red wine* versus *red tomato* (Misra et al., 2017).

We use polysemy and synonymy detection to induce synsets in an unsupervised way by combining linguistic and visual context (Chapter 6). This allows us to discover, for example, the polysemous meanings of *kiwi* (fruit, bird, person), and the synonymous meanings of the fruit sense of *kiwi* and *chinese grapefruit*. In the future, these methods could be applied to words humans use to describe properties of objects in the world, with those objects providing multi-modal context for the meanings of the words, discussed in Chapter 9.

Chapter 3

Learning to Interpret Natural Language Commands through Human-Robot Dialog

Intelligent robots need to understand requests from naive users through natural language. In this chapter we discuss a dialog agent for mobile robots that understands human instructions through semantic parsing, actively resolves ambiguities using a dialog manager, and incrementally learns from human-robot conversations by inducing training data from user paraphrases, as presented in (Thomason et al., 2015). Our dialog agent is implemented and tested both on a web interface with hundreds of users via Mechanical Turk and on a mobile robot over several days, tasked with understanding navigation and delivery requests through natural language in an office environment. In both contexts, we observe significant improvements in user satisfaction after learning from conversations.

Figure 3.1a gives an example conversation when the dialog agent had only slim initial training data for language understanding. The clarification questions in this conversation let the agent learn that *calender* and *planner* mean *calendar*. This work allows us to translate utterances into logical forms, and is the first step to resolving (1.1) in our work to integrate parsing and perception in an embodied robotic system.

Motivation

Many existing natural language instruction approaches either use simple language understanding (e.g., keyword search), or large corpora of hand-annotated training data to pair language with robot actions or action language. The former cannot account for language variation. The latter requires gathering annotated corpora, which can be expensive and can only account for variation observed in the training data. This chapter addresses these shortcomings via a dialog agent that

TASK TO COMPLETE

Dave Daniel wants the item in slot 5.

ROBOT How can I help?
 YOU please bring the item in slot 5 to dave daniel
 ROBOT To whom should I bring something?
 YOU dave daniel
 ROBOT What action did you want me to take involving dave daniel?
 YOU 5
 ROBOT What should I bring to dave daniel?
 YOU a calander
 ROBOT I'm sorry, but I couldn't pinpoint what you meant by that.
 ROBOT What should I bring to dave daniel?
 YOU a day planner
 ROBOT I'm sorry, but I couldn't pinpoint what you meant by that.
 ROBOT What should I bring to dave daniel?
 YOU calendar
 ROBOT You want me to bring calendar to dave daniel?
 YOU yes
 ROBOT I thought so
 ROBOT Happy to help
 YOU

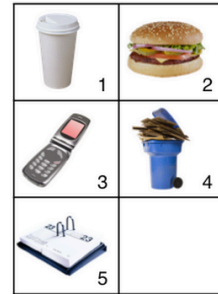
Final Task

DIRECTORY

People:

Alice Ashcraft; Secretary
 Francis ("Frannie") Foster
 Robert ("Bob") Brown
 Carol Clark, PhD
 Dave Daniel, PhD
 George Green; Intern
 Evelyn ("Eve") Eckhart
 Mallory Morgan; Director
 Peggy Parker, PhD
 Walter Ward; Supervisor

Items available to robot:



(a) The Mechanical Turk worker instructs the dialog agent to deliver a calendar item to Dave Daniel (left). Items are given pictorially (right) so that workers have to describe them with natural language.

Expression	Logical Form
<i>please bring the item in slot five to dave daniel</i>	$\text{action}(\text{bring}) \wedge \text{patient}(\text{calendar})$ $\wedge \text{recipient}(\text{dave})$
<i>calander</i>	$\text{patient}(\text{calendar})$
<i>a day planner</i>	$\text{patient}(\text{calendar})$

(b) The expression and semantic form pairs induced from the worker's responses to questions in the clarification dialog; these are used when retraining the semantic parser to learn new words and phrases.

Figure 3.1: This abridged conversation took place when the system had access to only the initial, small data (no additional training examples yet), and allows the system to learn several new words and phrases.

communicates with users through natural language while learning semantic meanings from conversations.

Our dialog agent integrates a semantic parser producing logical form representations of user utterances with a dialog manager that maintains a belief-state for the user’s goal. The agent starts with a few training examples for the parser and induces more during clarification dialogs with ordinary users. When the agent understands a user goal, it pairs the logical form representing that goal with previously misunderstood utterances in the conversation to form new training examples for the semantic parser (3.1). This allows the agent to incrementally learn new semantic meanings for previously unseen words and phrases. This approach is more robust than keyword search and requires little initial data. Further, it can be deployed in any context where robots are given high-level goals in natural language.

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

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

Natural language instruction can dictate a series of actions to a robot. Some approaches pair robot actions with language descriptions, then build models that map language instructions to action sequences (Misra et al., 2014; Tellex et al., 2011). We are concerned with interpreting high-level instructions rather than action sequences and don't rely as they do on a well-trained initial parser (Klein and Manning, 2003). Another approach enables a robot to learn a sequence of actions and the lexical items that refer to them from language instruction and dialog (She et al., 2014). We focus on acquiring new lexical items to overcome linguistic variation, rather than for referring to and teaching action sequences.

Other researchers have used semantic parsing to facilitate natural language instruction for robots. One approach learns a parser to map natural-language instructions to control language (Matuszek et al., 2013). We build on such approaches by augmenting our parser with new data in an incremental fashion from dialog. We also use world knowledge to ground natural language expressions. Other work uses restricted language and a static, hand-crafted lexicon to map natural language to action specifications (Matuszek et al., 2012b).

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

Our process of automatically inducing training examples from conversations is partly inspired by Artzi and Zettlemoyer (2011). They used logs of conversations that users had with an air-travel information system to train a semantic parser for understanding user utterances. Our approach to learning is similar, but done incre-

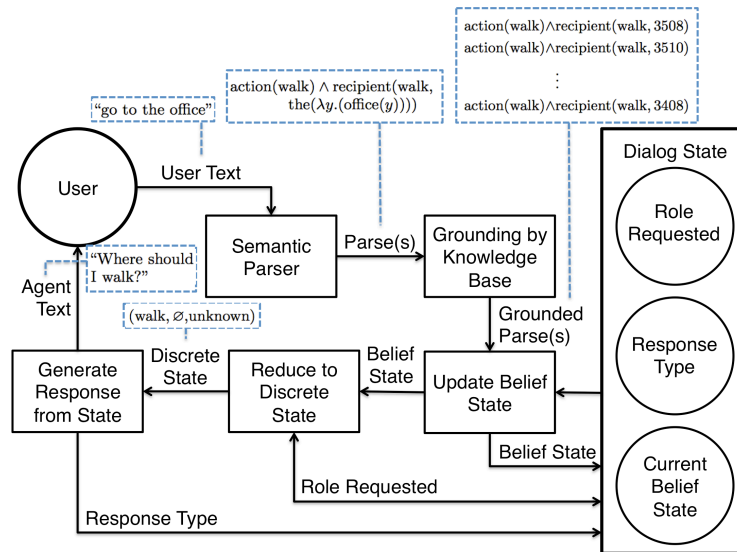


Figure 3.2: Dialog agent workflow processing user command *go to the office*.

mentally from conversations the agent has with users, and our training procedure is integrated into a complete, interactive robot system.

Methods

A human user first gives a command to our dialog agent, then the agent can ask clarification questions (Figure 3.2). The agent maintains a belief state about the user’s goal. When it is confident in this state, the dialog ends and the goal is passed on to the robot or other underlying system.

The agent produces a semantic form for each user utterance. In this work, we use the University of Washington Semantic Parsing Framework (SPF) (Artzi and Zettlemoyer, 2013a), a system for mapping natural language to meaning representations using λ -calculus and combinatory categorial grammar (CCG).

To get the system *off the ground* we initialize the parser with a small seed lexicon and then train it on a small set of supervised utterance/logical-form pairs. We use a seed lexicon of 105 entries (40 of which are named entities) and a training set of only 5 pairs.

The agent maintains a belief state about the user goal with three components: `action`, `patient`, and `recipient`. Each component is a histogram of confidences over possible assignments. The agent supports two actions: walking and bringing items, so the belief state for `action` is two confidence values in $[0, 1]$. `recipient` and `patient` can take values over the space of entities (people, rooms, items) in the knowledge base as well as a null value \emptyset .

Multiple meaning hypotheses may be generated from a user utterance. Consider:

expression *go to the office*

logical form `action(walk) ∧ recipient(the(λy.(office(y))))`

For n offices, this logical form has n groundings producing different meanings (see Figure 3.2). The agent can be confident that walking is the task, but its confidence in the n meanings for `recipient` is weakened. We use a confidence update based on the number k of hypotheses generated to track the agent's confidence in its understanding of each component of the request. For a user-initiative (open-ended) statement like this one, the agent updates all components of the belief state. For each candidate hypothesis $H_{i,c}$, with $0 \leq i < k$, $c \in \{\text{action}, \text{patient}, \text{recipient}\}$, the agent updates:

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

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

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

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

System-initiative responses are associated with a particular requested component. These can take the form of confirmations or prompts for components. For the former, user affirmation will update the confidence of all mentioned values to

1. For the latter, the positive and negative updates described above operate only on the requested component.

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

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

Each component c of S' is selected by choosing either T_c or *unknown* with probability $\operatorname{conf}(c = T_c)$. The component c with the minimum confidence is chosen as the role to request. If *unknown* is chosen for every component, the role requested is *all*. If *unknown* is chosen for no component, the role requested is *confirmation*. Some policy responses are given in Table 3.1. If each of the confidence values inspected during this process exceeds α , the conversation concludes. In all experiments, parameters $\alpha = 0.95, \gamma = 0.5$ were used.

Our agent induces parsing training examples from conversations with users to learn new lexical items. It uses dialog conclusions and explicit confirmations from users as supervision. The semantic parser in Figure 3.1a does not know the misspelling *calendar* and *calender*, the word *planner*, or number 5. When the user requests *item in slot 5* be delivered, it only confidently detects the `action`, *bring*, of the user’s goal. The `recipient`, *Dave Daniel*, is clarified by a system-initiative question. When the agent asks for confirmation of the `action`, the user does not deny it, increasing the agent’s confidence. While clarifying the `patient`, the user implicitly provides evidence that *calender*, *planner*, and *calendar* are the same. When two or more phrases are used in the same sub-dialog to clarify an argument, the eventual logical form selected is paired with the earlier surface forms for retraining. Figure 3.1 gives this example.

User-initiative responses generate similar alignments. One user’s conversation began *please report to room 3418*, which the agent could not parse because of the new word *report*. The agent understood the re-worded request *go to room 3418*,

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

Table 3.1: Representative subset of our policy π for mapping discrete states S' to questions.

and the former sentence was paired with the logical form of this latter for training. When the retraining procedure explored possible semantic meanings for *report*, it found a valid parse with the meaning of *go*, $S/PP : \lambda P.(action(walk) \wedge P(walk))$, and added it to the parser's lexicon. This meaning says that *report* should be followed by a prepositional phrase specifying a target for the walking *action*.

Evaluation

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

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

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

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

Mechanical Turk Experiments

The web interface shown in Figure 3.1a was used to test the agent with many users through Mechanical Turk. We performed incremental learning in batches to facilitate simultaneous user access. We assigned roughly half of users to the test condition and the other half to the train condition per batch. After gathering train and test results from a batch, we retrained the parser using the train conversation data. We repeated this for 3 batches of users, then we gathered results from a final testing batch in which there was no need to gather more training data. We used user conversations for retraining only when they achieved correct goals.

Navigation: Users were asked to send the robot to a random room from the appropriate train or test goals with the prompt *[person] needs the robot. Send it to the office where [s]he works*. The referring expression for each person was chosen from: full names, first names, nicknames, and titles. In this task, the corresponding

office number was listed next to each name, and the *items available* were not shown.

Delivery: Users were asked to tell the robot to assist a person with the prompt *[person] wants the item in slot [number]*. The $(person, item)$ pairs were selected at random from the appropriate train or test goals. To avoid linguistic priming, the items were given pictorially (Figure 3.1a).

For each train/test condition, we gathered responses from an average of 48 users per batch. Figure 3.3 (Top) shows the mean survey-question responses across test batches. We used an unpaired Welch’s two-tailed t -test to determine whether these means differed significantly. By batch 2, users felt that the agent understood them more than in batch 0. By batch 3, they felt that it frustrated them less. The dialog agent became more understandable and likable as a result of the semantic parser’s learning, even though it had never seen the test-batch users’ goals.

To determine whether learning reduced the number of utterances (turns) a user had to provide for the system to understand their goal, we counted user turns for dialogs where the user and agent agreed on the correct goal (Figure 3.3 (Bottom)). The performance between goal types differs. Because of an uncooperative worker specifying *go* as a user-initiative command eventually resolved as navigating to a particular person’s office, this word picked up a polysemous lexical entry during retraining meaning to go to that specific office, rather than whatever else was specified. In the future, creating negative examples for the parser from misunderstandings may ameliorate this (e.g. if the command does not lead to coherent grounded forms, create a negative example for the parser specifying that the next-best semantic form should be returned next time, since the top form was not helpful) and unlearn bad rules. In expectation, this only adds a single turn (the worker clarifying the meaning of *go*). By contrast, learning successfully, and statistically significantly, reduced the turns needed to understand multi-argument delivery goals. These goals are more difficult to specify, since they involve two argument (the patient item and recipient person), so there is more to gain from continued semantic parser learning. For example, the agent can learn *bring A to B* as well as *bring B A*, nicknames and titles for people, and multiple referring expressions for items given pictorially.

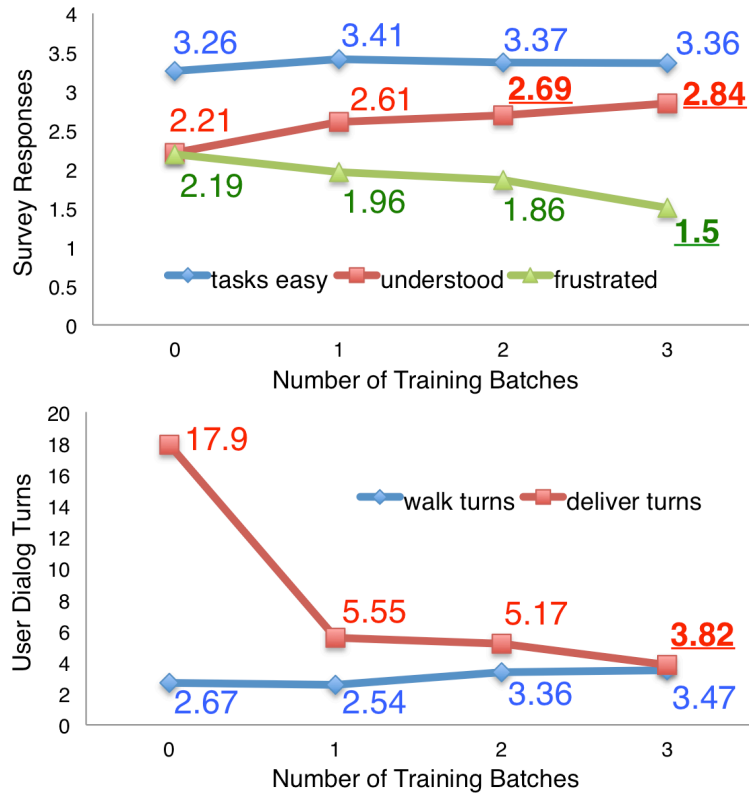


Figure 3.3: **Top:** Average Mechanical Turk survey responses across the four test batches. **Bottom:** Mean user turns in Mechanical Turk dialogs where the correct goal was reached. Means in underlined bold differ significantly ($p < 0.05$) from the batch 0 mean.

With respect to users’ free-form feedback, in testing batch 0, several enjoyed their conversations (*This was fun!! Wish it were longer!*). Several also commented on the small initial lexicon (*It was fun to try and learn how to talk to the robot in a way it would understand*). The responses by testing batch 3 had similarly excited-sounding users (*I had so much fun doing this hit!*). At least one user commented on the lexical variation they observed (*The robot fixed my grammatical error when I misspelled ‘calender’ Which was neat*). In addition to learning misspelling corrections and new referring expressions, the agent learned to parse things like *item in slot n* by matching *n* to the corresponding item and collapsing the whole phrase to

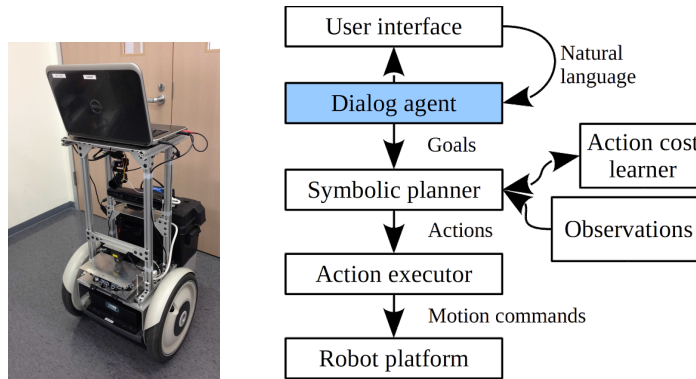


Figure 3.4: **Left:** Robot platform (Segbot) used in experiments. **Right:** Segbot architecture, implemented using Robot Operating System (ROS).

this meaning.

Segbot Experiments

The agent was integrated into a Segway-based robot platform (Segbot) as shown in Figure 3.4 (Left) using the Robot Operating System (ROS) (Quigley et al., 2009). The robot architecture is shown in Figure 3.4 (Right). Users interacted with the agent through a graphical user interface by typing in natural language. The agent generated queries to a symbolic planner formalized using action language \mathcal{BC} (Lee et al., 2013) from user goals.

For testing, users were given one goal from the navigation and delivery tasks, then filled out the survey. The task prompts included the directory panels used in the Mechanical Turk experiments pairing names and office numbers and showing items available to the robot for delivery (Figure 3.1a).

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

We then allowed the agent to perform incremental learning for four days in our office space. Students working here were encouraged to chat with it, but were

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

Table 3.2: Average Segbot survey responses from the two test groups and the proportion of task goals completed. Means in bold differ significantly ($p < 0.05$). Means in italics trend different ($p < 0.1$).

not instructed on how to do so beyond a panel displaying the directory information and a brief prompt saying the robot could only perform *navigation and delivery tasks*. Users in test conditions did not interact with the robot during training. After understanding and carrying out a goal, the robot prompted the user for whether the actions taken were correct. If they answered *yes* and the goal was not in the test set, the agent retrained its semantic parser with new training examples aligned from the conversation. A video demonstrating the learning process on the Segbot can be found at: <https://youtu.be/FL9ThJQOzb8>.

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

During training, the robot understood and carried out 35 goals, learning incrementally from these conversations. Table 3.2 compares the survey responses of users and the number of goals users completed of each task type in the `Init Test` and `Trained Test` groups. We allow human users to opt-out of a task if they feel frustrated with the system, and measure how often they are able to complete the target goal (proportion of successful goals) before and after the semantic parser learns from the 35 natural conversations had during the allowed training days.

We note that there is significant improvement in user perception of the robot’s

understanding, and trends towards less user frustration and higher delivery-goal correctness. Goal success rates mirror the results for dialog lengths in the large-scale Mechanical Turk experiments. In particular, navigation success remains the same, while three times more in-person human users are able to complete the delivery goal after retraining from natural interactions. Though users did not significantly favor using the robot for tasks after training, several users in both groups commented that they would not use guidance only because the Segbot moved too slowly.

Conclusion and Extensions

In this chapter, we introduce a dialog agent that improves its language understanding without requiring a large corpus of annotated data. This work uses a static, hand-coded dialog policy to resolve confusion the agent has when taking commands from human users. This policy centers around slot-filling for commands and their known arguments (e.g. Table 3.1), done by estimating a discrete belief about the user goal from an otherwise continuous one.

We also explore reinforcement learning to learn a better, Partially-Observable Markov Decision Process (POMDP) dialog policy (Young et al., 2013) that considers a continuous belief state alongside improving semantic parsing from conversations using the techniques presented in this chapter (Padmakumar et al., 2017). We find that jointly retraining the parser and dialog strategy, using conversation length as a metric for dialog success, outperforms retraining either in isolation. Changes to the parser affect the which dialog strategy is most efficient, since changes to the parser affect the underlying space over which the POMDP driving the dialog operates. The learned dialog policy selects which questions to ask based on a continuous agent belief state, in contrast with the work in this chapter, which samples a discrete belief state that is then fed into a static, template-based dialog-policy. In effect, in this chapter dialog control is handled by probabilistic sampling, which is replaced with a reinforcement-learning based POMDP agent for selecting a question in follow-up work. Such a policy converges to user goals more quickly by accurately taking an expected shortest dialog path from the current understanding

of the user's goal to a complete and confident understanding.

By learning to translate natural language phrases into semantic forms, and improving that translation over time by improving parser performance using data induced from conversations, the work in this chapter is the first step towards resolving commands, like (1.1), in our work to integrate parsing and perception in an embodied robotic system, discussed in Chapter 8. In the following chapter, we introduce multi-modal perception to go beyond visual sensory data to ground words like *light* and *mug*.

Chapter 4

Learning Multi-Modal Grounded Linguistic Semantics by Playing *I Spy*

Grounded language learning bridges words like *red* and *square* with robot perception. The vast majority of existing work in this space limits robot perception to vision. We build perceptual models that use haptic, auditory, and proprioceptive data acquired through robot exploratory behaviors to go beyond vision. In this chapter, we describe a system that learns to ground natural language words describing objects using supervision from an interactive human-robot *I Spy* game, as presented in (Thomason et al., 2016).

While corpora like ImageNet (Deng et al., 2009) can provide a large set of labeled images to learn classifiers for words and noun phrases, properties like *heavy* are grounded in non-visual space. Annotating a similarly large body of objects with non-visual properties and gathering robot perception or even features (like weight) about them is costly and does not generalize across different robotic platforms. We use the *I Spy* game as a paradigm to bootstrap a perceptual grounding system since it is fun for human users and requires less labor than strict annotation. This work provides a blueprint for perceptual grounding of the predicates *light* and *mug* from the earlier example (1.1). We later continually refine this bootstrapped perception in a fully integrated robotic system that uses dialog to clarify misunderstandings (Chapter 8).

In the *I Spy* game, the human and robot take turns describing one object among several, then trying to guess which object the other has described (Figure 4.1 (Left, Center)). We demonstrate that our multi-modal system for grounding natural language outperforms a traditional, vision-only grounding framework by comparing the two on the *I Spy* task.



Figure 4.1: **Top-Left:** the robot guesses an object described by a human participant as *silver, round, and empty*. **Top-Right:** a human participant guesses an object described by the robot as *light, tall, and tub*. **Bottom:** objects used in the *I Spy* game divided into the four folds, from fold 0 on the left to fold 3 on the right.

Motivation

Robots need to be able to connect language to their environment in order to discuss real world objects with humans. Learning to map from referring expressions such as *the blue cup* to an object referent in the world is an example of grounded language learning. Early work on grounded language learning enabled a machine to map from adjectives and nouns such as *red* and *block* to objects in a scene through vision-based classifiers (Roy, 2001). Most work has focused on grounding these kinds of language predicates through visual information. However, other sensory modalities such as haptic and auditory are also useful in allowing robots to discriminate between object categories (Sinapov et al., 2014b). This chapter explores grounding language predicates by considering visual, haptic, auditory, and proprioceptive senses.

A home or office robot can explore objects in an unsupervised way to gather perceptual data, but needs human supervision to connect this data to language. Learning grounded semantics through human-robot dialog allows a system to acquire the relevant knowledge without the need for laborious labeling of numerous objects for every potential lexical descriptor. A few groups have explored learning from interactive linguistic games such as *I Spy* and *20 Questions* (Parde et al., 2015; Vogel et al., 2010); however, these studies only employed vision. We use a variation on the children’s game *I Spy* as a learning framework for gathering human language labels for objects to learn multi-modal grounded lexical semantics (Figure 4.1).

There is a large body of work on grounding language for robots using visual information. For grounding referring expressions in an environment, many learn perceptual classifiers for words given some pairing of human descriptions and labeled scenes (Liu et al., 2014; Malinowski and Fritz, 2014; Mohan et al., 2013; Sun et al., 2013; Dindo and Zambuto, 2010; Vogel et al., 2010). Some approaches additionally incorporate language models into the learning phase (Spranger and Steels, 2015; Krishnamurthy and Kollar, 2013; Perera and Allen, 2013; Matuszek et al., 2012a). Incorporating a language model also allows for more robust generation of robot referring expressions for objects, as explored in (Tellex et al., 2014). In

general, referring expression generation is difficult in dialog (Fang et al., 2014). Since we focus on comparing multi-modal to vision-only grounding, our method uses simple language understanding and constructs new predicate classifiers for each unseen content word used by a human playing *I Spy*, and our basic generation system for describing objects is based only on these predicate classifiers.

Outside of robotics, there has been some work on combining language with sensory modalities other than vision, such as audio (Kiela and Clark, 2015). Unlike that line of work, our system is embodied in a learning robot that manipulates objects to gain non-visual sensory experience.

Including a human in the learning loop provides a more realistic learning scenario for applications such as household and office robotics. Past work has used human speech plus gestures describing sets of objects on a table as supervision to learn attribute classifiers (Matuszek et al., 2014; Kollar et al., 2013). Recent work introduced the *I Spy* game as a supervisory framework for grounded language learning (Parde et al., 2015). Our work differs from these by using additional sensory data beyond vision to build object attribute classifiers. Additionally, in our instantiation of the *I Spy* task, the robot and the human both take a turn describing objects, where in previous work only humans gave descriptions (Parde et al., 2015).

Methods

The robot used in this study was a Kinova MICO arm mounted on top of a custom-built mobile base which remained stationary during our experiment. The robot’s perception included joint effort sensors in each of the robot arm’s motors, a microphone mounted on the mobile base, and an Xtion ASUS Pro RGBD camera. The set of objects used consisted of 32 common household items including cups, bottles, cans, and other containers, shown in Figure 4.1 (Right). Some of the objects contained liquids or other contents (e.g., coffee beans) while others were empty. Contemporary work gives a more detailed description of this object dataset (Sinapov et al., 2016), but we briefly describe the exploration and modalities below.

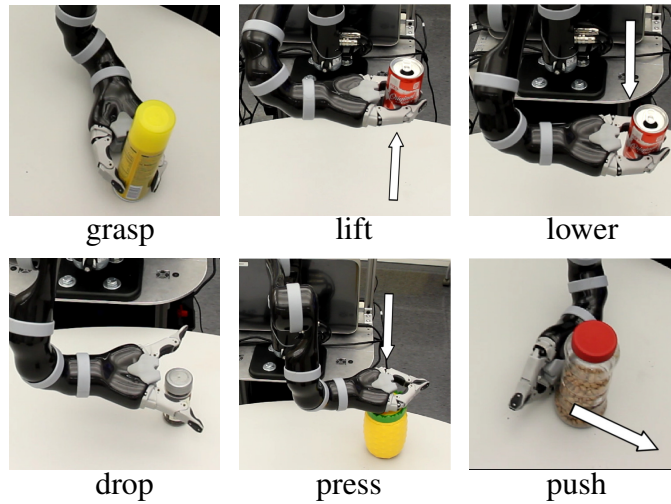


Figure 4.2: The behaviors the robot used to explore the objects. In addition, the *hold* behavior (not shown) was performed after the *lift* behavior by simply holding the object in place for half a second.

Prior to the experiment, the robot explored the objects using the methodology described by (Sinapov et al., 2014a), and the dimensionality of the raw auditory, haptic, and proprioceptive data were reduced comparably (final dimensionality given in Table 4.1). In our case, the robot used 7 distinct actions: *grasp*, *lift*, *hold*, *lower*, *drop*, *push*, and *press*, shown in Figure 4.2. During the execution of each action, the robot recorded the sensory perceptions from *haptic* (i.e., joint efforts) and *auditory* sensory modalities. During the *grasp* action, the robot recorded *proprioceptive* (i.e., joint angular positions) sensory information from its fingers. The joint efforts and joint positions were recorded for all 6 joints at 15 Hz. The auditory sensory modality was represented as the Discrete Fourier Transform computed using 65 frequency bins.

In addition to the 7 interactive behaviors, the robot also performed the *look* action which produced three different kinds of sensory modalities: 1) an RGB color histogram of the object using 4 bins per channel; 2) Fast point feature histogram (*fpfh*) shape features (Rusu et al., 2009) as implemented in the Point Cloud Library (Aldoma et al., 2012); and 3) deep visual features from the 16-layer VGG

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 4.1: The number of features extracted from each *context*, or combination of robot behavior and perceptual modality.

network (Simonyan and Zisserman, 2014). The first two types of features were computed using the segmented point cloud of the object while the deep features were computed using the 2D image of the object.

Thus, each of the robot’s 8 actions produced two to three different kinds of sensory signals. Each viable combination of an action and a sensory modality is a unique sensorimotor context. In our experiment, the set of contexts \mathcal{C} was of size $2 \times 3 + 6 \times 2 = 18$. The robot performed its full sequence of exploratory actions on each object 5 different times (for the *look* behavior, the object was rotated to a new angle each time). Given a context $c \in \mathcal{C}$ and an object $i \in \mathcal{O}$, let the set \mathcal{X}_i^c contain all five feature vectors observed with object i in context c .

For each language predicate p , a classifier G_p was learned to decide whether objects possessed the attribute denoted by p . This classifier was informed by context sub-classifiers that determined whether p held for subsets of an object’s features.

The feature space of objects was partitioned by context. Each context classifier $M_c, c \in \mathcal{C}$ was a quadratic-kernel SVM trained with positive and negative labels for context feature vectors derived from the *I Spy* game. We defined $M_c(\mathcal{X}_i^c) \in [-1, 1]$ as the average classifier output over all observations for object $i \in \mathcal{O}$ (individual SVM decisions on observations were in $\{-1, 1\}$).

Following previous work in multi-modal exploration, for each context we calculated Cohen’s Kappa $\kappa_c \in [0, 1]$ to measure the agreement across observations between the decisions of the M_c classifier and the ground truth labels from the I

Spy game (Sinapov et al., 2014b). We use κ instead of accuracy because it handles skewed-class data better than accuracy, which could be deceptively high, for example, for a classifier that always returns false for a predicate with few positive examples (such as *ball*, among our objects). We round negative κ up to 0. Given these context classifiers and associated κ confidences, we calculate an overall decision, $G_p(i)$, for $i \in \mathcal{O}$ for each behavior b and modality m as:

$$G_p(i) = \sum_{c \in \mathcal{C}} \kappa_c M_c(\mathcal{X}_i^c) \in [-1, 1]. \quad (4.1)$$

The sign of $G_p(i)$ gives a decision on whether p applies to i with confidence $|G_p(i)|$.

For example, a classifier built for ‘fat’ $\in P$ could give $G_{\text{fat}}(\text{wide-yellow-cylinder}) = 0.137$, a positive classification, with $\kappa_{gr,au} = 0.515$ for the *grasp* behavior’s *auditory* modality, the most confident context. This context could be useful for this predicate because the sound of the fingers’ motors stop sooner for wider objects.

Language predicates and their positive/negative object labels were gathered through human-robot dialog during the *I Spy* game. The human participant and robot were seated at opposite ends of a small table. A set of 4 objects were placed on the table for both to see (Figure 4.1). We denote the set of objects on the table during a given game \mathcal{O}_T .

Human Turn. On the participant’s turn, the robot asked him or her to pick an object and describe it in one phrase. We used a standard stopword list to strip out non-content words from the participant’s description. The remaining words were treated as a set of language predicates, \mathcal{H}_p . The robot assigned scores S to each object $i \in \mathcal{O}_T$ on the table:

$$S(i) = \sum_{p \in \mathcal{H}_p} G_p(i). \quad (4.2)$$

The robot guessed objects in descending order by score (ties broken randomly) by pointing at them and asking whether it was correct. When the correct object was found, it was added as a positive training example for all predicates $p \in \mathcal{H}_p$ for use

in future training.

Robot Turn. On the robot’s turn, an object was chosen at random from those on the table. To describe the object, the robot scored the set of known predicates learned from previous play. Following Gricean principles (Grice, 1975), the robot attempted to describe the object with predicates that applied but did not ambiguously refer to other objects. We used a predicate score R that rewarded describing the chosen object i^* and penalized describing the other objects on the table:

$$R(p) = |O_T|G_p(i^*) - \sum_{j \in O_T \setminus \{i^*\}} G_p(j). \quad (4.3)$$

The robot choose up to three highest scoring predicates \hat{P} to describe object i^* , using fewer if $S < 0$ for those remaining. Once ready to guess, the participant touched objects until the robot confirmed that they had guessed the right one (i^*).

The robot then pointed to i^* and started a follow-up dialog in order to gather both positive and negative labels for i^* . In addition to predicates \hat{P} used to describe the object, the robot selected up to $5 - |\hat{P}|$ additional predicates \bar{P} . \bar{P} were selected randomly with $p \in P \setminus \hat{P}$ having a chance of inclusion proportional to $1 - |G_p(i^*)|$, such that classifiers with low confidence in whether or not p applied to i^* were more likely to be selected. The robot then asked the participant whether they would describe the object i^* using each $p \in \hat{P} \cup \bar{P}$. Responses to these questions provided additional positive/negative labels on object i^* for these predicates.

Evaluation

In our *I Spy* task, the human and robot take turns describing objects from among 4 on a tabletop using attributes (Figure 4.1). A video demonstrating the *I Spy* task and robot learning can be viewed at:

https://youtu.be/jLHzRXPCi_w. As an example, we suggested participants describe an object as *black rectangle* as opposed to *whiteboard eraser*. Additionally, participants were told they could handle the objects physically before offering a description, but were not explicitly asked to use non-visual predicates.

Once participants offered a description, the robot guessed candidate objects in order of computed confidence until one was confirmed as correct by the participant.

In the second half of each round, the robot picked an object and then described it with up to three predicates. The participant was again able to pick up and physically handle objects before guessing. The robot confirmed or denied each participant guess until the correct object was chosen.

I Spy gameplay admits two metrics. The **robot guess** metric is the number of turns the robot took to guess which object the participant was describing. The **human guess** metric is the complement. Using these metrics, we compare the performance of two *I Spy* playing systems (**multi-modal** and **vision-only**). We also compare the agreement between both systems' predicate classifiers and human labels acquired during the game.

During the course of the game, the robot used its RGBD camera to detect the locations of the objects and subsequently detect whenever a human reached out and touched an object in response to the robot's turn. The robot could also reach out and point to an object when guessing.

To determine whether multi-modal perception helps a robot learn grounded language, we had two different systems play *I Spy* with 42 human participants. The baseline **vision only** system used only the *look* behavior when grounding language predicates. Our **multi-modal** system used the full suite of behaviors and associated haptic, proprioceptive, and auditory modalities shown in Table 4.1 when grounding language predicates.

Setup

Data Folds. We divided our 32-object dataset into 4 folds. For each fold, at least 10 human participants played *I Spy* with both the **vision only** and **multi-modal** systems. Four games were played by each participant. The **vision only** system and **multi-modal** system were each used in 2 games, and these games' temporal order was randomized. Each system played with all 8 objects per fold, but the split into 2 groups of 4 and the order of objects on the table were randomized.

For fold 0, the systems were undifferentiated and so only one set of 2 games

was played by each participant. For subsequent folds, the systems were incrementally trained using labels from previous folds only, such that the systems were always being tested against novel, unseen objects. This contrasts prior work using the *I Spy* game (Parde et al., 2015), where the same objects were used during training and testing.

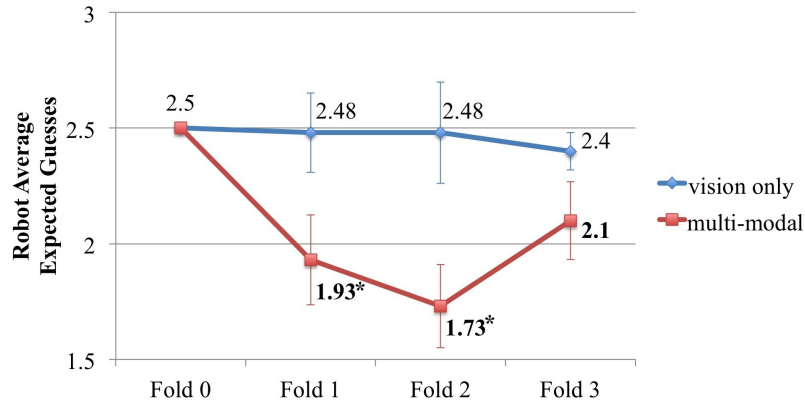
Human Participants. Our 42 participants were undergraduate and graduate students as well as some staff at our university. At the beginning of each trial, participants were shown an instructional video of one of the authors playing a single game of *I Spy* with the robot, then given a sheet of instructions about the game and how to communicate with the robot. In every game, participants took one turn and the robot took one turn. To avoid noise from automatic speech recognition, a study coordinator remained in the room and transcribed the participant’s speech to the robot from a remote computer. This was done discreetly and not revealed to the participant until debriefing, when the games were over.

Quantitative Results

To determine whether our **multi-modal** approach outperformed a traditional **vision only** approach, we measured the average number of robot guesses and human guesses in games played with each fold of objects. The systems were identical in fold 0 since both were untrained. In the end, we trained the systems on all available data to calculate predicate classifier agreement with human labels.

Robot guess. Figure 4.3 (Top) shows the average number of robot guesses for the games in each fold. Because we had access to the scores the robot assigned each object, we calculated the *expected* number of robot guesses for each turn. For example, if all 4 objects were tied for first, the expected number of robot guesses for that turn was 2.5, regardless of whether it got (un)lucky and picked the correct object (last)first. Two-point-five is the expected number for 4 tied objects because the probability of picking in any order is equal, so the expected turn to get the correct object is $\frac{1+2+3+4}{4} = \frac{10}{4} = 2.5$.

After training on just one fold, our **multi-modal** approach performs statistically significantly better than the expected number of turns for guessing (the



Metric	System	
	vision only	multi-modal
precision	.250	.378+
recall	.179	.348*
F_1	.196	.354*

Figure 4.3: **Top:** Average expected number of guesses the robot made on each human turn with standard error bars shown. *Bold:* significantly lower than the average at fold 0 with $p < 0.05$ (unpaired Student’s t -test). *: significantly lower than the competing system on this fold on participant-by-participant basis with $p < 0.05$ (paired Student’s t -test). **Bottom:** Average performance of predicate classifiers used by the **vision only** and **multi-modal** systems in leave-one-object-out cross validation. *: significantly greater than competing system with $p < 0.05$. +: $p < 0.1$ (Student’s unpaired t -test).

strategy for the untrained fold 0 system) for the remainder of the games. The **vision only** system, by contrast, is never able to differentiate itself significantly from random guessing, even as more training data becomes available. We suspect the number of objects is too small for the **vision only** system to develop decent models of many predicates, whereas **multi-modal** exploration allows that system to extract more information per object.

Human guess. Neither the **vision only** nor **multi-modal** system’s performance improves on this metric with statistical significance as more training data is seen. Human guesses hovered around 2.5 throughout all levels of training and sets of objects. This result highlights the difficulty of the robots turn in an I Spy framework, which requires not just good coverage of grounded words (as when figuring out what object the human is describing), but also high accuracy when using classifiers on new objects. Context classifiers with few examples could achieve confidence $\kappa = 1$, making the predicates they represented more likely to be chosen to describe objects. It is possible that the system would have performed better on this metric if the predicate scoring function R additionally favored predicates with many examples over those with few.

Predicate Agreement. Using leave-one-object-out cross validation to evaluate predicate classifiers, we calculated the average precision, recall, and F_1 scores of each against human predicate labels on the held-out object. Table 4.3 (Bottom) gives these metrics for the 74 predicates used by the systems. There were 53 predicates shared between the two systems. The results are similar for a paired t -test across these shared predicates with slightly reduced significance.

Across the objects our robot explored, our **multi-modal** system achieves consistently better agreement with human assignments of predicates to objects than does the **vision only** system.

Qualitative Results

Correlations to physical properties. To validate whether the systems learned non-visual properties of objects, for every predicate we calculated the Pearson’s correlation r between its decision on each object and that object’s measured weight,

height, and width. As before, the decisions were made on held-out objects in leave-one-out cross validation. We found predicates for which $r > 0.5$ with $p < 0.05$ when the system had at least 10 objects with labels for the predicate on which to train.

The **vision only** system led to no predicates correlated against these physical object features.

The **multi-modal** system learned to ground predicates which correlate well to objects' height and weight. The *tall* predicate correlates with objects that are higher ($r = .521$), *small* ($r = -.665$) correlates with objects that are lighter, and *water* ($r = .814$) correlates with objects that are heavier. The latter is likely from objects described as *water bottle*, which, in our dataset, are mostly filled either half-way or totally and thus heavier. There is also a spurious correlation between *blue* and weight ($r = .549$). This highlights the value of multi-modal grounding, since words like *half-full* cannot be evaluated with vision alone when dealing with closed containers that have unobservable contents.

Conclusion

In this chapter, we surpass past work on grounding natural language in robot sensory perception by going beyond vision and exploring haptic, auditory, and proprioceptive robot senses. We compare a vision only grounding system to one that uses these additional senses by employing an embodied robot playing *I Spy* with many human users. To our knowledge, ours is the first robotic system to perform natural language grounding using multi-modal sensory perception through natural interaction with human users.

We demonstrate quantitatively, through the number of turns the robot needs to guess objects described by humans, as well as through agreement with humans on language predicate labels for objects, that our multi-modal framework learns more effective lexical groundings than one using vision alone. We also explore the learned groundings qualitatively, showing words for which non-visual information helps most as well as when non-visual properties of objects correlate with learned

meanings (e.g. “small” correlates negatively with object weight).

This work, while introducing multi-modal grounding for a robotic system, relies on human-robot game to gather perceptual labels. In a deployed robotic system, such labels should be gathered, as much as possible, as needed and on-the-fly during goal-oriented conversations. In the following chapter, we introduce a conversation paradigm for gathering perceptual labels opportunistically during an object retrieval task.

Chapter 5

Opportunistic Active Learning for Grounding Natural Language Descriptions

Grounding perceptual predicates like *red* and *heavy* to objects in the real world is necessary for robots to carry out object identification and retrieval tasks. Object identification may occur in many deployed settings, such as homes (*grab my brown coffee cup*) or factory floors (*go lift the heavy pallet*). To build concept models for these perceptual predicates, we can use training data in the form of objects labeled with whether those concepts apply. In this chapter, we discuss an object identification task in which a human requests a target object and a robot interlocutor is allowed to ask questions about *local* objects, using an active learning strategy, to refine its concept models, as detailed in (Thomason et al., 2017)

Active learning identifies data points from a pool of unlabeled examples whose labels, if made available, are most likely to improve the predictions of a supervised model. Most research on active learning assumes that an agent has access to the entire pool of unlabeled data and can ask for labels of any data points during an initial training phase. However, when incorporated in a larger task, an agent may only be able to query some subset of the unlabeled pool. An agent can also *opportunistically* query for labels that may be useful in the future, even if they are not immediately relevant. Here, we demonstrate that this type of opportunistic active learning can improve performance in grounding natural language descriptions of everyday objects. We find, with a real robot in an object identification setting, that inquisitive behavior—asking users important questions about the meanings of words that may be off-topic for the current dialog—leads to identifying the correct object more often in future interactions.

This work provides a natural context for gathering data for the perceptual grounding required to understand *light* and *mug* from our earlier example (1.1). We later use these concept questions about objects as a sub-dialog in a larger command

understanding pipeline (Chapter 8).

Motivation

In machine learning tasks where obtaining labeled examples is costly (such as those where human interaction is required), active learning allows a system to select its own training data to obtain better performance using fewer labeled examples (Settles, 2010). This allows an agent to iteratively query for labels of examples from an unlabeled pool, and select examples it believes most useful for improving its model.

An important skill required by robots in a home or office setting is retrieving objects based on natural language descriptions. We consider an object identification task where humans can describe real-world objects using both visual and non-visual words (e.g. *red* and *heavy*). In this task, the pool of pre-labeled examples can be extremely limited, since curating a set of all words that apply to every object in an environment is a huge annotation effort for a human user. This motivates our use of active learning to query for additional labels.

Most research on active learning assumes the agent can query a human expert (or an automated oracle) during training for the label of any unlabeled example. Active learning can also be employed while an agent is engaged in such a task and only has access to a limited set of examples in the current environment. In a robot learning setting, physical limitations constrain the set of examples available at any given time.

In particular, a robot in operation will typically be restricted to querying about objects that are physically nearby. In addition, it may be engaged in a task with the human to whom the query is addressed, and for whom the query is unrelated. In such situations, the robot needs to be inquisitive—asking questions that may not be immediately relevant to the task at hand, *and* opportunistic—asking locally *convenient* questions that may not be optimal among all objects since only a subset of objects is available.

We call this setting *opportunistic active learning*, and it differs from existing

work on active learning in three key ways. First, the agent may not be able to ask queries that are globally most useful to improve its models, since the task setting limits the available objects. Second, the agent must decide whether or not to ask such queries while performing another task. Finally, the agent typically depends on some queries being useful for future interactions, but not necessarily the task at hand. Thus, queries have a higher cost than in traditional active learning setups where the goal of the system is simply to learn a good model.

We examine the usefulness of opportunistic active learning to improve an agent’s understanding of natural language descriptions of everyday objects. We consider a task where a robot must identify which member of a set of objects a human user is referring to using natural language. The robot learns classifiers based on multi-sensory information for language predicates that are used to ground natural language descriptions (as in Chapter 4). When trying to understand an object description from one user, the agent is allowed to query for predicate/object labels not directly related to the current interaction.

We compare two agents controlling the robot: one *task-oriented* agent that only asks questions relevant to the current dialog; and one *inquisitive* agent that is able to ask questions unrelated to the current dialog for expected future performance gains. We show that the inquisitive agent both quantitatively outperforms the task-oriented one at predicting the correct object described by human participants, and is rated qualitatively more fun and usable by participants. To our knowledge, ours is the first work to evaluate the effects of asking off-topic questions to human users interacting with a physical robot performing object identification to improve downstream task performance. The source code for this experiment is available at: https://github.com/thomason-jesse/perception_classifiers/tree/active_learning.

Most research in active learning is concerned with the design of appropriate metrics to evaluate possible queries’ likelihood of improving the current model. Examples include uncertainty sampling (Lewis and Gale, 1994), density-weighted methods (Settles and Craven, 2008), and the presence of conflicting evidence (Sharma and Bilgic, 2017). A survey can be found in (Settles, 2010). These

typically assume that the learner can query any example from the pool of unlabeled examples at any time. In contrast, in our work, the system is restricted so that it can only query for data about a subset of examples (objects, in our experiments) at any time.

Past work compares how human teachers perceive different types of queries a robot may pose during learning from a demonstration task (Cakmak and Thomaz, 2012). In that work, the robot required a human operator to aid in the robot’s perception, whereas the system presented in this chapter operates autonomously.

Turn-taking interactions where humans have to teach the robot concepts using positive and negative labeled examples are typical for interactive language grounding, but do not employ active learning (Dindo and Zambuto, 2010; Kollar et al., 2013; Parde et al., 2015; Lutkebohle et al., 2009). Other research uses human-robot interaction, employing forms of active learning to better ground predicates. In those works, the effectiveness of the active learning strategies is not explicitly tested (Vogel et al., 2010; Thomason et al., 2016; Skočaj et al., 2016; Vogel et al., 2010), ontological knowledge (pre-coding *red* as a color) is used during grounding (Mohan et al., 2012), or the predicates to be grounded are not drawn from an open-vocabulary of unrestricted user speech (Cakmak et al., 2010; Kulick et al., 2013).

We fill this gap, testing a strategy that asks human users *inquisitive* questions that are off-topic to the task at hand, studying their effects on downstream task performance and human users’ perceptions. Additionally, our work explores learning open-vocabulary predicates using noisy, high-dimensional visual, audio, and haptic information. We situate our evaluation within the real-world task of object identification, making the multi-modal, perceptual grounding component a prerequisite, but not ultimate goal for good performance.

Methods

We test the effectiveness of using active learning to obtain labels for predicates not relevant to the current dialog for long-term task success. We created an

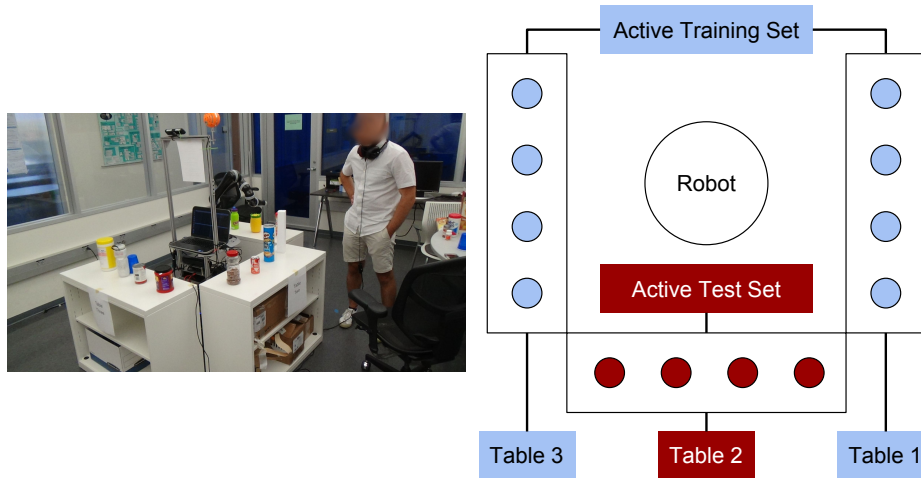


Figure 5.1: Participants describe an object on Table 2 from the *active test set* to the robot in natural language, then answer the robot’s questions about the objects in its *active training set* on the side Tables 1 and 3 before the robot guesses the described target object.

object identification task using a real robot to instantiate the problem. Figure 5.1 shows the physical setup of our task. The human participant and robot both start facing Table 2. This table holds objects in the active test set O_{te} . The tables flanking the robot (Tables 1 and 3) contain objects in the active training set O_{tr} .

Human participants engage in a dialog with the robot. The robot asks the human to describe one of the four objects in its active test set with a noun phrase. Participants are primed to describe objects with properties, rather than categories, given the motivating example *a fuzzy black rectangle* for *an eraser*. Participants are told that the robot had both looked at and interacted with the objects physically using its arm. A demonstration video of the robot system and dialog agents is available here: https://youtu.be/f-CnIF92_wo.

Natural Language Grounding

To connect the noun phrases offered by participants to sensory perception, the robot strips stopwords from the phrase and considers all remaining words as

perceptual predicates. We do not restrict the words that participants are allowed to use to describe objects, so our system learns from an open vocabulary of possible predicates. However, it is not equipped to handle multi-word predicates or those that require understanding phrases (e.g. *water bottle* or *not red*). The robot then creates classifiers to identify these predicates, using objects as positive and negative examples, and getting predicate labels for objects by asking questions about objects in its active training set. Predicates are treated as independent and a separate classifier is learned for each predicate.

Users offer words like *blue*, *cylinder*, and *heavy* when describing objects. We use a corpus of both visual and non-visual feature representations of objects and their features gathered by multiple interaction behaviors by previous work on an object ordering task (Sinapov et al., 2016). These are the same objects and features previously used in the *I Spy* game in Chapter 4. In that chapter, we established that using non-visual modalities when performing language grounding can help with non-visual words like *heavy*. We replicate that methodology here: training and ensembling SVM classifiers for each predicate to predict whether that predicate applies to a novel object (Thomason et al., 2016). For every predicate $p \in P$ for P the set of predicates known to the agent and object $o \in O_{tr} \cup O_{te}$, a decision $d(p, o) \in \{-1, 1\}$ and a confidence in that decision are calculated using Cohen’s kappa $\kappa(p, o)$ estimated from cross-validation performance on available examples. For predicates with too few examples to train SVMs (at least 2 positive and 2 negative examples are needed to fit the SVMs and obtain confidences), we set $d(p, o) = -1$ and $\kappa(p, o) = 0$ for all objects.

Active Learning Dialog Policy

After the participant describes a chosen target object in natural language, the robot asks m questions about objects in its active training set before guessing a target object. Figure 5.2 gives an overview of this dialog.

With probability q_{yn} (we set $q_{yn} = 0.2$ in all experiments), the robot points to an object and asks the yes/no question *Would you use the word p to describe this object?* for some predicate p . To select the predicate p and object $o \in O_{tr}$ to ask

about, we first find the objects in O_{tr} with the lowest confidence κ per predicate (ties broken randomly).

$$o_{\min}(p) = \operatorname{argmin}_{o \in O_{tr}} (\kappa(p, o)).$$

With uncertainty sampling as the query strategy (Settles, 2010), $o_{\min}(p)$ is the next object whose label is queried in a pool-based active learning setting for p 's perceptual classifier. However, because the system is attempting to learn multiple perceptual classifiers, it must also choose which of them should be updated. We weight predicates inversely proportional to their confidence in their least confident labels. That is, the predicate p , and its corresponding least-confidence object $o_{\min}(p)$, are chosen to query with probability:

$$\operatorname{prob}(p) = \frac{1 - \kappa(p, o_{\min}(p))}{\sum_{q \in P \setminus \{p\}} 1 - \kappa(q, o_{\min}(q))}. \quad (5.1)$$

When querying a predicate and object in the above manner, the robot physically turns to the table holding that object, points to it, and asks whether p applies. After getting this new positive or negative example, the robot updates p 's perceptual classifier.

With probability $1 - q_{yn}$, the robot instead selects a predicate p and asks *Could you show me an object that you would describe as p ?* The participant indicates a positive example object for p in the active training set or replies *none*, letting the robot know that all objects in the active training set are negative examples of p . A predicate p is selected uniformly at random from those with insufficient data to fit a classifier:

$$p \in \{q : q \in P \wedge \kappa(q, o_{\min}(q)) = 0\}, \quad (5.2)$$

with the additional constraint that predicates previously asked about in the current dialog with the participant are blacklisted from re-selection. To select an object in the active training set, the participant has to direct the robot to physically face the table containing that object. After getting this new positive label (or, in the case

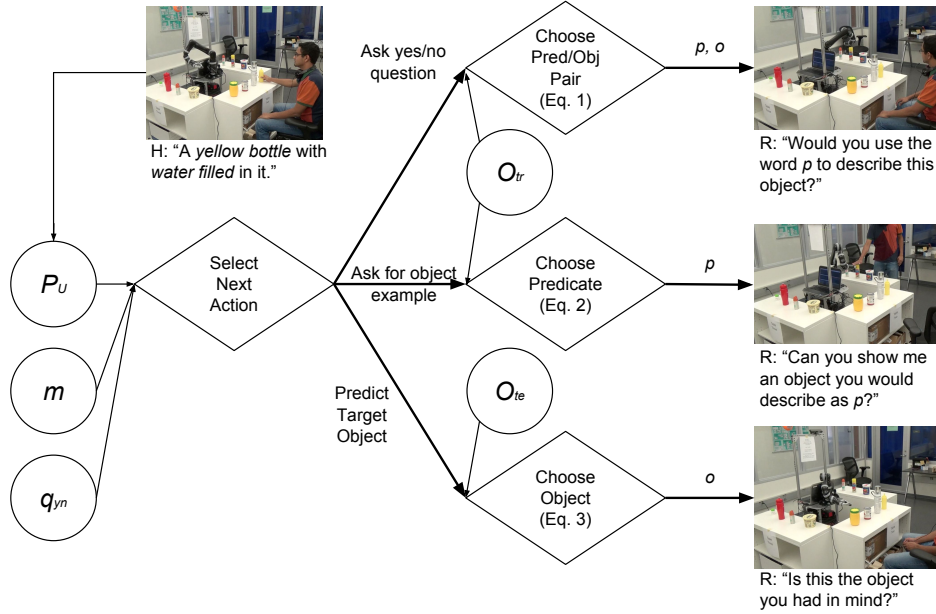


Figure 5.2: After extracting predicates P_U from a human description of a target object, the dialog agent asks up to m questions of two types, choosing one question type over the other with probability q_{yn} . When asking questions, new object labels for the chosen predicate p are restricted to objects in the active training set, $o \in O_{tr}$. The agent then guesses the target object in the active test set $o \in O_{te}$ given its updated grounding classifiers for predicates P_U .

of no object exhibiting p , $|O_{tr}|$ negative labels), the robot updates p 's perceptual classifier.

After asking questions, the robot evaluates the predicates $P_U \subseteq P$ of the participant's utterance against the objects in the active test set O_{te} , then turns to the table of test objects and points to the one that best fit its understanding of the description, $o^* \in O_{te}$,

$$o^* = \operatorname{argmax}_{o \in O_{te}} \left(\sum_{p \in P_U} d(p, o) \kappa(p, o) \right). \quad (5.3)$$

If the robot guesses incorrectly, the human points out the correct object. The target object is considered a positive example for predicates P_U used to describe the

object when the agent is fully retrained between rounds (i.e. when O_{te} is rotated out to become O_{tr} , and the robot is allowed to *see* these objects for building perceptual predicate classifiers).

Robot Implementation

Experiments are conducted on the BWIBots service robot platform (Khandelwal et al., 2017). The robot uses a Kinova Mico arm mounted on top of a custom-built mobile base using a Stanley Robotics Segway RMP which rotates to face the three tables holding the objects. The robot uses an Asus Xtion Pro RGBD camera mounted at the top of its frame to detect the locations of the objects after turning to face a table, and to detect when a human touches an object. The robot can also reach out and point to an object when asking whether a predicate applies. We implement robot behaviors in the Robot Operating System, perform text-to-speech using the Festival Speech Synthesis System (Taylor et al., 1998), and perform automatic-speech-recognition using Google Speech API, recording user speech through a Turtle Beach Ear Force P11 Amplified Stereo Gaming Headset. Once the dialog begins, the robot operates autonomously, asking for operator intervention only when the Xtion camera fails to detect the four expected objects on a tabletop surface (often due to being slightly non-orthogonal to a table after a few in-place rotations, or due to changing lighting conditions as the sun rose and fell).

Evaluation

We randomly divide the set of 32 objects explored in (Sinapov et al., 2016) into 4 folds of 8 objects each, shown in Figure 5.3. We index the folds into $\{0, 1, 2, 3\}$.

We compare two dialog agents controlling the robot. The *baseline* agent is only allowed to ask questions about the predicates relevant to the current dialog. That is, if a person describes the target object as *a blue cylinder*, then the *baseline* agent can only ask about *blue* and *cylinder*. We set $m = 3$ for the *baseline* agent. By contrast, the *inquisitive* agent is allowed to ask questions about any predicate



Figure 5.3: The objects used in our experiments, from fold 0 on the far left to fold 3 on the far right.

it had previously heard, from any user. Thus, the *inquisitive* agent can ask about *heavy* even if the current human says *a blue cylinder* to describe the target object. We set $m = 5$ for the *inquisitive* agent, making it both more talkative and less task-oriented than the *baseline* agent. It would be interesting to learn the optimal values of the parameters for task performance via reinforcement learning, which we leave for future work. In the final round of testing described below, the *inquisitive* agent is restricted to on-topic questions and has $m = 3$, making the agents differ only by their training strategies up to that point.

In the object identification game, the objects in the active training set are from fold n when the objects in the active test set are from fold $n + 1$. Both agents begin with no predicate models. Each human participant is assigned to one of the two agents before the session begins. Participants play two games each. Both games have the 8 objects in the active training set randomly ordered on the robot’s side tables. Between games, the 4 objects on the robot’s front table are alternated such that all 8 of the objects in the active test may be described by the participant. After the games, each participant fills out an exit survey. The agents are tested across three **rounds**, with objects from the active test set moving to the active training set between rounds. The rounds can be summarized by:

- **Round 1** with fold 0 as the active training set and fold 1 the active test set, the agents effectively differ only by the number of questions they can ask, since neither have seen any predicates before;

- **Round 2** with fold 1 as the active training set and fold 2 the active test set, the *inquisitive* agent can ask about the predicates in the current dialog or any predicate it learned from round 1;
- **Round 3** with fold 2 as the active training set and fold 3 the active test set, the agents both operate by the *baseline* rules (on-topic questions, $m = 3$), comparing the effects of the training strategies used in rounds 1 and 2.

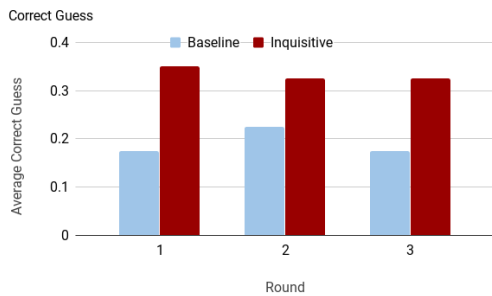
Between rounds, the dialogs that the agents have with their participants so far are aggregated and new predicate classifiers are trained for use in the next round. The agents are trained independently, with the *baseline* agent only using conversations it had, and the *inquisitive* agent only using conversations it had. This training aggregation is done in round-based batches so that the objects in the *active test set* are always unseen by the agents' trained classifiers at the time of any given conversation.

We hypothesize that:

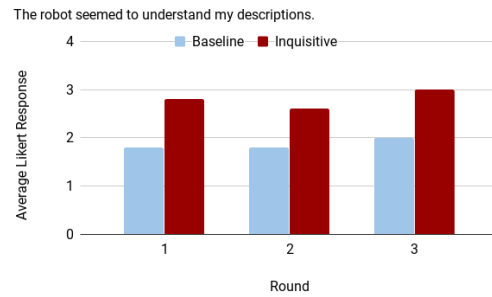
1. The *inquisitive* agent will guess the correct object more often than the *baseline* agent.
2. Users will not qualitatively dislike the *inquisitive* agent for asking too many questions and being off-topic compared to the *baseline* agent.

Experimental Results

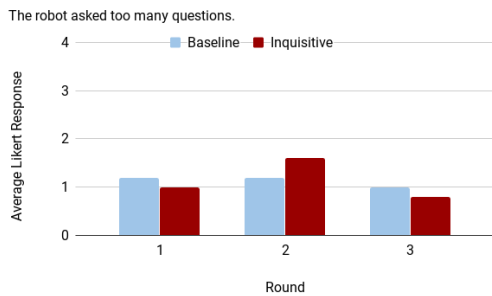
Five participants play two games each with the robot for each agent in each round. In total, 30 study participants, comprising graduate and undergraduate students and employees at our university, interact with the robot. After two games, participants fill out an exit survey by answering *Strongly Disagree*, *Disagree*, *Neutral*, *Agree*, or *Strongly Agree* (mapped to scores 0 – 4) to the questions shown in Figure 5.4.



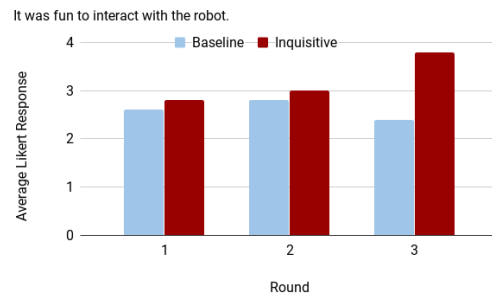
(a) Correct guesses



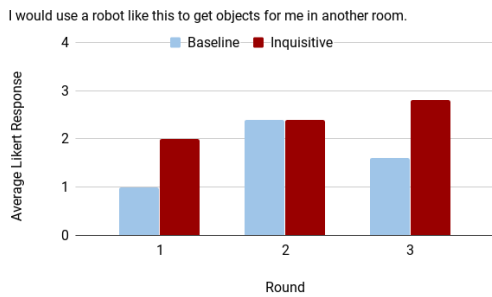
(b) *The robot seemed to understand my descriptions.*



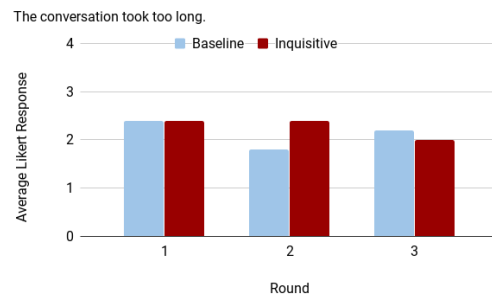
(c) *The robot asked too many questions.*



(d) *It was fun to interact with the robot.*



(e) *I would use a robot like this to get objects for me in another room.*



(f) *The conversation took too long.*

Figure 5.4: Comparing average robot correct guess and average user survey responses across the three rounds between the two agents. In round 1, the agents differ only in the number of questions they can ask. In round 2, the *inquisitive* agent can both ask more questions and ask about off-topic predicates. In round 3, the agents differ only in their training so far, and both have a maximum number of questions fixed to $m = 3$.

Results

Figure 5.4 shows the robot’s average correctness across rounds between the two agents, as well as the average results of exit survey questions. The *inquisitive* agent consistently outperforms the *baseline* agent at identifying the correct object (Figure 5.4a). This difference is statistically significant according to an unpaired, two-sample *t*-test between correctness of the *baseline* versus *inquisitive* agent considering all 15 dialogs each had with participants ($p = 0.035$). We note that in the event of T tied confidences for an object to select, with the correct object among those tied, we reward the robot $\frac{1}{|T|}$ correctness, regardless of the random choice it made among those T .

In round 3, when the agents were both restricted to $m = 3$ questions and only on-topic predicates, the difference in performance is entirely attributable to the training strategies of the agents so far, and the *inquisitive* agent again has a higher rate of predicting the target object. The *inquisitive* agent outperforms a random chance baseline (0.25 average correctness for 4 objects), while the *baseline* agent performs slightly worse due to noisy perceptual classifiers with few positive and negative examples. However, due to the low number of users per agent per round (five), the difference considering round 3 alone is not statistically significant ($p = 0.27$). The *inquisitive* agent is perceived as more understanding on average than the *baseline* agent (Figure 5.4b). These results support our hypothesis that the *inquisitive* agent will outperform the *baseline* agent at the object identification task.

The *inquisitive* agent is perceived as asking too many questions slightly more often than the *baseline* agent in round 2, when it can ask about predicates not related to the current dialog, but not in round 1, where it still asks 2 more questions than the *baseline* agent on average, but they are on-topic (Figure 5.4c). The trends are nearly identical for the similar question of whether users felt the conversation went on too long (Figure 5.4f). The *inquisitive* agent scores higher with human users across rounds than the *baseline* agent for the prompts about whether the robot is fun (Figure 5.4d) and whether the user would use a household robot like this one to get objects in another room (Figure 5.4e). These results support our hypotheses

First seen	Predicates Used by Round		
	round 1	round 2	round 3
round 1	71	32	32
round 2		37	14
round 3			24
Total	71	69	70

Table 5.1: The number of unique predicates introduced in each round and repeated in subsequent rounds. The diagonal shows predicates used for the first time in each round, while the bottom row shows the total unique predicates used (regardless of when they were first seen) per round.

that the *inquisitive* agent will not be disliked for asking too many questions or being off-topic.

Discussion

The *inquisitive* agent’s differences from the *baseline* agent in round 3 partially rely on predicates from previous rounds being used again in that round. In general, asking about an off-topic predicate only helps if that predicate will be seen again in the future. Table 5.1 shows the predicates introduced in each round as well as those repeated from a previous round. There is substantial overlap, indicating that the dataset of objects used is homogeneous enough that learning predicates from previous folds is helpful when identifying objects in unseen folds. Additionally, there were 132 unique predicates introduced throughout all thirty participants’ games, suggesting that the dataset is diverse enough to elicit a wide range of language predicates. Descriptions varied in length among users, as well, from 1 predicate in utterances like *heavy* to 8 predicates in the utterance “a *transparent plastic bottle* with *brown peanuts* inside it with the *red cap*.”

Conclusion

In this chapter, we introduce opportunistic active learning, where a system engaged in a task makes use of active learning metrics to query for labels potentially

useful for future tasks. We demonstrate that a robot using opportunistic active learning during an object identification task performs well in understanding unrestricted natural language descriptions of a target object. Our robot experiments simulate a household robot that can be used to retrieve distant objects and is allowed to first ask questions about nearby objects to help clarify its understanding of natural language predicates.

The robot can ask humans not just questions about words relevant for the current task (e.g. questions about *blue* and *cylinder* when told to *go get me the blue cylinder*) but about any words it currently understands poorly. We demonstrate that such an *inquisitive* agent not only outperforms an agent that stays on-topic with its questions at identifying the correct object described by a human user, but that users find the *inquisitive* agent, on average, more comprehending, fun, and usable in a real-world setting.

By performing semantic parsing rather than simple stopword removal for language understanding, we go on, in Chapter 8, to jointly understand language predicates and human commands. There, we additionally leverage opportunistic active learning to choose questions that best clarify perceptual groundings during task-oriented conversations.

The work in this chapter, as well as that on multi-modal perception with labels acquired in a language game setting (Chapter 4), reveals two important, further problems. First, performing the exploratory behaviors required to learn multi-modal concept models is costly in terms of time and operator effort for a robot. Leveraging the presence of a human operator, we should be able to ask for more than positive and negative associations between objects and language predicates. For example, rather than asking a user for an object she would describe as *red*, the robot might ask what behavior is appropriate for understanding *red*. In this way, the model for *red* could focus on perception related to the camera, sparing the robot the need for performing other exploratory behaviors besides looking at the object. We explore this possibility in detail in Chapter 7.

Second, the assumption made here that a word has a one-to-one relationship with its underlying meaning, or concept model, does not always hold. For example,

the word *light* can mean either light in weight or light in color. Humans in these experiments use *light* both ways, rendering its concept labels noisy and its perceptual classifier unhelpful for both senses. Further, a one-to-one assumption cannot take advantage of synonymy, such as that between *small* and *little*, whose object labels ought to be shared in a single predicate classifier for the underlying concept of being small in size. In the following chapter, we introduce an unsupervised method for discovering multiple word senses and synonymy between word senses when words are represented by multiple context modalities, such as visually and linguistically.

Chapter 6

Multi-Modal Word Synset Induction

A word in natural language can be polysemous—having multiple meanings—as well as synonymous—meaning the same thing as other words. Word sense induction discovers the senses of polysemous words. Synonymy detection detects when two words are interchangeable. For a robot understanding human language commands, performing both sense induction and synonymy detection is necessary to correctly understand commands and collate information for synonymous words.

In this chapter, we combine these tasks, first inducing word senses and then detecting similar senses to form word-sense synonym sets (*synsets*) in an unsupervised fashion, as presented in (Thomason and Mooney, 2017). Given pairs of images and text with noun phrase labels, we perform synset induction to produce collections of underlying concepts described by one or more noun phrases. We find that considering multi-modal features from both visual and textual context yields better induced synsets than using either context alone. Human evaluations suggest that our unsupervised, multi-modally induced synsets are comparable in quality to annotation-assisted ImageNet synsets, achieving about 84% of ImageNet synsets’ approval.

This chapter details a method that could be used to distinguish, for example, the polysemous senses of *light* from our earlier example (1.1). We leave unsupervised polysemy and synonymy detection for predicates represented in multiple, physical modalities for future work (Chapter 9).

Motivation

Semantic understanding in language is complicated by polysemous words that have multiple, distinct meanings, and by synonymous sets of words that have the same underlying meaning. The word *bank*, for example, has at least two distinct meanings: a financial institution and the edge of a river. Manually constructed lex-

ical resources such as WordNet (Fellbaum, 1998) organize noun phrase meanings into senses which can be taken on by one or more noun phrases. Sets of synonymous senses are called *synsets*. For example, one WordNet synset contains both *bank* and *depository financial institution*, two noun phrases that refer to the same underlying meaning.

The ImageNet (Deng et al., 2009) corpus provides images that can be used as visual context for a subset of WordNet synsets. ImageNet required extensive annotation to construct, is limited to its current coverage, and is only available in English. In this work, we introduce *multi-modal word synset induction*, which automatically creates an ImageNet-like resource from a raw collection of images and associated texts annotated with noun phrases. The only initial annotation required is an association between noun phrases and observations, and our method produces synsets without further supervision.

Word sense induction (WSI) automatically determines the senses of a word (Pedersen and Bruce, 1997). Text-based WSI is well-studied and discovers senses by clustering a word’s textual contexts. The multiple senses for *bank* can be recognized as two clusters: one near words like *money* and *deposit*; and another near words like *river* and *shore*. Word similarity tasks attempt to discover words with related meanings. We can discover synsets by performing this kind of similarity search over word senses. To our knowledge, this work is the first to chain polysemy detection via WSI and synonymy detection through sense similarity to induce synsets.

Other notions of context, such as images a word is used to describe, can also be used to discover word senses. For instance, the two readings of *bank* are both textually and visually distinct. When detecting polysemy via WSI and synonymy through similarity, we consider both textual and visual contexts for noun phrases. In the multi-modal perception of a robotics domain, even more contexts might be available, such as haptics and audio, as in Chapters 4 and 5.

For this task, we construct and release a corpus of images paired with web text, each labeled with a noun phrase, from ImageNet synsets, and induce synsets automatically from these. This corpus is large enough, in contrast to impoverished,

existing multi-modal object datasets like those used in Chapters 4 and 5, to test our unsupervised methods. We use the WSI metrics from the SemEval-2010 Word Sense Induction and Disambiguation task (Manandhar et al., 2010), which evaluate systems performing WSI, to measure the quality of the induced synsets against the gold standard from ImageNet. Additionally, we gather human judgments about the quality of induced synsets and ImageNet synsets.

A multi-modal approach using visual and textual features outperforms uni-modal approaches to synset induction in both automated and human evaluations. Human judgments rate our synsets from multi-modal induction as sensible about 84% as often as ImageNet’s, suggesting that our unsupervised synsets are comparable in understandability to human-constructed ones.

In distributional semantics, learning a single vector for an ambiguous word results in a representation that averages that word’s ambiguous senses. First identifying senses and then producing separate vectors for each sense has been shown to improve the performance of models of distributional semantics (Reisinger and Mooney, 2010). Word sense induction is typically approached from distributional, textual context (Pedersen and Bruce, 1997; Schutze, 1998; Bordag, 2006; Manandhar et al., 2010; Di Marco and Navigli, 2013). Building on these works, we go beyond sense induction to additionally group similar senses into synsets, and we use both visual and textual observations of noun phrases to do so.

Other work has used visual information to disambiguate word senses, but assumes the senses of each word are known in advance (Barnard and Johnson, 2005). Using both textual and visual information to perform WSI has been done on datasets where every input word is known in advance to be polysemous (Loeff et al., 2006; Saenko and Darrell, 2008). By contrast, our data contains polysemous, synonymous, and monosemous noun phrases. Additionally, we perform an explicit synonymy detection step to create synsets out of induced word senses, unlike other multi-modal word sense work (Lucchi and Weston, 2012). Our synonymy detection step is related to lexical substitution (McCarthy and Navigli, 2007), but at the word sense level.

Similar works use co-clustering in separate textual and visual spaces, treat-

ing textual clusters as word senses and visual clusters as lower-level iconographic senses, such as different viewpoints for or orientations of an object (Chen et al., 2015). We use deep image features from the VGG network (Simonyan and Zisserman, 2014) trained for object recognition, which ameliorates the need for iconographic distinctions. Some work uses images and text to discriminate between word senses, but takes multiple senses as known, rather than inducing them automatically (Kanishcheva and Angelova, 2016).

The VGG network is trained to take an image as input and identify the synset it belongs to in ImageNet. We hold out the synsets used to train VGG as validation data in our work. Past work has used the VGG network to extract visual features from objects (as in Chapter 4) for developing similarity metrics within ImageNet (Deselaers and Ferrari, 2011), and for lexical entailment detection (Kielar et al., 2015).

Methods

We created a large corpus of images paired with text from web pages containing those images by scraping ImageNet synsets. This gave us a gold standard set of synsets, together with multiple instances of visual and linguistic contexts for each concept represented by a synset. We were then able to compare our unsupervised reconstructions of synsets to these gold-standard ImageNet synsets.

Dataset

We selected a subset of ImageNet synsets that were leaves in the WordNet hierarchy (e.g. *kiwi* but not *bird*) and were not used to train the VGG network. Table 6.1 gives the number of noun phrases which participated in polysemous and synonymous relationships among these 6,710 synsets, S .

We took the synsets used to train the VGG network as a development data set, V . We performed reverse image search using an open-source API (<https://github.com/vivithemage/mrisa>) to get text for each image in V from web pages on which that image appeared. Images for which too little text

Noun phrase relationships			
synonymous	polysemous	both	neither
4019	804	1017	2586

Table 6.1: Number of noun phrases that are synonymous, polysemous, both, or neither in the ImageNet synsets S used in our experiments.

could be extracted were not included in the dataset. We performed latent semantic analysis (*LSA*) (Deerwester et al., 1990) on term frequency-inverse document frequency (*tf-idf*) vectors of bag-of-words representations of this text to create a 256-dimensional text feature space. In early experiments, we tried Word2Vec (Mikolov et al., 2013) embeddings trained over the development text associated with synsets of V , but achieved better performance from simple LSA, possibly due to the small size of the development corpus.

For each synset $s \in S$, deep visual features and LSA embedding features were extracted for up to 100 images per noun phrase associated with s in ImageNet. We arbitrarily selected the first $100 * |s|$ valid image URLs listed for the synset by the ImageNet API, eliminating ones for which too little text data could be found via reverse image search. Visual features were the activations of the 4,096-dimensional, penultimate layer of the VGG network given the image as input. This yielded a set of image observations I_s .

For each image, we gathered web text (about 400 words per image) as above and embedded it in our LSA space to get textual observations T_s . We expect text observations to be sense-specific for the images they are paired with, since, for example, a web page with a picture of a bank building is unlikely to discuss the rainfall and flash floods associated with rivers. So, for each s , up to 100 multi-modal observations $O_s = \langle I_s, T_s \rangle$ are available per noun phrase. This corpus of ImageNet synsets associated with text, VGG features, and LSA embedding features per synset observation URL is publically available at <https://github.com/thomason-jesse/synpol>.

For each noun phrase $np \in NP$, we associate observations with np from

each synset in which it participated by dividing each O_s uniformly among participating noun phrases (illustrated in Figure 6.1). We refer to noun phrase observations as O_{np} . We note that these even divisions may not reflect a realistic distribution of senses (i.e. the fruit sense of ‘kiwi’ dominates the bird and people senses), but different hyperparameters could be set for specific domain distributions in a deployed application.

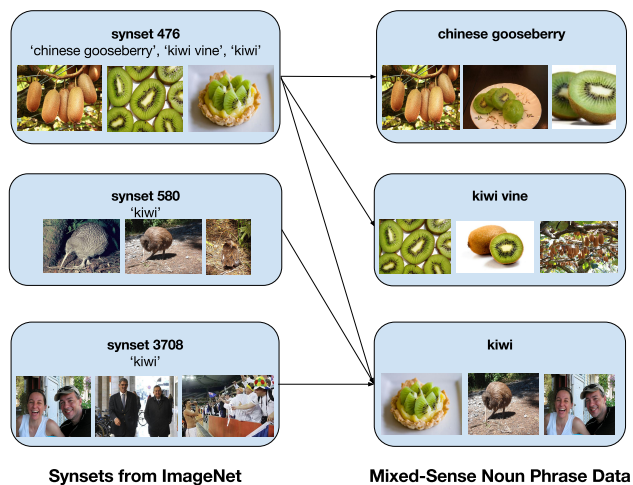


Figure 6.1: Noun phrase observations O_{np} (right) are extracted from ImageNet synsets (left). Our task is to automatically induce synsets from the image and text observations of these noun phrases in an unsupervised fashion.

Synset Induction

Given this corpus of noun phrase image and text observations, we perform polysemy-detecting WSI to induce senses (Figure 6.2) followed by synonymy detection to form synsets (Figure 6.3).

We performed synset induction using only visual features, only textual features, and both. Our induction algorithms are based on clustering. Using an early fusion paradigm (Bruni et al., 2014) and cosine distance to combine modalities, we

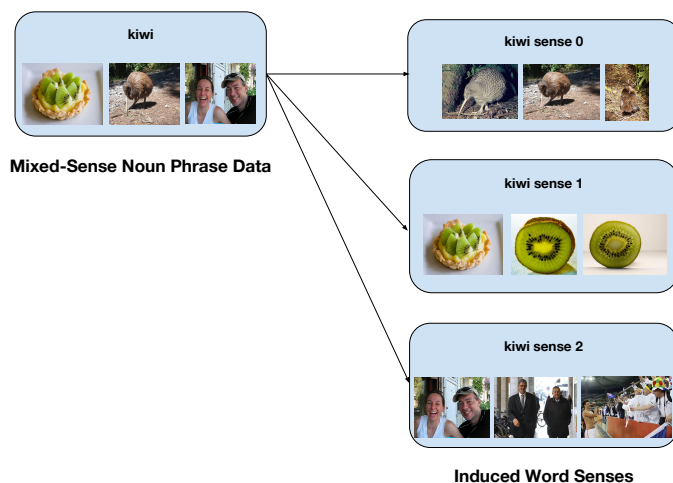


Figure 6.2: We induce senses for each noun phrase by clustering among its observation feature vectors in textual and visual space.

calculate distance $d(o_1, o_2)$ between observations as follows:

$$d(o_1, o_2) = (\alpha) \text{cosd}(I_{o_1}, I_{o_2}) + (1 - \alpha) \text{cosd}(T_{o_1}, T_{o_2}),$$

$$\text{cosd}(a, b) = 1 - \frac{a \cdot b}{\|a\| \|b\|},$$

where α controls the relative influence of visual (I) and textual (T) features. We perform vision-only, text-only, and multi-modal induction by setting α to 1, 0, and 0.5, respectively.

Polysemy detection

Polysemy detection is implemented through k -means clustering, where k is estimated for each set of observations O_{np} using the gap statistic (Tibshirani et al., 2001). Intuitively, the gap statistic selects the smallest number of clusters k that reduces within-dispersion compared to $k - 1$ by more than chance. Additionally,

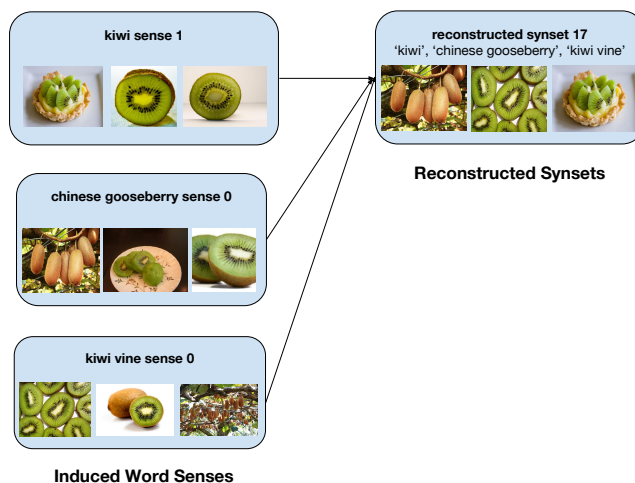


Figure 6.3: We induce synsets by calculating the mean sense observation vectors across induced senses, then clustering over those means.

we enforce a constraint that no induced sense has fewer than 20 observations (estimated as the mean senses per noun phrase minus one standard deviation in the development data). Consequently, the observations O_{np} for each noun phrase np are clustered into k senses, yielding sense observation sets O_{np,k_i} for $k_i \in 0 \dots k$. Together, these observation sets form a set of induced senses G .

Synonymy detection

Using the gap statistic to estimate a number of clusters k^* for synonymy detection is inappropriate because we know k^* is on the order of $|G|$. The gap statistic is best applied when looking for a minimum sensible k^* , and further sensible divisions of k^* well-separated clusters may exist within these larger clusters (Tibshirani et al., 2001). For synonymy detection, the number of synsets is closer to the total number of word senses than to 1.

Instead, we use a greedy merging approach. We compute a mean observation vector for each induced sense $O_{np,k_i} \in G$, as well as the pairwise distance $d(m_1, m_2)$ between all mean sense vectors. Greedy merges of the nearest means

produce a final set of K induced synsets, R , each of which comprises no more than L distinct word senses.

Membership in each induced synset $r \in R$ is the union of observations of the senses $g_a \dots g_b \in G$ whose observations were merged (i.e. $r = \cup\{g_a \dots g_b\}$). K is set based on the ratio of senses to synsets in the development data V (so K fluctuates depending on the number of senses discovered during polysemy induction to be clustered). The maximum number of senses per synset, $L = 32$, is also estimated from V .

Evaluation

Both automated and human evaluations demonstrate that multi-modal synset induction outperforms uni-modal induction. Human judges do *not* significantly favor ImageNet synsets over multi-modal, induced synsets; however, humans *do* favor ImageNet’s over uni-modally induced synsets.

Automated Evaluation

We computed the *v-measure* (Rosenberg and Hirschberg, 2007) of the induced synsets, calculated as the harmonic mean of their *homogeneity* and *completeness* with respect to the gold-standard ImageNet synsets. High homogeneity means the induced synsets mostly contain observations that correspond to a single gold synset, while high completeness means each gold synset’s observations are mostly assigned to the same induced synset. We do not compare our word sense induction method to past WSI datasets (Manandhar et al., 2010; Navigli and Vannella, 2013), because we take an additional synonymy detection step, and we consider textual and visual information jointly, while existing corpora use only text.

Homogeneity and completeness are defined in terms of the class entropies $H(S)$ and $H(R)$ of the gold-standard ImageNet synsets S , induced synsets R , and their conditional entropies $H(S|R)$ and $H(R|S)$. Specifically, homogeneity

$h(S, R)$ is calculated as follows:

$$\begin{aligned}
H(S) &= - \sum_{i=1}^{|S|} \frac{\sum_{j=1}^{|R|} a_{ij}}{N} \log \frac{\sum_{j=1}^{|R|} a_{ij}}{N}, \\
H(S|R) &= - \sum_{j=1}^{|R|} \sum_{i=1}^{|S|} \frac{a_{ij}}{N} \log \frac{a_{ij}}{\sum_{k=1}^{|S|} a_{kj}}, \\
h(S, R) &= \begin{cases} 1 & H(S) = 0 \\ 1 - \frac{H(S|R)}{H(S)} & H(S) > 0 \end{cases},
\end{aligned}$$

with a_{ij} the number of observations of gold synset S_i included in induced synset R_j , and N the total number of observations in the dataset. Completeness $c(S, R)$ is defined as follows:

$$\begin{aligned}
H(R) &= - \sum_{j=1}^{|R|} \frac{\sum_{i=1}^{|S|} a_{ij}}{N} \log \frac{\sum_{i=1}^{|S|} a_{ij}}{N}, \\
H(R|S) &= - \sum_{i=1}^{|S|} \sum_{j=1}^{|R|} \frac{a_{ij}}{N} \log \frac{a_{ij}}{\sum_{k=1}^{|R|} a_{ik}}, \\
c(S, R) &= \begin{cases} 1 & H(R) = 0 \\ 1 - \frac{H(R|S)}{H(R)} & H(R) > 0 \end{cases},
\end{aligned}$$

with the v -measure defined as the harmonic mean of $h(S, R)$ and $c(S, R)$.

We also computed the *paired f -measure* (Manandhar et al., 2010), the harmonic mean of the paired precision and recall between the ImageNet and induced synsets. Rather than count membership overlap between two sets, paired f -measure compares membership overlap between sets of sets.

Specifically, we count the number of observation pairs (o_i, o_j) that are members of both synset s and induced synset r to get an overlap score between each $s \in S$ and $r \in R$. There are $\binom{|s|}{2}$ observation pairs for each s and $\binom{|r|}{2}$ observation pairs for each r across all such s and r , comprising $C(S)$ gold pairs and $C(R)$ induced pairs, respectively. Then paired f -measure $f(S, R)$ is defined as the

harmonic mean of paired precision $p(S, R)$ and recall $r(S, R)$:

$$p(S, R) = \frac{|C(S) \cap C(R)|}{|C(R)|},$$

$$r(S, R) = \frac{|C(S) \cap C(R)|}{|C(S)|}.$$

Results are presented in Table 6.2. The multi-modal approach achieves the highest v -measure and is tied for highest paired f -measure. This unsupervised task operates over a whole dataset, not a train/test split where cross validation could be performed, so we have simply highlighted the highest score for each metric. These paired scores are low compared to those of strict word sense induction (Manandhar et al., 2010) because our method attempts to induce synsets, not just word senses, adding complexity.

	K	h	c	v	p	r	f	human
ImageNet	6710	1.0	1.0	1.0	1.0	1.0	1.0	0.470
vision	9976	0.897	0.888	0.893	0.326	0.440	0.375	0.388
text	6406	0.853	0.911	0.881	0.173	0.496	0.256	0.346
vision+text	8216	0.887	0.910	0.899	0.286	0.543	0.375	0.395

Table 6.2: Homogeneity (**h**), completeness (**c**), v -measure (**v**), paired precision (**p**), recall (**r**), f -measure (**f**), and **human** scores of our induced synsets with visual features only, textual features only, and both. Bold indicates highest value by modality (excluding the gold-standard ImageNet).

Homogeneity and paired precision are maximized when every observation has its own synset. Completeness and paired recall are maximized when all observations belong to a single synset. The vision-only system overproduces synsets and increases precision, while the text-only system underproduces synsets and increases recall. The multi-modal system is able to balance between these flaws to achieve high v -measure and paired f -measure.

Human Evaluation

We observed that ImageNet synsets do not necessarily match categories humans distinguish. For example, ImageNet distinguishes photos of Croatian from Ukrainian peoples, as well as having a synset for people who could be described as *energizers* (mostly men in suits). By contrast, multi-modal induction grouped senses of noun phrases referring to people together into one synset.

We created an Amazon Mechanical Turk task to evaluate the quality of our induced synsets according to human judgment. Given a noun phrase and a set of synsets that noun phrase participated in, annotators marked whether they thought the sets were *more sensible* or *more confusing*. Figure 6.4 shows the interface with one of the validation examples, discussed below.

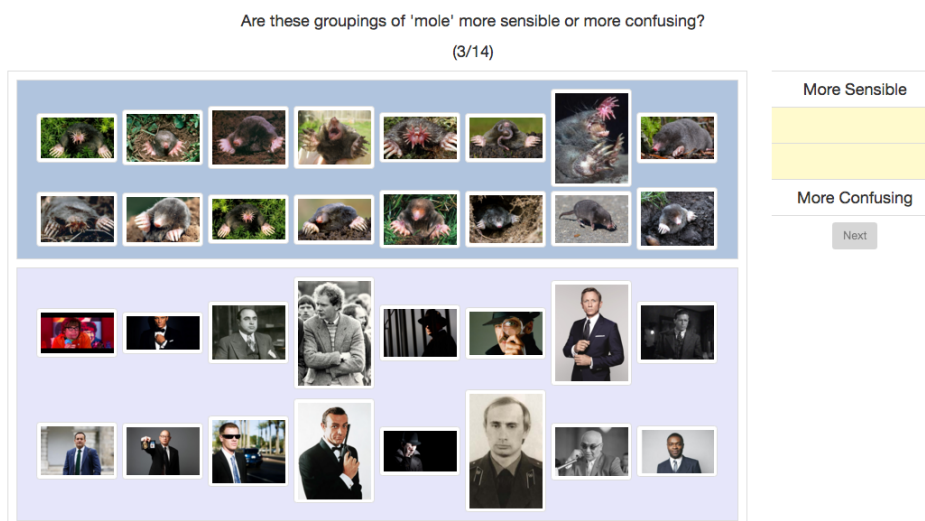


Figure 6.4: The Mechanical Turk interface used to gather annotations. The noun phrase *mole* was a hand-crafted validation example of *more sensible* synsets—one for the burrowing animal and one for the spy.

Annotators were walked through three examples of how the word *bank* might be split into synsets before the task began. Two senses containing bank (financial institution and riverbank) were shown, with one *sensible* example of them well-separated in two synsets, one *confusing* example of the senses lumped together in a single synset, and one *confusing* example where the senses were separated but

there were two distinct synsets for financial institutions even though this is a single concept.

Three noun phrases were selected randomly from the corpus for each annotator. Annotators evaluated vision-only, text-only, and multi-modal induced synsets, as well as the gold standard ImageNet synsets. The ordering of the 12 sets (3 noun phrases \times 4 models) was randomized before being shown to the annotator. Two hand-created validation examples—one of well-separated *mole* senses (animal and spy), and one of incorrectly grouped *crane* senses (birds and construction)—were inserted in random positions, and data from users who answered either of these incorrectly was discarded.

After removing data from users who failed validations (nearly half did—the task is challenging for Mechanical Turkers), and noun phrases assigned to multiple annotators who did not reach consensus (e.g. tie in whether sets of synsets are sensible or confusing), 156 noun phrases remained, annotated across all four models (624 annotator decisions total) by 58 distinct annotators. We calculated the average annotator decision per noun phrase/model combination (a simple, binary *more sensible* = 1, *more confusing* = 0), and averaged those decisions across noun phrases to calculate human scores per model, shown in Table 6.2.

ImageNet synsets are only rated more sensible than confusing about half the time in our sample, highlighting the noisiness of ImageNet synsets. We conducted paired (by noun phrase) Student's *t*-tests between all models and found that only the differences between ImageNet and the uni-modal models are significant ($p < 0.05$). Humans found multi-modal induced synsets sensible about 84% as often as ImageNet synsets ($.470 \cdot .84 \approx .395$), without requiring explicit annotations to build synsets from noun phrases and observations.

Figure 6.5 shows an example where annotators favored our multi-modal, induced synsets versus ImageNet. The patterns of vision-only induction overproducing synsets (e.g. two senses of *washboard*, *splashboard* for the presence and absence of a human) and text-only induction under-producing them (e.g. *washboard* and *dulcimer* instruments combined in one synset) are common. Multi-modal induction's advantage lies in balancing these opposing trends, producing more coher-

ent synsets like the two shown for *washboard*.

For other noun phrases, like *amphitheater*, ImageNet distinguishes it from *coliseum* while unsupervised induction recognizes their similarity, and human annotators agree with collapsing the two. Situations where ImageNet makes a distinction human annotators disagree with is also common among synsets of people, as in the *rapper/rock star* example in Figure 6.6. As an additional example, ImageNet separates nationalities like *Austrian* and *Croatian*, while automatic induction (across modalities) favors putting groups of people together without respect to nationality.

Conclusion

In this chapter, we introduce the task of *multi-modal word synset induction* and an unsupervised method to construct synsets from image and text observations labeled with noun phrases. Additionally, we create a dataset of image and text feature observations, drawn from ImageNet and reverse image search, and processed by the VGG network and LSA, labeled with noun phrases from ImageNet.

We show that a multi-modal, unsupervised clustering approach in which visual and textual features are considered together outperforms uni-modal clustering at the synset induction task both quantitatively and qualitatively. Human annotators rate our multi-modal, induced synsets sensible 84% as often as gold-standard ImageNet’s, suggesting our unsupervised method is competitive with manual annotation for creating synsets from noun phrase-level observations.

These methods can be applied to any vector representation of instances labeled with discrete classes that need to be disambiguated. For example, in a grounded language system where word senses are associated with real-world object properties in visual (Perera and Allen, 2013; Parde et al., 2015) or multi-modal space (Chapters 4 and 5), instances are object representations and labels are adjectives and nouns applied to those objects. Words like *round* are visually polysemous, since something can be flat and circular or spherical and still be called *round*. This work could tease out these meanings of *round* and subsequently join the meaning

of *spherical* to the appropriate sense. Additionally, discovering that *light* is polysemous across modalities (coloration versus weight) and joining the color sense to *bright* and the weight sense to *lightweight* could make robot-human communication clearer, since an embodied agent should prefer the less polysemous descriptor words when describing things to a human. We leave this kind of integration with robot perception for future work (Chapter 9).

A limitation of applying this method to multi-modal object representations is impoverished datasets of object representations. For example, there are only 32 objects considered in the work presented in Chapters 4 and 5. The effort required by both a human operator and a robot to create multi-modal object representations is substantial, and, at least for targeted applications, might be reduced by adding priors on which behaviors are appropriate. In the following chapter, we explore using unsupervised word embedding vectors and human annotations to guide object exploration for the task of learning a new perceptual concept, such as *red* or *heavy*.


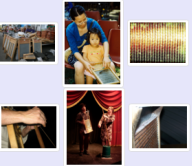
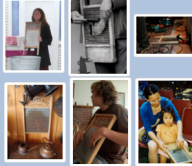

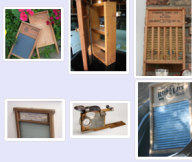


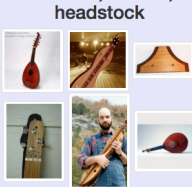

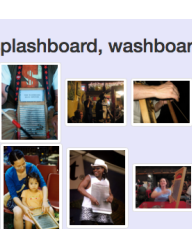
ImageNet	vision
<div data-bbox="326 411 558 625"> <p>splashboard, washboard</p>  </div> <div data-bbox="570 411 802 625"> <p>washboard</p>  </div>	<div data-bbox="846 411 1078 625"> <p>washboard</p>  </div> <div data-bbox="1089 411 1321 625"> <p>splashboard, washboard</p>  </div> <div data-bbox="846 640 1078 846"> <p>washboard, splashboard</p>  </div>
text	vision+text
<div data-bbox="326 909 558 1188"> <p>washboard</p>  </div> <div data-bbox="570 909 802 1188"> <p>king post, dugout, washboard, catapult, knothole</p>  </div> <div data-bbox="326 1203 558 1430"> <p>psaltery, washboard, dulcimer, cithern, headstock</p>  </div>	<div data-bbox="846 909 1078 1188"> <p>washboard</p>  </div> <div data-bbox="1089 909 1321 1188"> <p>splashboard, washboard</p>  </div>

Figure 6.5: Human annotators favored the multi-modal, induced synsets for noun phrase *washboard* over ImageNet’s and other models’ synsets. ImageNet fails to properly distinguish the *washboard* senses of a household object and instrument, vision alone creates too many instrument senses, and text alone overgeneralizes the instrument sense. Multi-modal induction properly separates the household object and instrument senses.

ImageNet	vision
<div data-bbox="326 468 558 688"> <p>rapper, doorknocker, knocker</p>  </div> <div data-bbox="558 468 797 688"> <p>rapper</p>  </div>	<div data-bbox="850 468 1083 688"> <p>rapper</p>  </div> <div data-bbox="1083 468 1321 688"> <p>rapper</p>  </div> <div data-bbox="850 772 1083 930"> <p>rapper, renegade</p>  </div> <div data-bbox="1083 699 1321 930"> <p>rapper, knocker, doorknocker</p>  </div>
text	vision+text
<div data-bbox="326 993 558 1203"> <p>rapper</p>  </div>	<div data-bbox="850 993 1083 1203"> <p>rapper</p>  </div> <div data-bbox="1083 1024 1321 1203"> <p>rapper, rock star</p>  </div> <div data-bbox="850 1213 1083 1413"> <p>rapper, doorknocker, knocker</p>  </div>

Figure 6.6: Human annotators favored ImageNet’s synsets for noun phrase *rapper* over all models’ induced synsets. ImageNet properly separates the door-knocker and performer senses, vision alone creates too many senses for the performer, text alone fails to distinguish the senses at all, and multi-modal mistakenly splits the performer sense into those with and without *rock star* qualities.

Chapter 7

Guiding Exploratory Behaviors for Multi-Modal Grounding of Linguistic Descriptions

A major limitation of existing approaches to multi-modal language grounding is that a robot has to exhaustively explore training objects with a variety of actions when learning a new language predicate like *light* or *tall*. In this chapter, we propose a method for guiding a robot’s behavioral exploration policy when learning a novel predicate to achieve high accuracy with lower exploration times, as detailed in (Thomason et al., 2018). We demonstrate our approach on two datasets in which a robot explored objects and was tasked with learning to recognize whether novel words applied to those objects.

We estimate which behaviors are relevant for a target predicate to be learned, such as *squishy*, from two sources. First, we consider the distance between the embedding vectors (Mikolov et al., 2013) of the predicate to be learned and those that are already known. The relevant behaviors for known neighbors in the embedding space, as estimated by leave-one-out accuracy on known objects, are likely relevant for the new predicate as well. Second, we use human annotations of which behaviors are applicable for a new predicate to be learned, for example that only the *press* behavior is relevant to determine whether something is *squishy*. Human intuitions about which behaviors are relevant can guide search, but we show that using these together with unsupervised word embedding neighbor information gives better performance than using either alone.

This work provides a methodology that can be used to build object representations on-the-fly when learning new concepts. It could be used together with opportunistic active learning to jointly acquire both object-predicate labels and object representations from exploratory behaviors relevant for those predicates. We leave such an integration for future work (Chapter 9).

Motivation

In Chapter 4, we demonstrated that using sensory data beyond vision to ground language predicates improves robotic performance over using vision alone (Thomason et al., 2016). Non-visual exploratory behaviors such as pushing, grasping, and lifting objects can be costly in terms of time (e.g. localizing an object with a camera in order to press down on it) and operator intervention (e.g. pushing an object off a table, requiring an operator to retrieve it).

In many settings, a robot needs to perform object exploration for a specific grounding task. For example, if someone asks a household robot to *get the heavy mug from the kitchen*, the robot may need to explore some novel objects in the kitchen to determine which one satisfies *heavy* and *mug*. If a dataset of unexplored objects labeled as *heavy* (or not) and *mug* (or not) is available, the robot should be able to explore those objects quickly to learn the concepts before traveling to the kitchen to identify the referent. Exhaustively performing all actions when exploring new objects to learn a novel predicate will, in expectation, yield the best accuracy, but that exploration scales poorly as the number of behaviors and objects increases.

In this chapter, we investigate using exploratory behaviors to learn a novel predicate on a time budget without sacrificing grounding accuracy. We use two datasets of predicate-object relationships that include both visual (*red*, *cylinder*) and non-visual (*heavy*, *full*) predicates that require haptic and auditory feedback to understand. We compare methods for deciding which behaviors to perform when exploring objects in order to learn a new predicate, beyond the obvious time-consuming option of performing all behaviors. One possibility is to utilize unsupervised information in the form of word embeddings, such as those produced by Word2Vec (Mikolov et al., 2013). The distance between two words' embedding vectors suggests their semantic similarity. If *squishy* is close to *rigid* in the embedding space, a robot may be able to learn *squishy* using just the exploratory behaviors that determine whether an object is *rigid*. Another possibility, if a robot is operating in a shared space with humans, is to ask a human which behaviors they would perform to evaluate the predicate.

We demonstrate that word embeddings help learn predicates using fewer behaviors. Our approach is independent of the embedding vectors used, and we compare embeddings from two different corpora, noting that as the categorical quality of the embeddings improve (e.g. colors close to colors, weights close to weights), so should the gains achieved by our approach. We also show that using human-provided behavior annotations speeds up learning in a domain of real-world objects with predicates from organic human descriptions in an embodied setting where behaviors must be performed in a certain order (e.g. an object must be *grasped* before it can be *lifted*).

Most past research in language grounding has focused on using the visual sensory modality. However, non-visual modalities can also be used to improve a robot’s ability to ground semantic information (Araki et al., 2012; Chu et al., 2013; Silberer and Lapata, 2014; Kiela and Clark, 2015; Gao et al., 2016b; Alomari et al., 2017b). A major limitation of these approaches is that they require the robot to perform exhaustive object exploration, i.e., the robot must explore each object with some fixed number of exploratory actions (e.g., *grasp*, *lift*, *shake*, *push*, etc.) during which it records non-visual sensory data. For example, in past work on learning multi-modal classifiers for a set of haptic adjectives, a robot performed seven different exploratory behaviors on 51 objects for a total of five times (Chu et al., 2013). While some methods have been proposed for how a robot should sequence its behaviors to minimize exploration time when classifying a novel object, these approaches still require exhaustive exploration during training (Sinapov et al., 2014a; Zhang et al., 2017).

One possible way to address this problem is to estimate the relevance of each behavior for the task of learning a novel predicate or category. Sinapov *et al.* (Sinapov et al., 2014b) show that a robot’s learning performance on a novel predicate (e.g., *squishy*) can be improved if the robot has some prior information about the predicate’s similarity to known words (e.g., *rigid* and *soft*). That paper stops short of exploring where such a prior could originate. In this chapter, we answer this question by using word embeddings to estimate the relevance of known words to novel ones. Additionally, we gather annotations from humans about which

behaviors they perceive as relevant for a given predicate (Figure 7.4). This is related to past human annotations gathered for relevant sensory modalities of words (Lynott and Connell, 2009).

To guide the robot’s exploration when learning a new word, we use distributional semantics to map words into high-dimensional vector spaces where their vector distances carry semantic information. Word2Vec uses a neural skip-gram model to create an embedding space for words given a large corpus (Mikolov et al., 2013). Related strategies consider context embeddings of words as well (Melamud et al., 2015). Past work has created multi-modal Word2Vec-style embeddings that consider textual context together with visual (Silberer and Lapata, 2014; Lazaridou et al., 2015; Kottur et al., 2016) or audio (Vijayakumar et al., 2017) context.

Recent work has used word embeddings to predict unseen verb causality information from seen verbs (Gao et al., 2016a), and affordances of unseen noun from seen nouns (Fulda et al., 2017). These are similar in spirit to our use of unsupervised word embeddings created from large, unannotated text corpora to assist with a supervised grounded language learning problem—predicting the multi-modal representations most helpful for understanding a novel predicate.

The problem we address bears some similarities to the zero-shot learning problem (Xian et al., 2017; Fu et al., 2015; Kodirov et al., 2015). In zero-shot learning, the task is to produce a classifier for a novel class label for which labeled data is unavailable, given some descriptor of that class label. In our case, the task is to produce a behavioral exploration policy when learning a new word given an embedding that relates the novel word to ones that are already learned. To our knowledge, this problem has not been addressed in the zero-shot learning literature.

Methods

In our experiments, we focus on learning a new predicate across a set of unexplored objects given a dataset of learned predicates and explored objects.

Let P be a set of predicates and O be a set of objects. Let the label function $\mathcal{L}(p, o) \in \{-1, 1\}$ indicate whether predicate $p \in P$ holds true for object $o \in O$.

Let B be the set of available exploratory behaviors and let C be a set of sensorimotor contexts, such that each context corresponds to a combination of a behavior (e.g., grasping an object) and a sensory modality (e.g., auditory features extracted from the sound detected during grasping).

The robot’s task is to learn predicate classifier models that can predict whether or not a predicate applies to an object given the multi-modal behavioral observations of that object. We learn these models from object-predicate labels as discussed in Chapter 3, reviewed briefly here. The robot learns an individual grounding classifier $G_{p,c}$ for each sensorimotor context c and predicate p . To determine whether the predicate applies to object $o \in O$, the weighted combination of these context-specific classifier outputs gives a consensus decision $d(p, o) \in \{-1, 1\}$ calculated as:

$$d(p, o) = \text{sgn} \left(\sum_{c \in C} w_{p,c} G_{p,c}(o) \right), \quad (7.1)$$

where $w_{p,c}$ is the estimated reliability weight of context c .

One way of setting the weight $w_{p,c}$ is to make it proportional to the classification performance (e.g., Cohen’s κ) of the classification function $G_{p,c}$ as estimated from training data. Once these reliability weights have been estimated, we hypothesize that the robot can then perform only a subset of its behaviors to achieve high classification performance on novel objects. Below, we formulate the problem of how a robot can estimate surrogate reliability weights for a *novel* predicate for which *no* training data is available to estimate κ .

Problem Formulation

Given a set of known predicates P , their labels on a set of explored objects $O_E \subset O$, an unseen predicate q to be learned, and an unexplored set of objects $O_U \subset O$ (with $O_E \cap O_U = \emptyset$) labeled for predicate q , we explore strategies for learning a classifier for q on an exploration time budget without sacrificing accuracy.

The robot’s task is two-fold: 1) estimate surrogate weights $w_{q,c}$ for the novel predicate q and each context $c \in C$; and 2) determine the order(s) in which to

perform behaviors $b \in B$ given their cost (e.g., time) and the estimated weights associated with their contexts on O_U . Reliability weights for q can then be re-estimated at test time given the features of labeled objects extracted from the guided exploratory behaviors.

Estimating Unseen Predicate Context Reliability Weights

A baseline strategy for estimating surrogate $w_{q,c}$ is to assign a uniform weight per context. We can also use word embedding distances to share context weights from known predicates P to unknown predicate q . For every pair of predicates $p, q \in P$ with word embedding vectors v_p, v_q we calculate the similarity as the positive cosine distance between those vectors:

$$poscos(p, q) = \frac{1}{2}(1 + \cos(v_p, v_q)) \in [0, 1]. \quad (7.2)$$

It is common to use cosine distance in high-dimensional embedding spaces to measure word vector dissimilarities because it is independent of features' magnitudes. We find the top- k most lexically similar predicates to q in an embedding space, $P_q \subseteq P, |P_q| = k$ (allowing more than k in the event of a tie) and take a similarity-weighted average of $w_{p,c}$,

$$w_{q,c} \approx \frac{1}{|P_q|} \sum_{p \in P_q} poscos(p, q) w_{p,c}. \quad (7.3)$$

Expected Values for Behaviors

Given a weight for every context, we calculate weights $w_{q,b}$ at the behavior level. These are obtained by calculating training object decisions at the context level, aggregating them using weights $w_{q,c}$ (for c a context of behavior b), and calculating κ confidences based on those behavior-specific decisions across the training

objects. Then we can calculate the expected value for each behavior as:

$$v(b) = w_{q,b} + \epsilon, \quad (7.4)$$

for some small ϵ such that behaviors with no confidence weight are not zero valued at training time (since they may yet prove useful at testing time).

Evaluation

We performed experiments on two datasets. The first has a small number of predicates and a representative set of objects that readily support effective learning, and thus clearly demonstrates the utility of the proposed approach. The second dataset has a large number of predicates that arose organically during human-robot interaction for a diverse set of household objects, and thus learning the predicates is much more challenging.

We visualize projections of those predicates in word embedding as well as κ confidence space, demonstrating qualitatively and quantitatively that using weight-sharing through word embeddings (Equation 7.3) can help a robot learn a new predicate with fewer exploratory behaviors than an exhaustive exploration approach.

We demonstrate that using word embeddings to share information and gathering behavior annotations for a new predicate allows a robot to learn that predicate on a time budget, with real-world behavior ordering constraints, more effectively than exploring considering the time to perform behaviors alone.

Learning object colors, weights, and contents

We demonstrate the effectiveness of surrogate reliability weight estimation using word embeddings to predict relevant contexts for a novel predicate given known predicates on a dataset of objects that readily support effective predicate learning.

Dataset Description

We use the dataset described by Sinapov et al. (2014b), in which a robot explored 36 different objects using 11 prototypical exploratory behaviors: *look*, *grasp*, *lift*, *shake*, *shake-fast*, *lower*, *drop*, *push*, *poke*, *tap*, and *press*. While performing these, the robot gathered sensory information from: proprioceptive joint-torque sensors for all 7 joints, audio from an Audio-Technica U853AW cardioid microphone, and vision from a Microsoft Kinect sensor. The objects were identical containers except along 3 different attributes: 1) color: *red*, *green*, *blue*; 2) weight: *light*, *medium*, *heavy*; and 3) contents: *beans*, *rice*, *glass*, *screws*. These variations resulted in $3 + 3 + 4 = 10$ total predicates in the set P that the robot was tasked with learning.

During the execution of the *look* behavior, the robot perceived 2 different sensory modalities, one corresponding to a color histogram of the object in the foreground, and the other comprising of a reduced size 10×10 RGB image of the object. For the remaining interactive behaviors, the robot recorded 2 types of sensory features, auditory and haptic, produced by the interaction with the objects. Thus, the robot’s set of sensorimotor contexts was of size $|C| = 11 \times 2 = 22$. The robot performed each of the 11 behaviors on each of the 36 objects 10 different times, resulting in a total of 3960 behavioral interactions.

Sample Predicate Embeddings

Figure 7.1 (a) shows a sample 2D projection of the Google News Word2Vec embeddings corresponding to the 10 predicates in this dataset. Figure 7.1 (b) shows the 2D projection for the lexical substitution-focused embeddings that consider context embeddings (Melamud et al., 2015). The projection was computed using Multi-Dimensional Scaling (MDS) (Kruskal and Wish, 1978).

Figure 7.1 (c) shows an embedding based on each predicate’s reliability weights estimated as agreement κ associated with each sensorimotor context in C .

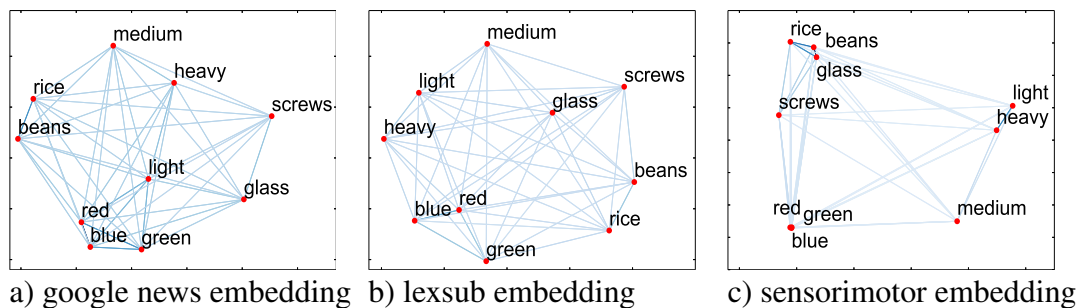


Figure 7.1: a) 2D projection of the Google News Word2Vec embedding of the 10 predicates used in the first experiment; b) 2D projection of the lexical substitution-focused embedding; c) 2D projection of an embedding constructed based on the relevant sensorimotor contexts for each of the 10 predicates. Shared structure can be seen between the word embeddings (a,b) and the sensorimotor embeddings of robot experience (c), which we leverage for learning novel predicates.

For each predicate $p \in P$, a feature vector f_p of size $|C|$ was computed such that the i^{th} entry corresponded to the confidence κ_i for context $c_i \in C$. These vectors were used to compute a $|P| \times |P|$ distance matrix using Euclidean distance. Notably, the visualizations show that there is some shared structure between the lexical embeddings and this sensorimotor embedding. In particular, attributes of similar types (e.g. colors) appear close together in both embedding types. We show that exploiting this shared structure can be used to improve learning novel predicates.

Evaluation and Results

The proposed methodology was evaluated using a *leave one predicate out* approach: during each run, the robot learned multi-modal grounded classifiers for 9 of the 10 total predicates P , using 12 fully explored and labeled objects that were randomly sampled from the entire set of 36 objects. When learning the remaining predicate, the robot was given a budget of N behaviors to use during both training and testing. The robot estimated the context reliability weights for the novel predicate using the lexical substitution-focused word embeddings (Melamud et al.,

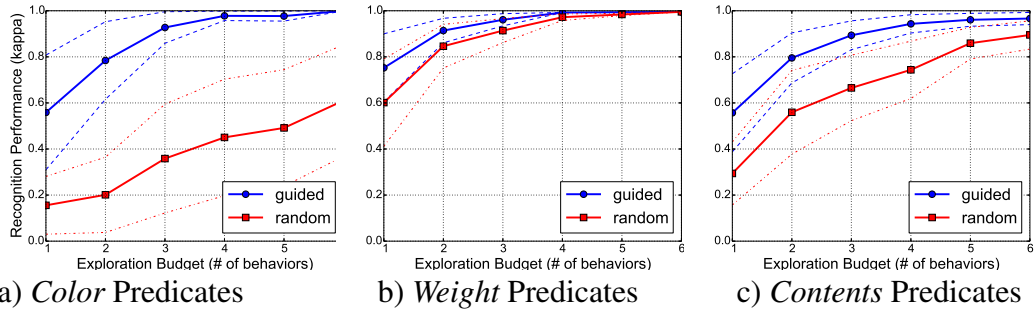


Figure 7.2: Test-time κ performance of classifiers for learning a new predicate based on the reliability weight estimation strategy used at test time as more behaviors are allowed. **random** chooses the next exploratory behavior at random, while **guided** uses word embeddings to select known neighbor predicates from which to estimate reliability weights for behaviors. The dotted lines denote the variance over 75 simulation runs.

2015) via Eq. 7.3, with $k = 7$, and propagated these to the behavior level. These estimates were then used to compute a distribution over behaviors B , which was used to sample a subset of size N (the budget) used for both training and testing. In this experiment, we do not explicitly model behavior transitions, but instead assume that any behavior can be performed at any time and that all behaviors have equal cost. In the next experiment (Subsection 7.3.2), we consider realistic behavior transition constraints and the more difficult set of objects introduced in Chapter 4. The context-specific predicate recognition models were implemented by a Support Vector Machine (SVM) with an RBF kernel.

The results of this test are shown in Figure 7.2. Each of the three plots contains the average κ recognition rates for the three types of predicates: a) colors (red, green, and blue), b) weights (light, medium, and heavy), and c) contents (beans, rice, screws, and glass). The proposed method is compared against the baseline approach of randomly selecting b behaviors using a uniform prior. Given sufficient budget, all methods perform all behaviors and achieve identical accuracy; examining these reduced budgets shows the effectiveness of our approach under exploration time constraints.

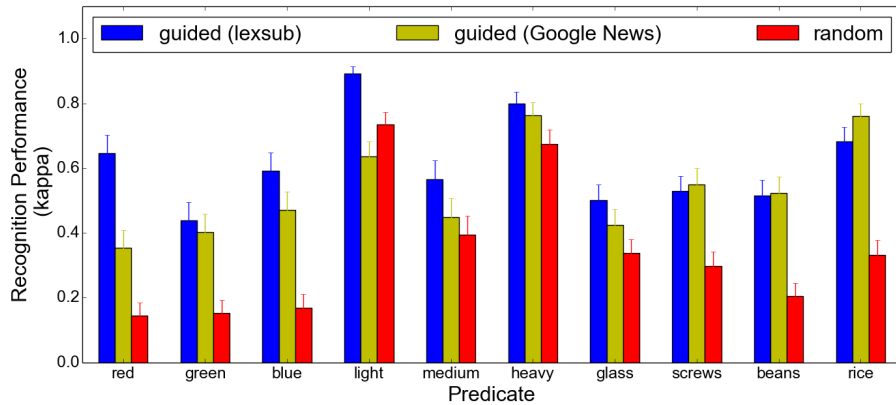


Figure 7.3: Recognition results for all 10 predicates using just 1 exploratory behavior selected according to three different conditions: *guided* with lexical substitution predicate embedding, *guided* with Google News predicate embedding, and *random*. The bars denote standard error.

The proposed method enables the robot to reach good recognition rates ($\kappa > 0.95$) faster than random exploration, with the difference especially noticeable for color- and contents-related predicates. Figure 7.3 shows recognition results using a budget of $N = 1$ behavior for two different embeddings: lexical substitution and Google News. For some of the predicates, the lexical substitution embedding performs substantially better; in particular, the Google News embeddings links the word *light* with the colors and thus, the first behavior chosen when learning it tends to be *look*, which does not provide informative signals regarding the weight of the object (they all have the same size). This is related to the polysemy of *light*, discussed in Chapter 6. The lexical substitution embedding puts *light* closer to the other two weight-related predicates and thus achieves the best performance.

Below, we evaluate the proposed method on a much more challenging dataset in which the robot was tasked with learning words provided by everyday human users, and constrained to perform behaviors in a realistic order while considering the time it takes to perform each behavior.

Behavior	Time (s)
drop	9.8
grasp	22
hold	5.7
lift	11.1
look	0.8
lower	10.6
push	22
press	22

Table 7.1: The time in seconds needed to perform each exploratory behavior.

Learning words from everyday human users

We use feature representations from multiple behaviors and modalities for 32 objects using 8 exploratory behaviors (Figure 4.2), the same as those used for the experiments in Chapters 4 and 5. The time in seconds needed to perform each exploratory behavior is given in Table 7.1. For every object, there are features from every sensorimotor context, as described in earlier chapters. Seven of the exploratory behaviors have two sensory modalities, and the *look* behavior has three, resulting in $|C| = (7 \times 2) + (1 \times 3) = 17$ contexts.

Predicate Annotations

We consider the 81 predicates used by players in the *I Spy* game evaluation performed in Chapter 4. In this work, we gathered full annotations between those 81 predicates and the 32 objects (allowing us to set $\mathcal{L}(p, o)$ for every predicate p and object o). We gathered 3 annotators’ opinions about whether each predicate applied to each object. We took a majority vote between the 3 annotators when there was a disagreement. To reduce annotator fatigue, each annotator labeled predicates for 8 of the 32 total objects, requiring 12 annotators in total to gather all labels. The average pairwise κ between annotators was 0.576. Figure 7.4 shows all the predicates given positive labels for a sample object.



text, bright, cup,
large, round, heavy,
container, red, full,
water, cylindrical,
colored, thing, hol-
low, top, plastic,
white, cap, cylinder,
medium-sized, tall,
liquid, object, bottle

Figure 7.4: Predicates with positive labels for the object in the picture, from annotations gathered in this work.

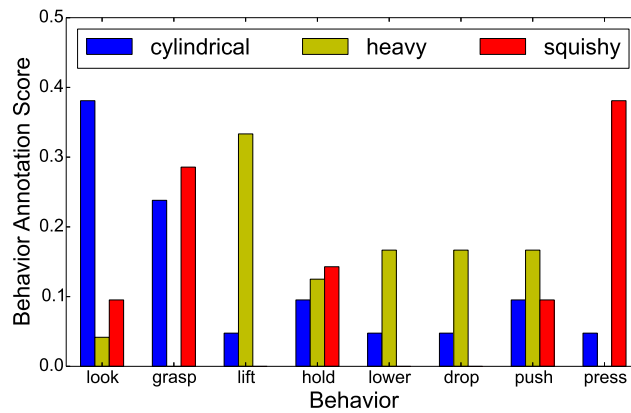


Figure 7.5: Behavior annotations for three predicates in the dataset: *cylindrical*, *heavy* and *squishy*. Scores correspond to the ratio of annotators who rated the behavior as relevant for recognizing whether the predicate applied to an arbitrary object.

Behavior Annotations

For each of the 81 predicates, we gathered annotations in order to create a distribution over behaviors relevant for that predicate. Annotators were asked to mark which exploratory behaviors they would engage in to determine whether a given predicate applied to a novel object. Annotators were shown how the behaviors were performed by a robot, for example that *press* was performed by coming down from above the object until the hand touched it. Annotators could mark multiple behaviors for each predicate, but were required to mark at least one.

We gathered annotations from 14 people, then discarded the annotations from those whose average pairwise κ agreement with all other annotators was less than 0.4. This cutoff left us with 8 annotators whose average agreement was $\kappa = .475$. We assign each behavior a value for each predicate of the ratio of annotators (out of these 8) who marked it relevant, so that for every $p \in P, b \in B$ we have an annotation score $A(p, b) \in [0, 1]$. Figure 7.5 shows the behavior annotation scores for three predicates. We release the predicate-object labels and predicate behavior annotations as a publically available dataset: https://github.com/thomason-jesse/object_exploration.

In addition to estimating $w_{q,c}$ from Eq. 7.3 (e.g. top- k nearest word embedding lexical neighbor predicates), we estimate it from behavior annotations alone (Eq. 7.5) and from an interpolation of behavior annotations and semantic neighbors (Eq. 7.6). For C_b the set of contexts for behavior b and b_c the behavior associated with context c :

$$w_{q,c} \approx \frac{1}{|C_b|} A(q, b_c); \quad (7.5)$$

$$w_{q,c} \approx \frac{1}{|C_b|} A(q, b_c) * \frac{1}{|P_q|} \sum_{p \in P_q} \text{poscos}(p, q) w_{p,c}. \quad (7.6)$$

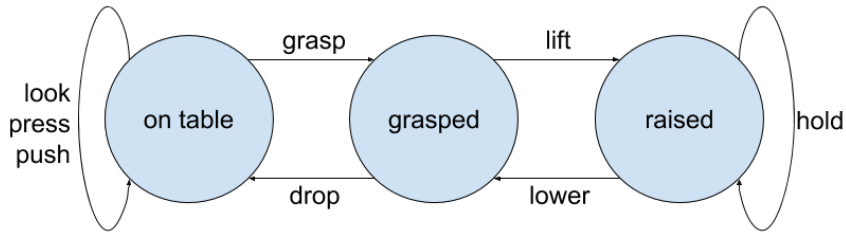


Figure 7.6: Exploratory behavior actions as transitions in an object state graph.

Choosing an Exploration Policy

Given the values of each behavior (Eq. 7.4) for an unknown predicate q , the presuppositions of each behavior, the time to perform each behavior $t(b)$, and a time limit per object for exploration T , we can sample a sequence of behaviors to use when evaluating predicate q . Figure 7.6 describes the state effects of behaviors on the object being explored, while Table 7.1 gives the time in seconds to perform each. Because there are 5 observations per behavior per object available, each behavior can be performed in an exploration policy up to five times, making enumerating all policies intractable.

We take a Monte-Carlo-style approach, sampling a large number of behavior sequences through weighted random walks, then choosing one sequence among all those that maximizes reliability weight while minimizing time. To sample a sequence of behaviors, we start at the *on table* state (Figure 7.6), choosing any available behavior with probability proportional to $v(b)$ (Eq. 7.4) with respect to other available behaviors. For example, from the *on table* state, the probability of choosing *press* is

$$p(\text{press}) = \frac{v(\text{press})}{v(\text{look}) + v(\text{press}) + v(\text{push}) + v(\text{grasp})},$$

assuming *press*, *look*, *push*, and *grasp* have each been performed fewer than 5 times and there is enough remaining time in the budget given the sequence so far to execute each (e.g. $t(\text{press}) \leq T$). When no outgoing behaviors meet these constraints, a sampled sequence ends.

In our experiments, we sample 100 sequences S for every training trial. Of those sampled, we first select the subset \hat{S} of sequences with the highest value, then randomly choose one among those tied for the shortest exploration time,

$$\hat{S} = \text{set-argmax}_{s \in S} \left(\sum_{b \in s} v(b) \right);$$

$$s^* \in \text{argmin}_{s \in \hat{S}} \left(\sum_{b \in S} t(b) \right).$$

The chosen sequence s^* is used to explore the unseen objects O_U , extracting features for training classifiers for novel predicate q .

Experiments and Results

We randomly split the 32 objects into 16 explored objects O_E and 16 unexplored objects O_U . We then perform leave-one-predicate-out cross validation, holding predicate q out. For predicate q , we are given the labels $\mathcal{L}(q, o)$ for $o \in O_U$. We then perform leave-one-object-out cross validation, deciding on a training behavior sequence for q , using it to explore 15 of the unexplored objects, re-estimating context reliability weights as κ agreement, and finally exploring the held-out object and assigning a label for q based on these new reliability weights. In this way, we can obtain agreement statistics with true labels for every held-out predicate, aggregating these to compare different surrogate reliability weight estimations for choosing an exploration policy.

Our leave-one-predicate-out experiment operates over the 48 predicates for which O_U had at least 2 positive and 2 negative object examples for the predicate. We calculate word embedding distance (Eq. 7.2) using Google News Word2Vec embeddings, use linear SVMs as context-level classifiers, set $k = 3$ (Eq. 7.3), and set $\epsilon = 0.001$ (Eq. 7.4). The alternative, lexical substitution-focused embeddings (Mohan et al., 2013) perform similarly. For every time budget T and surrogate reliability estimate compared, the behavior sequence sampling and leave-one-object-out cross validation was repeated 100 times to obtain average performance. Figure 7.7 shows

these average performances. The time budgets sampled are chosen so that each behavior has time to be performed one, two, and three times each, if the policy chooses homogeneously. With a sufficient time budget, all methods are able to perform all behaviors five times (the maximum), achieving convergent performance.

Figure 7.7 shows the average κ agreement achieved by grounding classifiers trained under different surrogate weight estimation strategies. In this more difficult set of objects and predicates, borrowing weights from nearest lexical neighbors in word embedding space (**lex**) is insufficient to improve grounding accuracy on a behavior time budget. Unlike the clear-cut predicates of the previous experiment (Subsection 7.3.1), the predicates arising from human users in this dataset do not form clearly defined semantic clusters like those visible in Figure 7.1.

However, behavior annotations (**ba**) improves performance. The best performance for grounding classifiers is achieved when considering these together with lexical neighbor information (**ba+lex**). We postulate that this occurs because there is a slight mismatch between the behaviors that humans would use to determine properties versus what is actually helpful to a robot. Conversely, human intuitions about which behaviors are relevant help prune out information from erroneous lexical neighbors in this more complicated set of predicates.

These results demonstrate that gathering behavior annotations for an unseen predicate can improve grounding performance on a time budget, and performance is further boosted by using word embeddings to share neighboring predicates' reliability weights.

Figure 7.7.b) shows the performance when learning *bottle*. The top- k neighbors for *bottle* are *canister*, *tub*, and *container*. The resulting **lex** reliability-weight sharing emphasizes the *drop/audio* context—listening, perhaps, for the thumping sound of the tub and canister containers. Figure 7.7.c shows the performance when learning *tall*, and demonstrates behavior annotations keeping performance steady as exploration time increases by focusing on the *look* and *press* behaviors, while other strategies explore irrelevant behaviors.

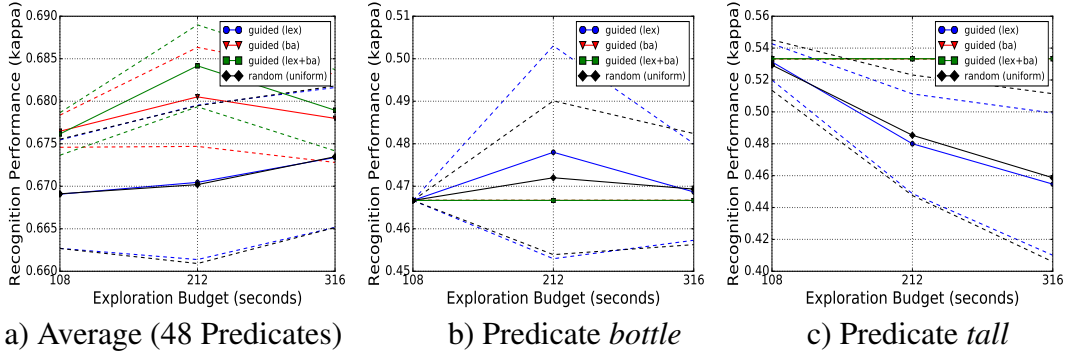


Figure 7.7: Test-time κ performance of classifiers for learning a new predicate based on reliability weight estimation strategy used at test time for three time budgets. **uniform** assigns reliability $\frac{1}{|C|}$ to each context, **lex** (Eq. 7.3) estimates reliability weights from neighbor predicates, **ba** (Eq. 7.5) from behavior annotations alone, and **ba+lex** (Eq. 7.6) from behavior annotations interpolated with **lex**. The dotted lines denote the variance over 100 simulation runs. (a) Across all predicates, **lex** alone does not outperform the **uniform** baseline, but when combined with behavior annotations **ba+lex** achieves the best performance overall. *Bottle* (b) is a predicate for which **lex** helps and behavior annotations do not, and *tall* (c) is a predicate for which the opposite is true.

Conclusion

Current methods for grounding object concepts in behavioral exploration and multi-modal perception suffer when a robot needs to exhaustively perform all of its actions to determine which ones are useful for learning the target concept. In this chapter, we address this problem through a framework for guiding a robot’s behavioral exploration of objects when learning new words. In this framework, given a novel word, the robot computes an exploration policy specific to that word by relating it via word embeddings to words that have already been learned.

We demonstrate that our method allows the robot to learn new words faster, in terms of the number of different behaviors the robot needs to perform on objects to learn the target word (Subsection 7.3.1). We also demonstrate that behavior annotations gathered from human users can be integrated into the framework to

further improve predicate recognition performance under a time budget as well as physical and temporal constraints (Subsection 7.3.2).

In future work, behavior annotations could be gathered from human users on-the-fly in a dialog setting, using a learned human-robot dialog policy to estimate when behavior annotation questions are warranted. We leave such integration for future work (Chapter 9). Using modality annotations (Lynott and Connell, 2009) may also further boost performance.

In this framework, a set of unexplored objects with labels of whether a new predicate to be learned applies is given as input. This kind of information could come from human-robot conversation, when a human uses a new word that the robot needs to then learn in order to complete a task. As discussed in Chapter 3, natural language understanding modules for things like commands can require large amounts of annotator effort to train, but can be built up incrementally by leveraging information in human-robot dialogs. Further, applying priors from known predicates, as we do in this chapter, requires a set of learned predicates and corresponding predicate-object labels. In Chapters 4 and 5, we discussed strategies for getting these labels during a language game or object identification task, respectively. In the following chapter, we explore combining these strategies to incrementally improve both natural language understanding and perception modules on a robot through human-robot conversation.

Chapter 8

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

In the preceding chapters, we have discussed work related to improving language understanding on a robotic platform. Two main thrusts of this thesis are parsing natural language to underlying meanings, as in Chapters 3 and 6, and on bringing robotic sensors to bear on the real world to build multi-modal models of perceptual concepts, as in Chapters 4, 5, and 7.

In this chapter, we present a holistic system for jointly improving parsing and perception on a robotic system for natural language commands through human-robot dialog. This learning agent uses clarification questions in a conversation with a human interlocutor to understand language commands. The agent induces additional training data for a semantic parser, similar to the ideas presented in Chapter 3, strengthening its parsing over time. The agent also engages in short clarification sub-dialogs related to word synonymy when identifying possible perceptual words, addressing some of the issues raised in Chapter 6. Finally, the agent uses opportunistic active learning, presented in Chapter 5, to ask questions about nearby objects to refine perceptual classifiers on-the-fly during command dialogs.

We evaluate this language understanding agent on Mechanical Turk with hundreds of users. Figure 8.1 shows the Mechanical Turk interface we created, with an example command and the beginning of a human-agent dialog. We find that, as more training conversations are seen by the agent, users are better able to communicate tasks to the agent. Qualitatively, users rate the agent more favorably for use in deployed tasks as more training conversations become available.

Because a web interface does not allow users to physically interact with objects, those users tend not to use non-visual predicates like *heavy*. However, in a physical setting, users do use these non-visual predicates, as established in Chapter 4. In light of this, we also demonstrate the agent on a physical robot with

ROBOT What should I do?

YOU move a red can from the middle lab to the mallory's office

ROBOT You want me to move an item from somewhere to somewhere (**not** give it to someone)?

YOU yes

ROBOT Where should I move something from on its way somewhere else?

YOU the middle lab

ROBOT To where should I move something from **here**?

YOU

Say

The diagram shows a floor plan with the following layout:

- Top row: Office (H), Office (N), Pod, Conference, Office, Office (D), Office (W), Pod, Office (S), Office.
- Middle section: Lounge, Pod, a greyed-out area, Lab, Lab (with a red X), Lab.
- Bottom row of middle section: Printer, Kitchen.
- Bottom row: a greyed-out area, Office (M), Pod, Office (P), Office (B), Office (R), Lounge, Conference, Pod.
- Far right: Lab.

Figure 8.1: Mechanical Turk web interface used to conduct our experiments. The user types a command to the learning agent, which replies with questions to clarify the command until the user confirms that the agent has correctly understood. In this conversation, the agent has just asked a confirmation question—whether it has understood the right location from where the *red can* should be taken. The colored text *here* refers to the pink-outlined map below the conversation window.

an arm (Khandelwal et al., 2017), using the learning agent as a backend to drive human-robot dialog. Initializing the learning agent based on conversations from Mechanical Turk, we then give it the command “*Move a rattling container from the conference room by the lab to Bob’s office*” and demonstrate its ability to recognize *rattling* as a new word to be learned, learn a concept model on-the-fly for this word, and successfully select a *rattling container* to move from the specified source room to the specified target room. This demonstration can be viewed at: <https://youtu.be/Uz0bC-4byMc>.

Motivation

Robots deployed in human environments need to be able to understand commands from humans. As discussed in previous chapters, substantial annotated resources are needed to perform such command understanding. Gathering corpora of, for example, language commands and their underlying meanings, together with language labels for objects in the environment a robot may interact with, is expensive. Moreover, a corpus of language commands for an office robot, together with explored objects and their language labels (*coffee cups, heavy mugs, etc.*) are unlikely to be useful for a robot operating on a factory floor, in a hospital, or in a military operation.

Thus, throughout this thesis, we have focused on developing dialog agents for robots that use conversation with humans to clarify understanding given sparse initial language understanding resources. In this way, the agents should be able to acquire the language and perception capabilities they need in the environment in which they are deployed, over time, from talking with humans in that environment. In this chapter we present a system that improves both its parsing and perception components over time. The algorithms are based on the dialog strategies for eliciting training pairs for a semantic parser (Chapter 3), and on opportunistic active learning strategies for choosing questions about how words relate to objects (Chapter 5) when building multi-modal predicate classifiers to understand perceptual concept words (Chapter 4). We also touch on the issues raised in Chapter 6, by asking

human users about whether an unseen, perceptual concept word is a synonym of an already known word, possibly merging those words to the same underlying concept (for example, *tall* and *long*).

To our knowledge, this is the first integrated, embodied robotic system that can jointly acquire a more general semantic parser and learn new ontological concepts that connect to the physical world, using visual, haptic, and auditory signals from object behaviors as a model against which to ground perceptual concepts.

In addition to the related work discussed in Chapter 2, a few recent works are particularly relevant for the system presented here. Command understanding is related to question answering, in that some questions require exploration actions, as explored in a recent research paradigm: embodied question answering (Das et al., 2017a). In that work, however, the language inputs are constrained to templates and restricted to always be *unambiguous*. In our work, we explicitly handle ambiguity by asking the user clarification questions, and using their responses as a signal for improving parsing performance in future conversations.

In existing work, attribute learning is sometimes done through exploring objects and aligning novel objects with novel attributes (Tucker et al., 2017). Here, we instead leverage the presence of the human interlocutor to continue dialog and directly ask clarification questions about new concepts to build perceptual models of them on-the-fly.

Recent work leveraged word embeddings to recover from rare or previously unseen words during a language grounding task (Bastianelli et al., 2016), a strategy we employ together with augmenting our parsing training data from conversations, such that successful lexical substitutions guided from word embeddings are learned as permanent lexical entries for future interactions. We also use word embeddings to find possible synonyms when a new perceptual word is encountered, touching on the synonymy problem handled more fully in Chapter 4 using a method similar to that used to guide behaviors in Chapter 7.

Other work in the dialog community has used Mechanical Turk to carry out experiments despite the difficult nature of dialog (long, engaging task, compared to most Mechanical Turk HITs, which are fast lists of questions) (Ring et al., 2016;

Su et al., 2018). In our evaluation, Mechanical Turk workers have a harder time with the tasks and finish them less often than in the evaluation performed in Chapter 3. The tasks are quite difficult: users were asked to have three full dialogs with the agent, and the agent asked many clarification questions together with questions about whether words applied to objects (to learn better perceptual concept models). Creating a more focused evaluation per individual worker may have ameliorated these effects, and future human-robot dialog evaluations done through crowdsourcing should be wary of the mental strain these tasks put on workers.

Methods

We have implemented a conversational dialog agent that uses a semantic parser to translate human utterances into semantic meaning representations, then grounds those meaning representations using both a static knowledge base of facts about an office environment and perceptual concept models that consider multi-modal representations of physical objects. 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.

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 to better understand future human user commands. Additionally, these conversations provide predicate-object labels from questions selected using opportunistic active learning, enabling us to build and refine perceptual concept models to better ground perceptual descriptions of objects given by human users.

Table 8.1 enumerates some of the recurring terminology and symbols we use in this section, for clarification and reference. The following subsections discuss the semantic parser, perceptual concept models, language grounding formalism, agent dialog policy, and the learning paradigm used to carry out the evaluations.

Symbol	Description
\mathcal{A}	A dialog agent
CCG	A set of CCG syntactic categories
D	A set of pairs of language strings and semantic forms
\mathcal{C}	The set of sensorimotor contexts of explored objects
\mathcal{D}	A grounding procedure producing denotations
\mathcal{G}	A perceptual classifier
\mathcal{L}	A lexicon
P	A set of predicates
\mathcal{P}	A parser
\mathcal{O}	An ontology
S	A set of semantic forms
\bar{S}	A parser score vector
T	A set of word tokens
\mathcal{U}	A top-down parsing generation procedure
θ	A parser parameter vector
κ	The confidence of a perceptual classifier
π	A dialog policy
ϕ	A parser feature vector

Table 8.1: Symbols used throughout this chapter.

Semantic Parser

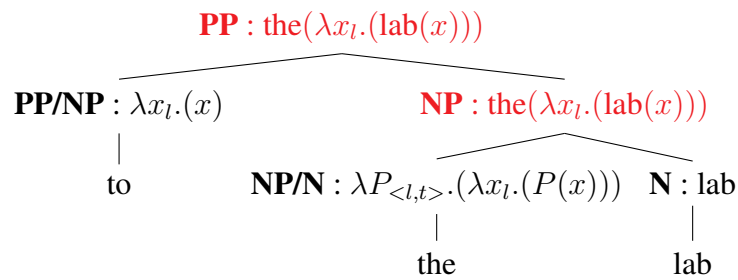
Semantic parsing translates human language commands to meaning representations that a robot can reason with. Learning to perform these translations is non-trivial and requires an expert annotator to create an ontology of concepts, a lexicon that maps words to compositions of those concepts, and examples of sentences paired with their meaning representations. These paired examples are used to learn a set of parameters in the parser for performing structured inference—translating the linear sequence of words into a compositional meaning structure.

In this work, we use the Combinatory Categorical Grammar (CCG) formalism (Steedman and Baldridge, 2011) in our lexicon to perform Cocke-Kasami-Younger (CKY) chart parsing (Younger, 1967) on input sentences. Our parser is functionally similar to the University of Washington Semantic Parsing Framework (Artzi and Zettlemoyer, 2013a), but we use a unification-based training procedure rather than a template-based procedure. Further, we add ontological entries dynamically during conversations with human users (for example, when a new perceptual concept like *red* is used for the first time). Finally, we use word embeddings to guide parsing when out-of-vocabulary words are present in an input utterance. Each of these differences is discussed in more detail below.

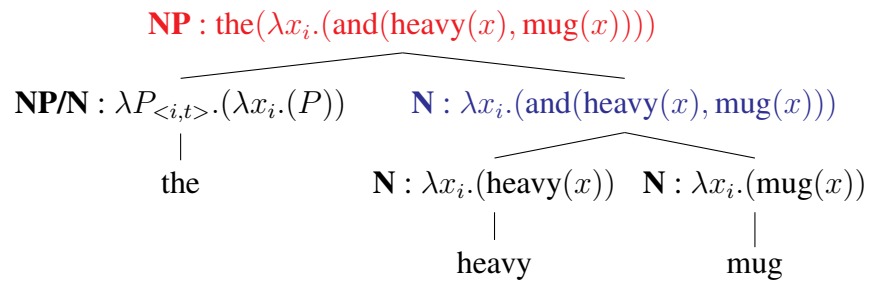
Given the syntactic tree of a sentence, meaning representations can be propagated from the leaf level to the root to find the composed meaning (Figures 2.1 and 8.2). We construct a parsing framework with the abstractions necessary for facilitating grounding and utilizing word embeddings in mind. The source code for this parsing framework is available at <https://github.com/thomason-jesse/tsp>. Next, we briefly describe the semantic parser’s design and training paradigm.

A parser $\mathcal{P} : \mathbb{P}(T) \times \mathcal{L}_{\mathcal{O}} \rightarrow S$ takes in a sequence of word tokens $x = (x_1, x_2, \dots, x_N) \in \mathbb{P}(T)$ 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} and outputs a semantic parse $s \in S$, the set of all semantic parses possible from the ontology \mathcal{O} .

In this work, the ontology \mathcal{O} includes constants—people (*alice*), objects (o_3), and rooms (r_2)—and predicates—room types (*lab*), relations (*westof*), and



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



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

Figure 8.2: CCG-driven λ -calculus parses of the expressions *to the lab* and *the heavy mug* to demonstrate the **function application** and **merge** parsing rules.

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	$\lambda x_p.(\lambda P_{\langle l, t \rangle}.(\text{the}(\lambda y_l.(\text{and}(P(y), \text{pos}(x, y)))))$
next	N/PP\N	$\lambda P_{\langle l, t \rangle}.(\lambda x_l.(\lambda y_l.(\text{and}(P(y), \text{adjacent}(x, y)))))$
david	NP	david
lab	N	lab
metallic	N	metallic

Table 8.2: Some lexical entries created from the utterances given in the initialization phase of the experiment described in Section 8.3. 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. *person*, *item*), which helps constrain search during parsing. Expected input and output types are specified in brackets ($\langle \text{input}, \text{output} \rangle$). The special symbol *the* functions to select a single atom among those specified by its λ -headed argument (i.e. to find the atom that satisfies the argument expression).

perceptual concepts (*small*). Each constant is associated with a type, such as *person* or object *item*. Each predicate specifies a number of expected arguments and their types, and returns true or false (e.g. *red* is a predicate that tells whether an argument item constant is red in color).

The lexicon \mathcal{L}_O is a data structure that contains information about how individual word tokens relate to ontology \mathcal{O} , and what CCG category they belong to. The lexicon is a set of triples, $(t, c, s) \in \mathcal{L}_O$ 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 8.2 gives some example lexicon entries from the initialization phase of the experiment described in Section 8.3.

The meaning of a sequence of tokens x is the root of its semantic parse tree. If each token x_i has an entry in \mathcal{L}_O , we know the CCG categories of each. The CCG

categories determine how the semantic forms of the tokens compose to form a tree. CCG categories can combine by *function application*, for example when a prepositional phrase is formed from a preposition and a noun phrase (Figure 8.2a). CCG categories that are identical can *merge*, and the resulting semantic form is headed by an *and* predicate whose children are the forms that were merged (Figure 8.2b).

The examples in Figure 8.2 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 an *the* predicate with said λ child (e.g. $\text{the}(\lambda x : i.(\text{lab}(x)))$). This type-raising increases the size of the lexicon and the polysemy in searching over bare noun words, but makes parsing ungrammatical phrases in user responses, such as *bring cup*, possible.

Given a lexicon $\mathcal{L}_{\mathcal{O}}$, its corresponding ontology \mathcal{O} , and a set of pairs $(x, r) \in D$ of token sequences and root semantic forms, we train a statistical parser following the methodology presented in prior work (Liang and Potts, 2015). In particular, we define a feature vector $\phi(x, y) \in \mathbb{R}^N$ where every entry i in $\phi(x, y)_i$ represents some value relating the input utterance x to the latent semantic tree y (rooted at the corresponding r), or between constituent nodes of the tree y , with N being the total number of such feature values. A summary of the features we consider is given in Table 8.3. Next, we discuss how the latent tree y is discovered from the pair (x, r) to facilitate training.

Parsing an Input Utterance

Given a parameter vector θ of length $|\phi|$, we score parse trees \hat{y} as the dot product $\theta \cdot \phi(x, \hat{y})$. We set θ so that each subset of feature types is parameterized by a probability distribution, such that the likelihood of the parse \hat{y} given x is equivalent to $\theta \cdot \phi(x, \hat{y})$. 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 to attempt to find the \hat{y} parse with the

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 the parent-child relationships in the semantic root of y
Skip given token	Entries indicating, for each token, whether it was skipped

Table 8.3: 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, accounting for child argument positions.

minimum negative log likelihood (highest probability).

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 considered by each of these stages is limited to a fixed-size beam. These greedy 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 according to its CCG category in C_x , and composing these semantic leaves according to the structure of C_x to form a completed semantic tree \hat{y} . Because this procedure is controlled by beam searches, additional completed semantic trees can be sampled by continuing the process after the highest-probability tree is produced. For full details of the implementation and hyper-parameters for this process, we refer to reader to the source code linked above.

Training the Semantic Parser

To learn values for the parameter vector θ , we consider all the pairs $(x, r) \in D$ (in random order), and 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:

$$y^* = \operatorname{argmax}_{\hat{y} \in P(x)} (\theta \cdot \phi(x, \hat{y}));$$

$$y \in P(x) | \operatorname{root}(y) = r.$$

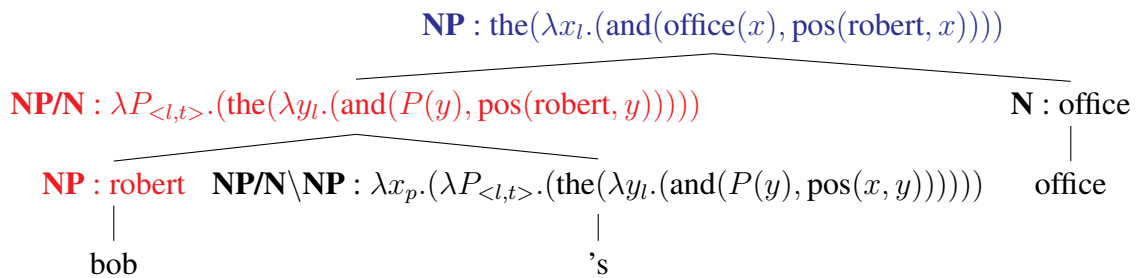
That is, y^* is the latent parse tree produced by $P(x)$ that maximizes the current vector of parameter weights, θ , while y is the latent parse tree produced by $P(x)$ that is rooted at the target semantic meaning r . If $y^* = y$, then θ correctly guides \mathcal{P} to produce the desired parse from the given input. If y^* is never found, then the beam search terminated before a valid parse is found.

If y is not found, \mathcal{P} may not have the lexical entries necessary to translate x to r , so we attempt to produce new lexical entries to bridge the gap. Given a partial parse, $Y = (\hat{y}_0, \hat{y}_1, \dots, \hat{y}_k)$, a sequence of parse trees rooted at spans of x but failing to come together under a single root, and the target root r , we can perform a *top-down* search from r by reversing the function application and merge rules for the CCG parsing we use, described above. Top-down parsing by reversing our parsing rules is conceptually similar to previous work on inducing CCG grammars using higher-order unification operations (Kwiatkowski et al., 2010). We denote the lexical entries derived from this procedure $\mathcal{U}(Y, r)$. Figure 8.3 gives an example.

In this way, we can derive new candidate lexical entries for \mathcal{L} based on the training data D . We add $\mathcal{U}(Y, r)$ to \mathcal{L} and the resulting latent tree y (rooted at the target r for input pair $(x, r) \in D$) is derived. Given non-matching y^* (the top-scoring parse in $P(x)$ given the current θ) and y (the parse rooted at the target semantic form r), we update θ as follows.

The parser \mathcal{P} maintains a running score vector, \bar{S} , of every feature. This vector \bar{S} is parallel to both the features of ϕ and the parameters θ . 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:

$$\bar{S}_i \leftarrow \bar{S}_i + \alpha(\phi(x, y) - \phi(x, y^*))_i, \quad (8.1)$$



(a) A **top-down** search from the **known root** on the noun phrase *bob's office* is used to reveal 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 8.3: An example of *top-down* completion for the parse of *bob's office* and the resulting lexical entry. In this example, *bottom-up* parsing was unable to proceed past the leaves of the tree. In general, *top-down* parsing does not begin until *bottom-up* has produced subtrees that can no longer be combined.

for a learning rate α (in our experiments, $\alpha = \sqrt{D}$). This training procedure is similar to the hinge-loss training used in previous work on learning statistical semantic parsers (Liang and Potts, 2015), but does not enforce a hinge penalty. The hinge penalty can encourage over-fitting by forcing the parameters of the parser to adhere more closely to training data, so we remove it due to the known small size of our training set. In general, our training procedure is simply a stochastic gradient descent method that minimizes the negative log likelihood of the correct parse y given the scores \bar{S} . Given this vector of scores, we calculate θ as a collection of negative log probabilities over the feature types enumerated in Table 8.3.

This training procedure iterates over the examples in D in random order, updating \bar{S} for each pair in D , and updating θ after passing over all of D . This procedure is repeated for a fixed number of epochs, or until all $y_i^* = y_i$ across D (convergence).

Using Word Embeddings to Aid Semantic Parsing

In order to assign a semantic meaning to a sequence of tokens x , each token x_i that is not skipped must have an entry in the lexicon \mathcal{L} from which its semantic meaning can be determined. Alternatively, when a known semantic root r is available during training, top-down parsing can continue from where bottom-up can no longer continue, discovering new potential meanings for previously unknown words x_i . At test time, a known root r is unavailable. Therefore, we use word embeddings (Mikolov et al., 2013) to augment the lexicon at test time to attempt to recover from out-of-vocabulary words, an idea similar in spirit to previous work (Bastianelli et al., 2016), but formally integrated into our parsing pipeline.

Specifically, for an unknown word x_i at test time, 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 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} , $\text{sim}(x_i, t)$. If their similarity is 1, the θ value is copied, otherwise a similarity-weighted penalty is added to the θ score. After this process is repeated for every unseen x_i in x , parsing can proceed as described above.

At test time, this augmentation helps recover from out-of-vocabulary words, such as *grab* being used for *deliver*. In Section 8.2.5, we will discuss how, with completed conversations, recovering from an out-of-vocabulary word by inducing a lexicon entry at test time can lead to adding a permanent lexicon entry at a subsequent training step, after the true semantic root r for the sequence x containing the out-of-vocabulary word is inferred from the conversational context.

Multi-modal Perception

Once a command has been translated into a semantic form, *grounding* that semantic form to actions, objects, and rooms in the real world must take place before the robot can act on the command. For objects, perceptual concepts like *red* and *heavy* require considering sensory perception of physical objects. We build multi-



Figure 8.4: **Left:** The robot used in our experiment. **Right:** The objects explored by the robot for grounding perceptual predicates.

modal concept models to connect sensory perception to concept labels. First, we extract multi-modal feature representations across various sensorimotor contexts by exploring those objects with a robot arm.

The robot used throughout this work is a Kinova MICO arm mounted on top of a custom-built mobile base, pictured in Figure 8.4 (Left). The robot’s perception includes joint effort sensors in each of the robot arm’s motors, a microphone mounted on the mobile base, and an Xtion ASUS Pro RGBD camera. The set of objects O used in our experiments consists of 32 common household items including cups, bottles, cans, and other containers, shown in Figure 8.4 (Right), explored by a robot arm (Sinapov et al., 2016). Some of the objects contain liquids or other contents (e.g., coffee beans) while others were empty. These are the same objects, explored with the same behaviors, as those presented in Chapters 4 and 5.

The predicates P in our ontology can be based on static facts from a knowledge base or learned perceptual concepts. We partition P into P_s : the subset of predicates referring to static facts, and P_c : the subset of predicates referring to perceptual concepts, so that $P = P_s \cup P_c$ and $P_s \cap P_c = \phi$.

Given a 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 $c \in \mathcal{C}$, using its feature representation of the objects. We call this ensemble a predicate classifier, \mathcal{G}_p . At the SVM level, a decision on object o is the majority vote of its observations \mathcal{X}_o^c for the relevant context c in which the SVM operates. That is, for

every observation of o in c (e.g., multiple photographs of the object from different angles from the *look* behavior), each observation is evaluated by the SVM for c and their majority decision is used for ensembling in \mathcal{G}_p . 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 context SVMs, with a weight for each SVM equal to its reliability estimated using leave-one-object-out cross validation on the available labeled objects for p . The decision is accompanied by a confidence $\kappa_p(o) \in [0, 1]$ based on the sum of these reliabilities, and derived from the Cohen’s kappa agreement between the true labels and the predicted labels during cross-validation (negative κ values are rounded up to zero). Full details regarding predicate classifiers are presented in Chapter 4.

If multiple labels are given for the same predicate and object, the majority class label is used during training. When the robot interacts with multiple people, differing opinions can produce this situation (for example, people differ on whether color words like *red* and *orange* apply to objects of a color between these two). In the event of a tie, that object is not considered when training the predicate classifier for the concept. 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). The confidence κ values are also used to drive an opportunistic active learning strategy for improving concept models during conversations, as described in Section 8.2.4.

Language Grounding

To execute a command, an utterance is first translated into a semantic form. For the robot to act on that form, its λ variables must be instantiated. In our work, this is done by querying a knowledge base of facts available to the agent, together with perceptual concept models the agent learns over time from user interactions.

Given a semantic root $r \in S$, the goal of a grounder is to instantiate all the 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 , which contains only semantic forms with no λ expressions. In particular, the ground-

ing denotation procedure can be defined as $\mathcal{D} : S \times P \rightarrow S \times [0, 1]$, such that $\mathcal{D}(r) = \{(g_1, c_1), (g_2, c_2), \dots, (g_k, c_k)\}$, 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 unviable—there is no way to satisfy the constraints on the λ variables in r —while a confidence score of 1 means the grounding is unambiguous—it is the only solution to instantiating the λ variables in r .

We implement a recursive grounding procedure to gather the set of possible grounded parses g_i and confidences c_i . At a high level, 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 $p \in P$ being applied to some constant $a \in \mathcal{O}$. For $p \in P_s$ static fact predicates, $p(a)$ is retrieved from a table as $g = true$ or $g = false$ with confidence $c = 1$. Static facts include room types (*office*) and relations (*owns(robort, 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 possibilities are returned: $g_j = true$ with $c_j = \kappa_p(a)$ when $p(a) = 1$ and $g_k = false$ with $c_k = (1 - \kappa_p(a))$ when $p(a) = -1$, where κ_p denotes the confidence in the decision. For the *and* predicate, *true* is returned if the children of *and* match with confidence equal to the product of the child confidences (implicitly assuming independence). For predicates in P that return instances, such as the *a* predicate, variable instantiations that have resulted in *true* values are returned along with their confidences. Since our domain is closed-world (e.g. we know all the objects, people, and rooms in the discourse), we normalize these confidences to form a probability distribution over possible instantiations.

Figure 8.5 gives an example denotation propagated through the parse of *a heavy mug* for a subset of two objects (though in reality all objects in \mathcal{O} are considered), with the final confidences in the objects returned by the predicate *a* normalized to form a probability distribution. This procedure runs on parses of complete commands (e.g. *move a heavy mug from the kitchen to alice’s office*) or simple noun phrases like *a heavy mug* that appear as responses to clarification questions. The

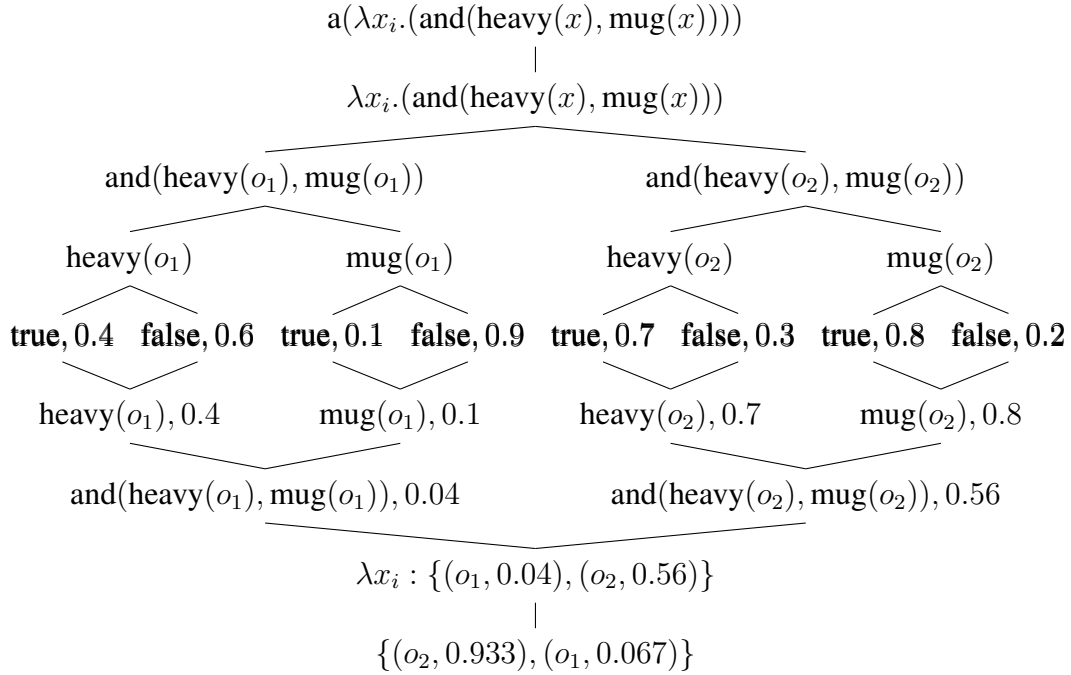


Figure 8.5: Graphical depiction of the grounding procedure for the semantic parse of *a heavy mug* for a toy example world with only two physical objects, o_1 and o_2 . The λ variable is instantiated for every object in the world, and the confidence of the predicates applied to those instances are propagated to derive a confidence in each denotation. In the end, the a predicate induces a probability distribution over possible denotations of the λ variable.

probability distribution based on confidences returned with the list of possible denotations is used as part of an update procedure for helping the agent understand the user’s intent, as described in the next section.

Dialog Policy

We implement a conversational dialog agent \mathcal{A} for command understanding similar to that in Chapter 3. The differences between this agent and the previous one are: 1) grounding semantic parses in both static knowledge and perceptual knowledge; 2) dynamically adding new ontological predicates for novel perceptual concepts; 3) leveraging opportunistic active learning for refining perceptual con-

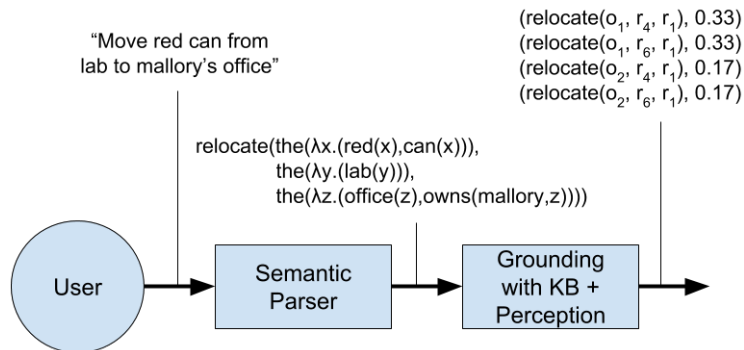


Figure 8.6: Example semantic parse ($\mathcal{P}(x)$) and grounding with static (Knowledge Base) information and perceptual knowledge ($\mathcal{D}(\mathcal{P}(x))$) in a toy world with two objects, one room belonging to *Mallory*, and two rooms that are *labs*.

cept models on-the-fly; and 4) semantic parser training from pairs of utterances and denotations. The agent engages in dialogs with human users to refine its understanding of natural language commands by inducing additional training data for its parsing and perceptual grounding components.

Clarification Dialog Policy

Dialog begins with a human user commanding the robot to perform a task. The agent maintains a belief state modeling the unobserved true task in the user’s mind, and uses the language signals from the user to infer it. In our experiments, one user issued the command *Move red can from lab to mallory’s office* (Figure 8.6). The command is first parsed by the agent’s semantic parser \mathcal{P} (Section 8.2.1), then grounded against static and perceptual knowledge with denotation procedure \mathcal{D} (Section 8.2.3), which results in a list of pairs (g_i, c_i) of denotations of the semantic parser’s understanding of the command and associated confidence values. Using this list of denotations and confidences, we update the agent’s belief state, then engage in a clarification dialog to refine that belief.

The agent’s belief state, \mathcal{B} , is a mapping from semantic roles (components of the task) to probability distributions over the ontological constants that can fill those roles. The belief state models the *action*, *patient*, *recipient*, *source*, and *goal*

semantic roles. Different actions utilize different roles. For example, the *walk* action takes only a *goal* argument (the room to walk to), while the *deliver* action takes both a *patient* (the object to be delivered) and a *recipient* (the person to deliver to).

In our experiments, the possible actions are *walk*, *deliver*, and *relocate*. Walking instructs the robot to move to a different room, delivering to take a physical object to a person, and relocating to move a physical object from one room to a different room. The belief state for the *action* role is initialized with uniform probabilities across these three actions. The remaining role belief states are initialized with half of the initial probability mass on an *unknown* constant, \emptyset , indicating that the role is not known or is not necessary for the action the user has in mind, and the remaining half of the probability mass is distributed uniformly across all constants that can fill the role.

For a user utterance x and associated denotations $(g_i, c_i) \in \mathcal{D}(\mathcal{P}(x))$, this 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 8.6, the distribution over *action* will put all probability on *relocate*, while the distribution over the *patient* (object to be relocated) will assign 0.66 of the mass to object o_1 and 0.34 of the mass to object o_2 .

We call this collection of distributions \mathcal{B}_x , a mapping from semantic roles to the distribution over constants that can fill them based only on the most recent user utterance x . Then we update the agent’s belief according to:

$$\mathcal{B}(r, a) \leftarrow (1 - \rho)\mathcal{B}(r, a) + \rho\mathcal{B}_x(r, a), \quad (8.2)$$

for every semantic role r and every constant a . The parameter ρ controls how much to trust 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. The highest-probability constant for every semantic role in the current belief state \mathcal{B} , together with which among those roles has the least probability, are used

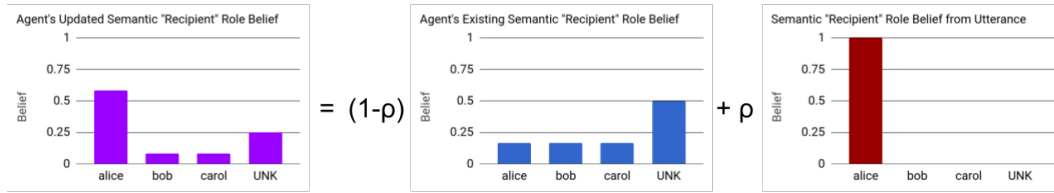


Figure 8.7: Example belief update for the *recipient* role, where the initial agent belief (uniform over people with half of the prior on \emptyset , or UNKnown) is interpolated with an unambiguous belief that the recipient should be *alice*. This could arise, for example, from the question *To whom should I bring something?* being answered *alice*.

to select a question. A static dialog policy π that chooses a follow-up *confirmation* or *clarification* question from these. Confirmation questions ask whether a certain constant in a role is correct. Clarification questions ask the user to rephrase a referring expression for what should fill a role. Table 8.4 gives some examples of the policy π .

For 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 8.2, such that \mathcal{B} reflects the confirmation. 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, such that the update in Equation 8.2 reduces the belief only for denied constants (e.g. *I should walk to room 1?* denied decreases both walking as the *action* and room r_1 as the *goal*). However, a previously confirmed role never loses probability mass (for example, once a user confirms *walking* as the action, if the agent asks if 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. Then the constants for each semantic role in B (which each have the whole probability mass for their respective role from being confirmed) can be passed on to a robotic planner to be executed in the environment.

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

Table 8.4: Samples of the agent’s static dialog policy π for mapping belief states to questions. In the Mechanical Turk experiments described in Section 8.3, constants like people (p_1), objects (o_1), and rooms (r_1, r_2) were represented pictorially, with pronouns (*this person, this, here, there*) in place of their variables in the sentence shown.

Detecting Perceptual Words and Synonyms

When describing objects in the real world, humans can use words the agent has never heard before. Some of these are perceptual concepts—words that need to be grounded in the physical world. In prior work, a stopword list is used to remove non-content words, and all content words in human descriptions of objects are considered perceptual concept words. This generality resulted in building unnecessary and confusing concept models for non-perceptual words like *thinking*, extracted from utterances like *I am thinking of a red bottle*. In this work, we take a more robust approach that hypothesizes whether a new word might be perceptual based on known perceptual words, then asks the human user for confirmation.

For an input sequence of tokens x in a user utterance, we identify all words for which the parser, \mathcal{P} , does not yet have lexical entries. If one of the nearest neighbors (in our work, we look at the nearest 3) of x_i in \mathcal{P} according to word-embedding distance has a semantic form involving a perceptual predicate $p \in P_c$, we ask 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*

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 8.8: Web interface example of learning *tall* is a synonym of *long*. Possible synonyms are selected by word vector embedding similarities.

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 induced lexical entries, as described in Section 8.2.1). If the user answers *yes*, we attempt to discover whether x_i is a synonym of an already known perceptual concept, such as one of the identified neighbors.

We rank the nearest neighbors of x_i by distance and sequentially ask 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 , such that they share an underlying perceptual concept model. For example, in our experiments, previously unseen word *tall* was added as a synonym for the known word *long* (Figure 8.8).

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} with syntax and semantics parallel to other concept words 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} , we search for parses in $\mathcal{P}(x)$ as described in Section 8.2.1. In this way, we both expand the ontology to represent new concepts and partially account for concept synonymy, fighting the concept-object label spar-

sity introduced by treating all new words as new concepts, as was done in Chapters 4 and 5.

Opportunistic Active Learning during Conversation

A human user may use a novel perceptual word during a conversation to describe an object. Additionally, the concept models for known perceptual words may be unreliable due to few labeled objects. We introduce opportunistic active learning questions as a sub-dialog routine for the agent, in which it can query 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 the *remote* test object items (e.g. items that are physically in a different room but being discussed in the conversation). We call these local objects the *active training set*, and the remote objects the *active test set*. Objects in the active training set can be labeled by asking the human questions during a conversation about whether particular predicates apply to them.

In Chapter 5, we established that an agent asking questions about both on-topic (used in the current conversation) and off-topic (irrelevant to the current user’s needs) predicates outperformed an agent that only asked about predicates in the current human description. Thus, we allow our agent to ask both on- and off-topic questions, moving to off-topic ones only if there are no more useful on-topic labels to query.

After the human user gives their initial command, x , a sub-dialog for improving perceptual concept models begins. 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 divide 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 (such that $(P_C^x \cup \bar{P}_C^x = P_C) \wedge (P_C^x \cap \bar{P}_C^x = \emptyset)$). A predicate p is considered “in the command” if it is present in the logical form of any token $x_i \in x$ according to the lexicon \mathcal{L} . To determine potential active-learning queries, we use the confidence $\kappa(p, o)$ for $p \in P_C$ and $o \in O$ based on the κ agreement with

known labels when cross-validating the classifier of p against the labeled objects for p (Section 8.2.2). 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 $d(p, o) = -1$.

We fix the maximum number of questions, M , to ask (in our experiments, as in past work, we set $M = 5$). Questions are asked about on-topic predicates, $\hat{P}_C = P_C^x$ until there are no more questions that can be formulated for these predicates about the available *active training set* of objects, O_{tr} . A question cannot be formulated for a predicate if all the objects in O_{tr} are labeled for that predicate already. The robot then continues with $\hat{P}_C = P_C$ until M questions have been asked or the same is true of P_C . That is, the robot is allowed to ask about off-topic predicates once on-topic, label-gathering questions have been exhausted, but may still stop short of M questions if there are no label-gathering questions to ask about any predicates.

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

$$K_{te}(p) = |O_{te}|^{-1} \sum_{o \in O_{te}} \kappa(p, o). \quad (8.3)$$

The robot decides which predicate to ask about next with probability proportional to 1 minus this confidence:

$$pr(p) = \frac{1 - K_{te}(p)}{\sum_{q \in \hat{P}_C} (1 - K_{te}(q))}. \quad (8.4)$$

After selecting a predicate p^* to query by sampling from this probability distribution, the robot must decide whether to ask for a label of a particular object among the active training set, for a positive example object, or for a negative example object. This decision differs slightly from Chapter 5, where we did not query for negative example objects, and chose between question types with a fixed probability.

We calculate the confidence of p^* among active training set objects $K_{tr}(p^*)$

similar to the calculation for the active test set (Equation 8.3). If $K_{tr}(p^*)$ falls below a fixed threshold, we consider the classifier untrustworthy for determining a useful specific object in need of a label, and the robot instead asks for a new positive or negative example (in our experiments, we set this threshold to 0.7). In this case, if the labels for p^* on O_{tr} are majority-class positive, the robot 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 robot 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 described in the next section, users are presented with images of all objects O_{tr} and can click one as an 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 $K_{tr}(p^*)$ is above the confidence threshold, we trust the classifier of p^* to identify an object $o^* \in O_{tr}$ whose label would be most valuable to obtain:

$$o^* = \operatorname{argmin}_{o \in O_{tr}} (\kappa(p^*, o)). \quad (8.5)$$

We frame this question as *Would you use the word ‘ p^* ’ when describing o^* ?* In the Mechanical Turk interface, o^* in the string above 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), we sample a new p^* from the distribution of predicates in \hat{P} (Equation 8.4), and formulate a question for it instead.

In this way, the agent is able to query the user for labels on the objects O_{tr} to improve its perceptual classifiers before continuing its clarification conversation.

We perform this active-learning sub-dialog regardless of the nature of the user command, including, for example, *navigation* commands that do not require perception. In that case, $P_x = \emptyset$, $\hat{P} = P$, so the agent can ask about any predicate that needs improvement (that is, all predicates are off-topic if the command does not require perception). The perception sub-dialog ends when no more useful questions can be asked (all predicates are labeled for all objects O_{tr}) or M questions have been asked. The agent concludes with *Thanks. Now, back to business.* and the clarification dialog resumes.

Learning from Conversations

In this work, we use the same basic parser retraining strategy as in Chapter 3; however, a key difference in our current retraining procedure is that it makes a distinction between denotation parses and latent semantic parses. That is, we differentiate between *robert's office* and its denotation, r_1 , where in the strategy presented in previous work, the denotation would be treated as the latent form of *robert's office*, weakening its ability to generalize from induced data and creating more reliance on a robust initial lexicon. We first induce utterance-denotation pairs from conversations, then induce latent semantic forms that connect those utterances and denotations.

Inducing Utterance-Denotation Pairs from Conversations

Every time the agent asks the user a clarification question, it is associated with the least-confident role r being queried (or *All* roles, if the conversation is just starting), as shown in Table 8.4. When a user answers a clarification question, we obtain a soft alignment between the constant that should fill role r and the human's natural-language answer. When the conversation concludes, the final chosen task b gives the action and arguments. Thus, for every r in chosen task b , where $b(r)$ 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*. If the final inferred task is $\text{deliver}(o_1, p_1)$, we create a pair $(a\ \text{heavy}\ \text{mug}, o_1)$ where o_1 is the denotation object referred to by *the heavy mug*. Furthermore, we can pair the original user command (e.g. *bring the mug to bob*) with the final task denotation, $\text{deliver}(o_1, p_1)$. Figure 8.9 gives an example of a conversation where we can pair both the final task denotation with the original user command and a clarification question response with a denotation role (the *goal*). Rather than training the parser directly on these pairs, it is important to discover the latent semantic form between them. In particular, the latent form should be plausible given the utterance and also be translated into the denotation by the agent’s grounder after consulting world knowledge. Next, we describe how we perform this latent form induction.

Inducing Utterance-Semantic Form Pairs from Utterance-Denotation Pairs

Previously, we have discussed taking an input utterance as a sequence of tokens x to produce a logical semantic form $(y, c_p) \in \mathcal{P}(x)$ by invoking a semantic parser \mathcal{P} (here, c_p denotes the confidence score the parser assigns to y , which is a log likelihood plus fixed penalties). Additionally, we have detailed how a semantic form y can be grounded, instantiating its λ expressions by consulting predicates that operate on static and perceptual knowledge, to produce a denotation $(g, c_d) \in \mathcal{D}(y)$. Here, we explain how to induce a latent form $y \in S$ given a pair (x, g) such that $y \in \mathcal{P}(x)$ and $g \in \mathcal{D}(y)$. That is, a semantic parse that can be derived from the input utterance and has a denotation matching the known one for that utterance. We formulate this training and finding of the latent semantic form similar to past work on learning statistical, compositional semantic parsers (Liang and Potts, 2015).

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. First, 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 because it was produced from x and can be instantiated to the denotation g . Let such matching parses be

ROBOT What should I do?

YOU go to the middle lab

ROBOT You want me to go somewhere? (**not** manipulate any objects)

YOU yes

ROBOT Where should I go?

YOU the lab in the middle

ROBOT You want me to go to **here** (**not** manipulate any objects)?

YOU yes

YOU type your response here...

Say

The robot navigates to **here**.

To advance to the next task, click the button below.

Okay

Figure 8.9: 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* representing the center lab in the diagram. This interface was used in our Mechanical Turk experiments, discussed more in Section 8.3.2.

represented by y_i^* .

We then score each parse 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:

$$\text{score}(y_i^*) = c_{p,i} + \log(c_{d,i,j}). \quad (8.6)$$

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 limit the number of latent forms to 10 (e.g. $i \in \{1, 2, \dots, 10\}$) during search.

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

$$y^* = \operatorname{argmax}_{y_i^*}(\text{score}(y_i^*)). \quad (8.7)$$

In this way, $y^* \in \mathcal{P}(x)$, $g \in \mathcal{D}(y^*)$, and y^* has the highest interpolated log likelihood between the parsing and grounding stages connecting x and g . Now the pair (x, y^*) can be used as training data to refine the parser \mathcal{P} . Table 8.5 gives an example for finding the latent semantic parse between *the heavy mug* and object o_2 , the grounding for which was given as the example in Figure 8.5.

Evaluation

We evaluate our agent with hundreds of users through the Mechanical Turk interface. The agent has access to static facts about the world as well as multi-modal object feature representations against which it can train perceptual concept models. We deploy the agent with human users who instruct 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 demonstrate that the agent better understands human commands, and is rated more favorably by human users, after training its parsing and perception components from dialogs. In particular, training both parsing and perception components outperforms training only the perception components, and training neither (e.g. comparing just to the initial agent).

$y \in \mathcal{P}(x)$	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 8.5: Example of finding the best latent parse for x : *the heavy mug* given known, correct denotation o_2 . In this example, the alternative parses arise from the parser choosing to skip individual words in x , but, in general, alternatives can arise from things like prepositional phrase attachment ambiguity, lexical polysemy, and more. For the purposes of demonstration, in this example the parser favors skipping the noun *mug*, but the re-ranking score after grounding shows that including it leads to a better parse. The highest score is marked in **bold**, and is paired with the best latent parse for this utterance-denotation pair.

After training the agent using data from conversations it had with users on Mechanical Turk, we bootstrap an embodied agent on a physical robot using what it learned. We demonstrate this physical robot and its ability to carry out the tasks considered in the Mechanical Turk experiments in an actual office setting. While Mechanical Turk limits users to describing objects only by their visual properties, the embodied agent, initialized from Mechanical Turk, can begin to learn non-visual concepts like *heavy*.

In the remainder of this section, we discuss the experimental design setup, the Mechanical Turk evaluations, and the implementation on the physical robot.

Experiment Design

We deploy the agent in a simulated office environment similar to the real world office our physical robot operates in at a university campus, with the same floorplan but anonymized names and titles for the people on the floor. This environment is populated by rooms, people, and object items. Facts about the rooms and people are static and represented in a fixed knowledge-base. Facts about the object

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

Table 8.6: 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), together with the arguments taken by each task. For relocation, note that the source and goal locations are selected to always differ.

items have to be learned by building perceptual concept models (Section 8.2.2).

The agent can perform three kinds of tasks: navigation, delivery, and relocation. We fix 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 use 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 8.6 gives the total number of tasks for each action type as well as how many were included in these subsets.

Including the same 8 objects in all sets ensures that the perceptual concepts needed to identify those objects are the same across conditions. Otherwise, variance would be introduced by new objects being described in the test condition distinct from those in the training condition, where many new words with no corresponding concept models might be used. The objects that appear in these tasks are never visible to the predicate concept models during training.

Initialization Phase

Static facts about the predicates in the ontology \mathcal{O} that do not involve physical perception need to be made available to the agent. At the same time, predicates in \mathcal{O} that do require physical perception need to be connected to untrained concept models whose labeled data can then come from opportunistic active learning dur-

ing conversations with human users. To initialize a semantic parser, we run a small initialization phase using the web interface described in Section 8.3.2.

Sixteen users (graduate students at the university across several fields) engaged with a faux-agent using the web interface. They were shown three tasks (one navigation, one delivery, and one relocation) to instruct the agent to complete, each drawn from the initialization set. After sending their command, the faux-agent simply replied *What's another way you could phrase that command?* In this way, each user supplied two versions of the same command for each of the three tasks.

We used these commands as a scaffold on which to build an ontology \mathcal{O} that could represent the concepts users invoked to talk about locations and object items. In particular, for the 26 locations in our map, in addition to semantic map labels like *lab* and *office* (7, in total), we created 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 in the map and their offices (ownership), and for a room to be unoccupied, there were 19 predicates involving locations. For the 32 object items, 20 initial perceptual concepts were used to refer to the 8 objects involved in the delivery and relocation tasks.

We also annotated a lexicon \mathcal{L} based on the gathered utterances. 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 from the initialization commands. We also added prepositions and determiners as they were used in commands, action word synonyms (e.g. *visit* for the walking action), and polysemous action constructions (e.g. *move A from B to C* versus *get A from B and take it to C*). Several kinds of confirmations were added, such as *yes*, *yep*, and *sure*. Finally, we added single lexical entries for people in the office environment based on their first names, like *robert* for person *b* (the directory of people shown in the interface elucidates these relationships). Note that while the object items and rooms exist in the ontology and can be part of denotations, they cannot be directly referenced because they have no lexical entries. Thus, users must always describe objects using their physical properties, and rooms using their labels and spatial relations to other rooms.

We then hand-annotated a subset of the commands with their corresponding semantic parses. This gave us initialization pairs D_0 with which to train an initial parser given the crafted ontology and lexicon. We trained an initial parser with the $|D_0| = 44$ pairs annotated from the initialization. The performance of this initial parser, with perception modules that are initially blank (but can be learned on-the-fly using opportunistic active learning), is what we compare against after retraining the parsing and perception components, as presented in Section 8.3.2.

When creating these initial parsing resources, we made a few decisions to simplify the language understanding. First, though some commands used negation (e.g. *the pod not next to a conference room*), we chose not to include negation in our parsing model due to its sparse usefulness in the task. Additionally, we do not handle 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]*). As a result of these decisions, and several commands that would have introduced single-use ontological concepts, the number of pairs ($|D| = 44$) is lower than the 72 commands we gathered from 12 users providing 2 commands each for 3 tasks. Next, we discuss the training procedure used to update the parsing and perception modules after batches of users have conversed with the agent.

Training Procedure

We use these initial parsing resources to create a baseline agent \mathcal{A}_1 with a parser \mathcal{P}_1 trained only on the initialization pairs D_0 mentioned above and concept models for several predicates $P_{c,1}$, but with no initial object examples against which to train them. Our work focuses on learning from conversations, and so, past the initialization phase described above, all learning for the parser and perception modules arises naturally from conversations the agent has with humans. To elicit training conversations, we provide each Mechanical Turk worker with a navigation, delivery, and relocation task drawn from the training set of tasks.

We divide the training procedure into three phases, each associated with 8 objects from the active training set, which can be queried about during conversa-

tions using opportunistic active learning. Between phases, the parser and perception models are retrained from the conversations the agent has had so far. The purpose of dividing training into phases is two-fold. First, periodically retraining the parser can ameliorate users becoming discouraged from lexical variation. Since the parser will know more words and phrases after each retraining step, it will understand more of what users initially say, and they will be less inclined to back off to what they perceive as simpler language when the agent asks clarifying questions. Second, limiting the active training set to 8 objects is consistent with Chapter 5, and limits user fatigue when asked open-ended questions (for positive and negative examples). Retraining also means words the current user hasn't used and that weren't in the initialization phase can then arise during queries. Each phase i is carried out by agent \mathcal{A}_i , after which parser \mathcal{P}_{i+1} and concept predicates $P_{c,i+1}$ are trained to instantiate agent \mathcal{A}_{i+1} .

For each training phase $i \in \{1, 2, 3\}$, we create N Mechanical Turk Human Intelligence Tasks (HITs) for workers to complete, where each HIT includes one of each task type: navigation, delivery, and relocation, drawn from the training set of tasks. Workers who complete all three tasks are considered when training the next parser and perception concept models. Table 8.8 summarizes the Mechanical Turk task and the final numbers of workers who completed various parts. Among workers who completed all tasks, two types of conversational training data are extracted. First, if the worker completes a task correctly, we induce utterance-denotation pairs from that conversation for use in inducing utterance-semantic form pairs to train the parser. Second, regardless of task correctness, if the agent asked the worker any questions about object-predicate relationships using opportunistic active learning, we include the worker's answers to retrain the perceptual concept models. Additionally, if the worker answered questions about whether words were perceptual in nature, and whether perceptual words were synonyms of known words, those answers were considered during retraining the parser and perception modules.

As discussed in Section 8.2.4, we induce utterance-denotation pairs from conversations by examining which semantic role(s) the agent asked for versus what the workers said in response, once the correct semantic roles are confirmed, giving

pairs D_i . Between phases, we train \mathcal{P}_{i+1} with $\bigcup_{j=0}^i D_j$, that is (the initialization pairs D_0 plus the induced pairs from all phases so far).

Before training perceptual concept models across many user conversations, we consider which concepts should be added as novel, and which should be added as synonyms. We consider a word w a candidate perceptual concept if at least one worker said *yes* when asked if it was perceptual in nature and subsequently labeled at least one object as a positive example for w . We then look at all the words v that workers voted on w being a synonym of, and, if a majority of users voted *yes*, then we add w to the lexicon as a word with the same meaning as the underlying perceptual concept for v . These ontological and lexical updates to P_{i+1} happen *before* retraining on D_i .

We aggregate the object-predicate labels obtained during each worker’s conversations as training data for the perceptual concept models. We use the majority vote across workers to add an object-predicate pair as either a positive or negative training example, disregarding object-predicate pairs in the event of a tie (e.g. to handle workers disagreeing about whether a word applies to an object). These additional examples are used to train predicate concept models $P_{c,i+1}$.

After adding new concept models to the lexicon and ontology of P_{i+1} , we are equipped to induce utterance/semantic-form pairs D_i from utterance/denotation pairs. This is because new concept words, and new synonyms, are present in the utterances of D_i . Thus, our training of \mathcal{P}_{i+1} proceeds over a fixed number of epochs E as follows:

First, new utterance/semantic-form pairs $D_{i,e}$ are induced from the utterance/denotation pairs in conversations from phase i using the current parser $\mathcal{P}_{i+1,e}$, where the first epoch ($e = 0$) has new lexical and ontological entries for new concepts and synonyms from phase i . That is, after adding new entries for new perceptual words, we get induce parser training data that might need these new words. Next, $\mathcal{P}_{i+1,e+1}$ is trained by passing once over the pairs $D_{i,e}$ (we train the next epoch parser). Now new pairs $D_{i,e+1}$ are induced using $\mathcal{P}_{i+1,e+1}$, and the process repeats (in our experiments, we set $E = 10$). That is, we induce the latent semantic forms from utterance/denotation pairs and retrain the parser at each epoch. This is be-

cause parser training can affect which latent semantic forms are selected (i.e. as the parser improves, more accurate latent forms can be found). The final trained parser $\mathcal{P}_{i+1,E} = \mathcal{P}_{i+1}$ is used in the next phase of the experiment for agent \mathcal{A}_{i+1} .

After three phases, agent \mathcal{A}_4 with parser \mathcal{P}_4 and perception models $P_{c,4}$ is tested by interacting with users trying to accomplish tasks from the unseen test set of tasks. We also test an alternative agent, \mathcal{A}_4^* , with parser \mathcal{P}_1^* and perception models $P_{c,4}$. This agent serves as an ablation to see the effects of trained perception models versus those along with a trained parsing model. Parser \mathcal{P}_1^* has its lexicon and ontology expanded to include the new predicates and synonyms discovered retraining perception models $P_{c,4}$, but its training iterates only over pairs D_0 , not additional utterance-semantic form pairs induced from conversations.

Performance Metrics

Quantitatively, we measure the number of clarification questions the agent asked per user and task, as well as the *semantic f-score* of each user on each task. The number of clarification questions is the number of questions required before all roles of the command are confirmed by the user. This measure does not include questions about whether words are perceptual, whether a new perceptual concept is a synonym of a known one, or the questions asked during sub-dialogs using opportunistic active learning. We only consider this metric when a worker confirms the correct task, since we are interested in reducing successful conversation lengths.

The semantic *f-score* of a task is a measure of the agreement between the task the user confirmed and the task they were instructed to convey, and is used as a measure of how close users came to conveying the correct task (Corona et al., 2017). This is calculated as the harmonic mean of the precision and recall between the sets T_U and T_G , where T_U is the set of pairs (r, a) of roles and constants the user confirmed, including a pair for (action, a) for a an action, and T_G the gold task specification pairs. For example, if the user confirmed $\text{deliver}(o_2, p_1)$ when the gold

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 8.7: Survey prompts posed at the end of our evaluation sessions with human users. Users answered on a 7-point Likert scale the degree to which they agreed or disagreed with each of these statements.

task specification was $\text{relocate}(o_2, r_1, r_2)$, then:

$$\begin{aligned}
T_U &= \{(\text{action}, \text{deliver}), (\text{patient}, o_2), (\text{recipient}, p_1)\}, \\
T_G &= \{(\text{action}, \text{relocate}), (\text{patient}, o_2), (\text{source}, r_1), (\text{goal}, r_2)\}; \\
\text{precision}(T_U, T_G) &= \frac{|T_U \cap T_G|}{|T_U|} = \frac{1}{3}, \\
\text{recall}(T_U, T_G) &= \frac{|T_U \cap T_G|}{|T_G|} = \frac{1}{4}, \\
f(T_U, T_G) &= 2 \cdot \frac{\text{precision}(T_U, T_G) \cdot \text{recall}(T_U, T_G)}{\text{precision}(T_U, T_G) + \text{recall}(T_U, T_G)} = 0.286.
\end{aligned}$$

We also consider user’s answers to survey questions about their experience with the agent. Each question was answered on a 7-point Likert scale: *Strongly Disagree* (0), *Disagree* (1), *Slightly Disagree* (2), *Neutral* (3), *Slightly Agree* (4), *Agree* (5), *Strongly Agree* (6). Survey questions are given in Table 8.7. In particular, we evaluate whether users consider the agent more understanding and less frustrating, as well as whether they would more consider using the agent for real-world tasks, after training the parsing and perception components from conversations. Users are also able to provide optional, free-form text feedback as a part of completing this survey, and their responses provide some anecdotal insight into their experiences with the agent.

Mechanical Turk Evaluation

We conducted a large-scale evaluation of our procedures for learning parsing and perception from conversations using Amazon Mechanical Turk as a crowdsourcing platform for workers to interact with our dialog agent.

Web Interface

Workers connect to our web interface and engage in three conversations, then fill out a survey about their experience using the agent. Workers are initially instructed *Give your commands all at once, as opposed to in individual steps*. At the beginning of each conversation, workers are presented with a new task, together with the instructions: *Command the robot with a complete sentence. The robot does not understand questions, but it may ask you questions of its own. The robot understands high-level commands, so it doesn't need step-by-step instructions, and it doesn't matter what location it starts in.*

To avoid biasing workers towards certain words (linguistic priming), we present tasks by describing the target state of the world after the task is completed. For example, for the navigation task, the prompt is: *Give the robot a command to solve this problem: The robot should be at the X marked on the green map*. A green-highlighted floor map is shown along with this text, with a marker for the target room (Figure 8.10). For delivery tasks, the problem is phrased *This person needs the object shown below*. The person and object are shown as picture panels of a person icon (as in the right panel of Figure 8.10) and the object item. For relocation tasks, the problem is phrased *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 with markers are shown below the problem.

Confirmation and clarification questions in dialogs are asked by color-coding reference phrases like *this object, this person, here, and there* and showing a corresponding picture with the matching color highlight. Figure 8.11 shows a clarification and confirmation question for the *goal* semantic role of the navigation task.

The agent also asks questions in sub-dialogs regarding whether perceptual

**Give the robot a command to solve this problem:
The robot should be at the X marked on the green map.**

Letters in offices indicate that the corresponding person owns that office.

Person Name

- (B) Robert "Bob" Brown
- (D) David "Dave" Daniel
- (H) Dr. Heidi Hughes
- (M) Mallory "Mal" Maroon
- (N) Dr. Nancy Nagel
- (P) Peggy Parker, Intern
- (R) Richard Rogue, Secretary
- (S) Dr. Sybil Smalt
- (W) Walter Ward, Supervisor

Figure 8.10: Web interface prompt for an example navigation task. The map on the left marks the target room to avoid linguistic priming (e.g. *at the X marked on the green map* can be used instead of *the middle pod*). The panel on the right shows the names of people to establish office ownership relationships (e.g. *the supervisor's office*).

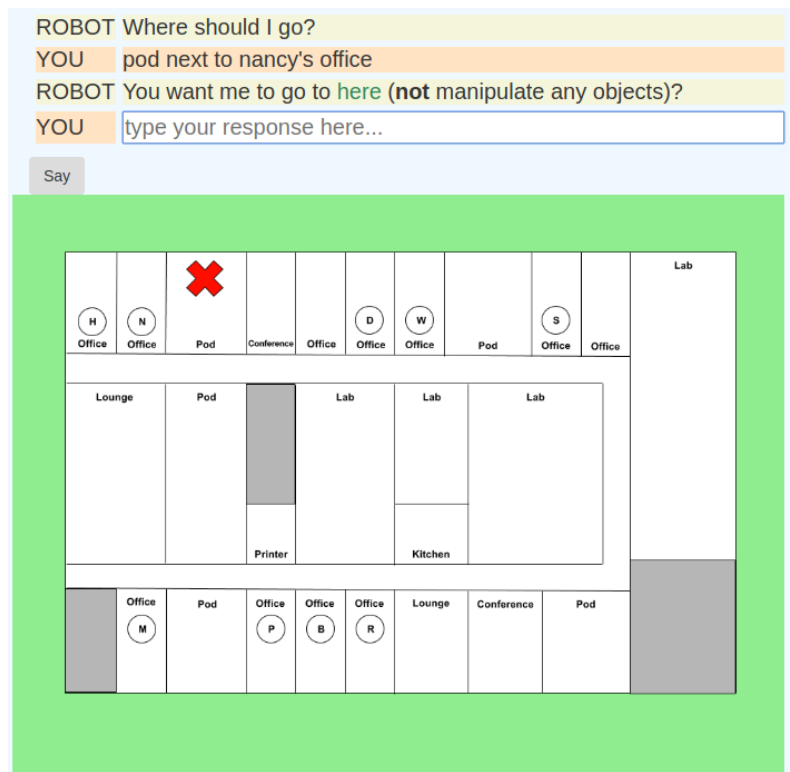


Figure 8.11: Web interface when the agent asks for a semantic role confirmation. The green word *here* refers to the green-bordered map below, the *goal* of the command. In other tasks, like relocation, green- and pink-bordered maps separately refer to the *goal* and *source* semantic roles during such confirmation questions. In this example, the agent asked the clarifying question *Where should I go?* just before the confirmation, eliciting a referential phrase *pod next to nancy's office* that can be tied to the denotation (the referent room) for training the parser.

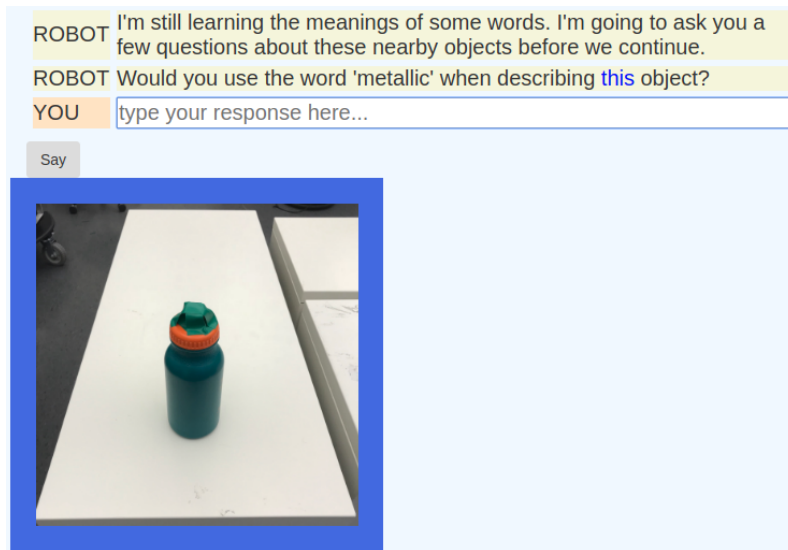


Figure 8.12: Web interface for the agent asking whether a predicate applies to a particular object. The blue word *this* refers to the blue-bordered object image.

concept words apply to objects. Figure 8.12 shows an example of a yes/no question for whether a word applies to an object. In this case, the worker types a response. Figure 8.13 shows the panel used for open-ended positive/negative example questions (asking, among available objects, which one a word does or does not apply to). In this case, workers click one of the objects to select an answer, or the *Shake Head* button to say none of presented objects satisfies the request.

Human Intelligence Task (HIT) Setup and Completion Data

We run 50 HITs at a time on our server. This was based on the maximum number of processes our server could spawn for connecting workers without exceeding the its available RAM. For the train condition, we run two batches per fold of active training set objects, for a total of $2 \times 3 \times 50 = 300$ workers. 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 run three batches of 50 workers each for a total of $3 \times 50 = 150$ workers.

Each worker completes three tasks: navigation, delivery, and relocation, in



Figure 8.13: Web interface for the agent asking for a positive/negative example of a particular 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?*

that order, drawn from either the train or test set of tasks, then completes a survey as described above. The conversational dialog agent has access to word embeddings, which it uses to preemptively expand its per-worker lexicon with induced entries for out-of-vocabulary words (Section 8.2.1). These entries are not visible when the parser is retrained, but correct ones can be induced through the procedure for learning from denotations.

Workers were paid \$1 per completed HIT. Each HIT is available to submit for 2 hours after starting. The dialog-agent server enforces a time limit of 1 hour per worker. If a worker’s conversations exceed 1 hour, the agent times out, allowing the worker to advance to payment. This is to assuage workers who believe the HIT takes far too long, though it may discard workers who are doing multiple HITs in parallel and only spending a little of the hour on ours. If, in an individual task conversation, a worker replies to the agent (either by writing commands, selecting objects, or writing clarifications) 30 or more times, a button appears on the interface allowing them to end the HIT and advance to payment. This also prevent workers from feeling exploited by the long, difficult nature of the HIT. The same option is offered to workers if the agent takes more than two minutes to respond (this delay only happened when the agent crashed from RAM being exceeded on the server managing the agents for workers). These allowances help maintain our Mechanical Turk recommender account’s usability and trustworthiness with workers, but reduce the total number of complete interactions our agent has with humans.

Finally, we *vet* the remaining set of workers by removing two categories. First, for workers who complete HITs more than once in our experiment, we consider their conversational data only from the *first* time they did one of our HITs. Second, we remove workers who confirmed *navigation* commands with the agent for all three target tasks (navigation, delivery, and relocation). On inspection, these workers identify, during the navigation task (the first dialog), that the robot will advance to the next phase once a command is confirmed, and they continue issuing navigation commands because these are fast to resolve.

Table 8.8 gives a breakdown of the numbers of workers who engage with our HITs through different experimental conditions. Only workers who submit the

Condition	Number of Workers					
	Submitted HIT	Completed Tasks	Vetted	Nav. Correct	Del. Correct	Rel. Correct
Train (A_1, A_2, A_3)	297	162	113	36	44	18
Untrained (A_1)	150	67	44	17	22	10
Test* (A_4^*)	148	83	50	20	29	10
Test (A_4)	143	79	42	16	23	10

Table 8.8: Breakdown of the number of workers in our experiment. We here count only workers that **submitted** the HIT with the correct code, meaning they completed the HIT but may not have finished all three tasks due to timeout or overly long conversations. Workers that **completed** all tasks and the survey finished the HIT entirely. **Vetted** workers are ones that were both first-time participants and did not confirm only navigation commands for all tasks. Finally, we note the number of vetted workers that confirmed the correct task and whose conversations were used to train the parser (at train time) or evaluate conversation length (at test time). The Train condition (A_1, A_2, A_3 agents) draws from the training set of tasks, while the Untrained (A_1 untrained agent), Test* (A_4^* agent with trained perception and untrained parser), and Test (A_4 agent with trained parser and perception) conditions draw from the test set of tasks.

HIT with the correct survey code (i.e. actually use the interface) are considered for training the system (for the train condition) and evaluation (for the test condition). For training the parser and for evaluating conversation lengths, conversations are only included when the worker confirms the correct task.

The low number of workers that complete the tasks given that they submitted the HIT at all gives a sense of how difficult the HIT is compared to others on Mechanical Turk. The number of vetted workers is even lower, due in large part to repeat users who enjoyed the task enough to come back and do it again (even though this was explicitly discouraged in the HIT description), by contrast.

Quantitative Performance Results

Figures 8.14 and 8.15 give quantitative measures of the agent’s performance in the untrained condition (A_1), the trained condition where *only* the perception

modules are updated based on user conversations (\mathcal{A}_4^*), and the trained condition where *both* the parsing and perception modules are updated based on user conversations (\mathcal{A}_4). We measure the number of clarification questions answered by users (e.g. omitting those about word synonyms and those regarding objects’ relationship to perceptual concepts) and semantic slot f -measure (e.g. the agreement between the semantic roles in the target task versus the task confirmed by the user).

The number of clarification questions per successful dialog (one that ends with the user confirming the correct task) is hypothesized to decrease as more conversations become available for training the parsing and perception components of the agent. Because this metric requires users to communicate correct tasks, the number of users we can average over to estimate the agent’s performance is low, resulting in high standard deviations.

For *navigation* (Figure 8.14a), only the action and the goal room need to be communicated, so the number of clarification questions is typically low. Only parser training can affect this task, since perception on real-world objects is not required. We find that training the parsing model from conversations doesn’t change performance, possibly because the number of arguments involved is small. This is similar to the result demonstrated in Chapter 3 for navigation tasks.

For *delivery* (Figure 8.14b), the action, real world object to be delivered, and the recipient person need to be communicated. Training both parsing and perception makes a dent in conversation length, but, counter-intuitively, when training only perception the conversations become a little longer. This may be a result of the untrained parser incorrectly translating utterances involving many new predicates (which have lexical entries as per the perception training scheme, but have not appeared in paired $(x, r) \in D$ parser training instances for updated θ parser parameters). That is, without parser training, out-of-vocabulary words can be mapped to the incorrect perceptual concept models (for example, function words like *should* mapping to the concept *can* because of the function sense of *can* placing them closer together in word embedding space). It is important not only to refine perception modules, but to give the parser examples of how those models are invoked from user utterances.

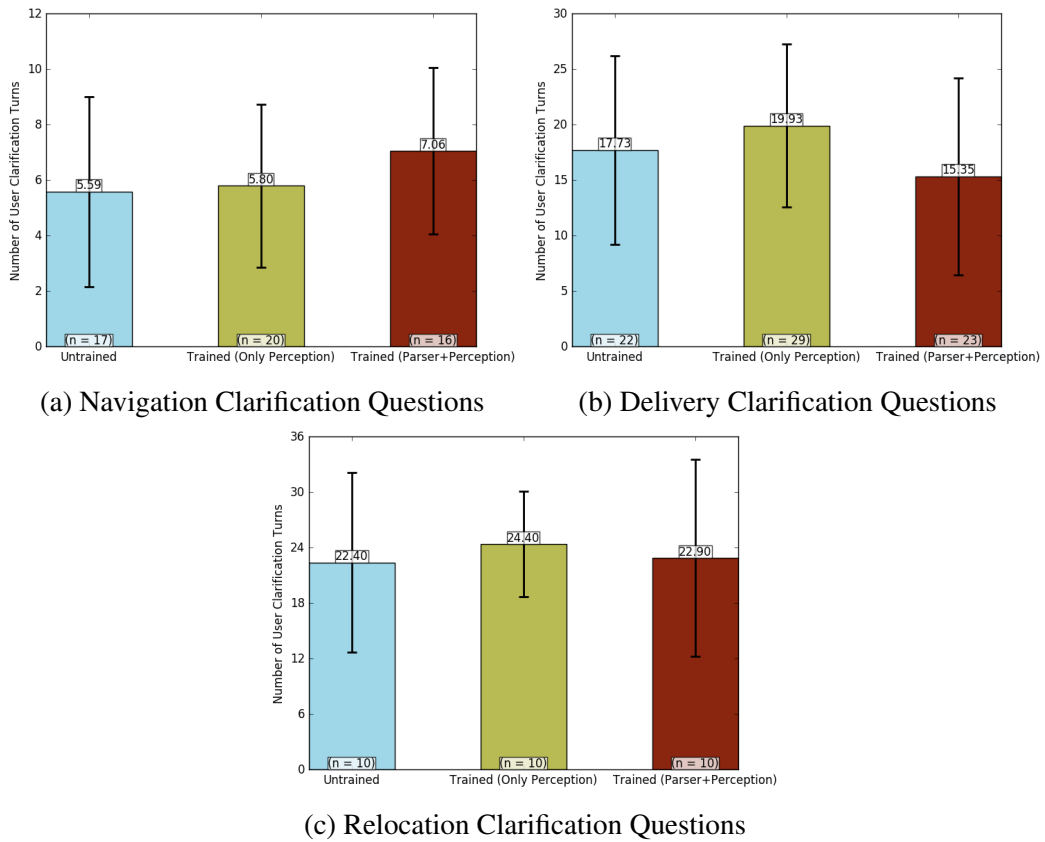


Figure 8.14: The average number of clarification questions the agent asked users (lower is better). In this and other figures, plots display the number of users n in parentheses at the bottom of each bar representing the average over an experimental condition. Additionally, the numerical averages represented by the bars are given in text on top of each condition bar. The black lines in each condition bar represent the standard deviation.

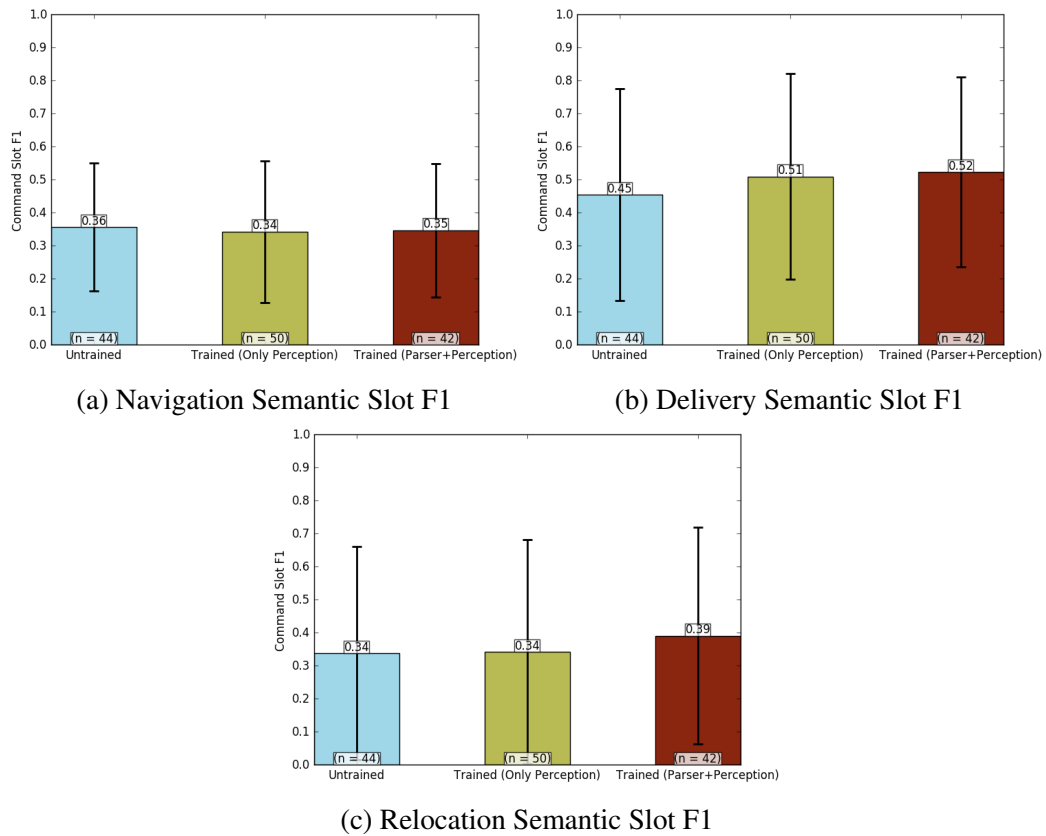


Figure 8.15: The average semantic slot f scores between the semantic roles in the target task and the task confirmed by the user (higher is better).

For *relocation* (Figure 8.14c), the action, real world object to be relocated, and the source and goal locations need to be communicated. Due at least in part to the low number of users who finished this task correctly across conditions, it is difficult to discern the degree to which training shortened the number of clarification questions. Training on this small data was not enough to substantially shorten conversations. By contrast, when considering user f scores with target tasks, we can see that training gives a bump in performance on this task.

The f scores between the tasks users confirmed with the robot and the gold tasks they were assigned provides us insight into how far users were willing to go to communicate the correct intent. These scores reflect both user frustration (e.g. giv-

ing up early with a wrong task specification) and the agent’s misunderstandings, adding noise and contributing to the modesty of the increases in performance after training. The advantage of this score is that we can consider all users who completed the tasks, regardless of their correctness. We hypothesize that the f score will be higher when the agent is operating with trained parsing and perception modules.

For *navigation* (Figure 8.15a), the f score remains almost constant before and after training, regardless of which models are trained. This could be largely due to the low number of semantic roles (2) involved.

For *delivery* (Figure 8.15b), the f score increases most when we retrain the perception module (likely because the *patient* argument, the physical object, is easier to select). Adding parser training gives a further bump in performance.

Finally, for *relocation* (Figure 8.15b), the f scores only increase when we retrain both the parsing and perception modules. This is consistent with the difficulty of the task: two locations on the map, referring to which becomes easier with a better parser, and an object in the real world, referring to which becomes easier with better perception modules.

User Survey Results

Figure 8.16 gives qualitative measures of the users’ experience with the agents. Figure 8.17 gives user responses to prompts about whether they would use the robot to accomplish tasks in the real world similar to the three they just commanded it to perform. As described before, these prompts answered along a 7-point Likert scale ranging from *Strongly Disagree* (0) to *Strongly Agree* (6).

For experience metrics, we hypothesize that users will find the system more understanding and less frustrating when trained versus untrained, with training both parsing and perception better than perception alone. Regarding whether the user felt the robot understood them, in general (Figure 8.16a), there is little change between the untrained and trained conditions. This may be reflective of whether users felt frustrated by the robot (Figure 8.16b), where we see a similar lack of change but consistently higher agreement. That is, across conditions, Mechanical Turkers were

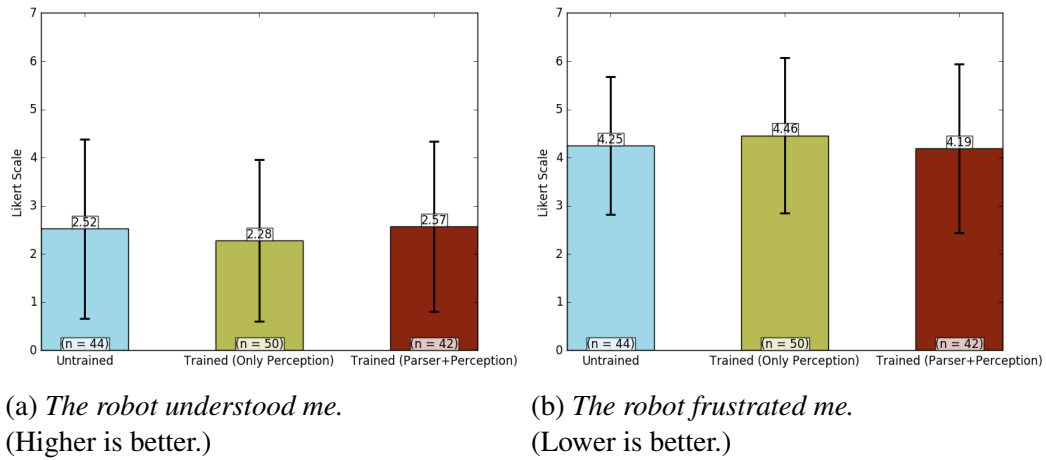
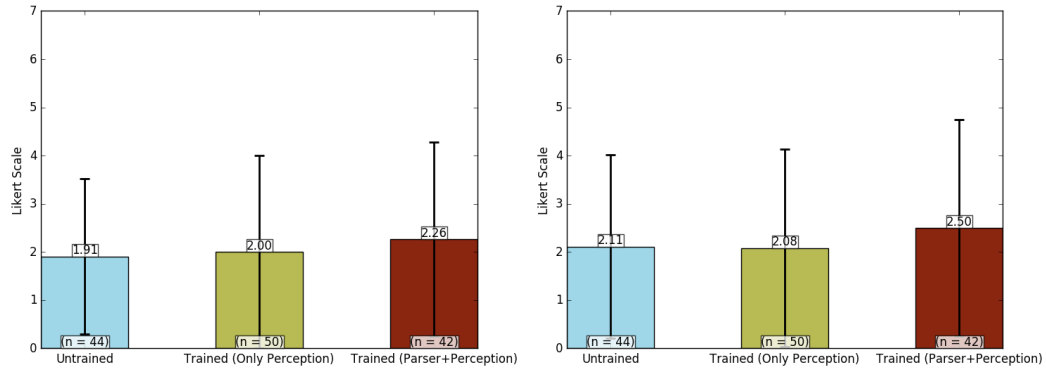


Figure 8.16: Survey prompt responses about task experience.

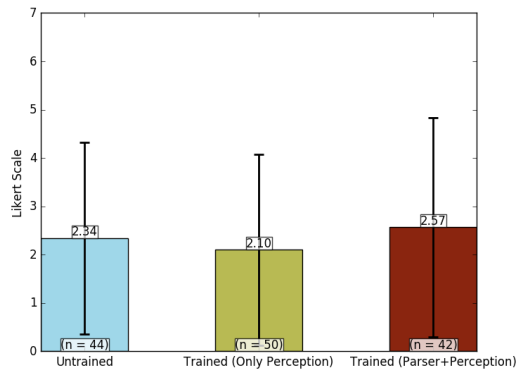
frustrated by the task (this is unsurprising, given its length, the degree of conversational involvement and consequent time, and the reward of only \$1), regardless of the agent’s quantitative performance.

The users’ qualitative feelings about **usability** suggest whether a deployed agent in a physical robot might actually be utilized by humans. We hypothesize that users will rate the system more usable for each task after training the parsing and perception modules. Across all three tasks, we see a slight increases in user ratings of usability between the untrained condition and the trained parsing and perception module condition. Notably, for every task, training the perception system alone is insufficient to increase usability ratings. For *navigation* (Figure 8.17a), this result is consistent with this task not requiring perception. For *delivery* (Figure 8.17b) and *relocation* (Figure 8.17c), this is more surprising, since perception is required for these tasks. The improved parser may affect users’ perception of the agent as a whole, regardless of its performance on individual tasks, making the ratings users give to the usability of these three tasks co-dependent.

Though we discard users who take HITs more than once (in different batches) during our vetting process, we can still consider their subjective experience. In particular, we can track their responses on the survey’s open response text box as they finish HITs in different conditions. For example, one user participated in HITs



(a) *I would use a robot like this to help navigate a new building.* (b) *I would use a robot like this to get items for myself or others.*



(c) *I would use a robot like this to move items from place to place.*

Figure 8.17: Survey prompt responses about usability (higher is better).

across two learning phases and then one testing condition, experiencing the agent as it acquired more and more robust parsing and perception components. The first time that user completed a HIT, it was with agent \mathcal{A}_2 (one phase of training), and the user wrote: *Ugh. I can never figure out how to get it to understand that red and white container with the snap lid! It always goes for the soda can instead. Argh. ...* The second time, with agent \mathcal{A}_3 , the user wrote: *A good day for Mr.Robot. It's nice to have progress...* Finally, with agent \mathcal{A}_4 , used with test set tasks, the user wrote: *Wow. It's made some progress. It was a lot easier to parse this time...*

Learned Perceptual Concept Models

The agent acquires new perceptual concept models (25 in total), and synonym words for existing concepts, during the three phases of training. The learned concept models are noisy, given that Mechanical Turk workers are often inattentive or even combative in the long HIT. For example, nine workers labeled a uniformly yellow mustard container as a positive example for *red*. Nonetheless, these learned models quantitatively and qualitatively improve user experience with the agent. Table 8.18 shows the distribution of confidence given by the learned perceptual concept model for *can* during testing tasks. The soda can object takes more probability than the other objects, while the foam block receives no probability mass.

When examining concept models through cross-validation on the training objects, the concept models for *blue*, *brown*, *container*, *cylinder*, *long*, *red*, and *small* achieve an average $\kappa = 0.63$ with the majority votes from human annotators. These are the models for which, when evaluating in cross-validation, more than the majority class is returned (most predicates have a strong majority class baseline that the concept models learn or fall back on, depending on available labels). The agreement is fairly high for these color and shape concepts, several of which have multi-modal interpretations (e.g. *small* can be evaluated based on object weight and height, as well as visually). The average agreement with human annotators for all 25 learned concept models (including those for which the majority class is always returned) is even higher at $\kappa = 0.79$.

Learned Perceptual Concept Model Distribution Results for *can*

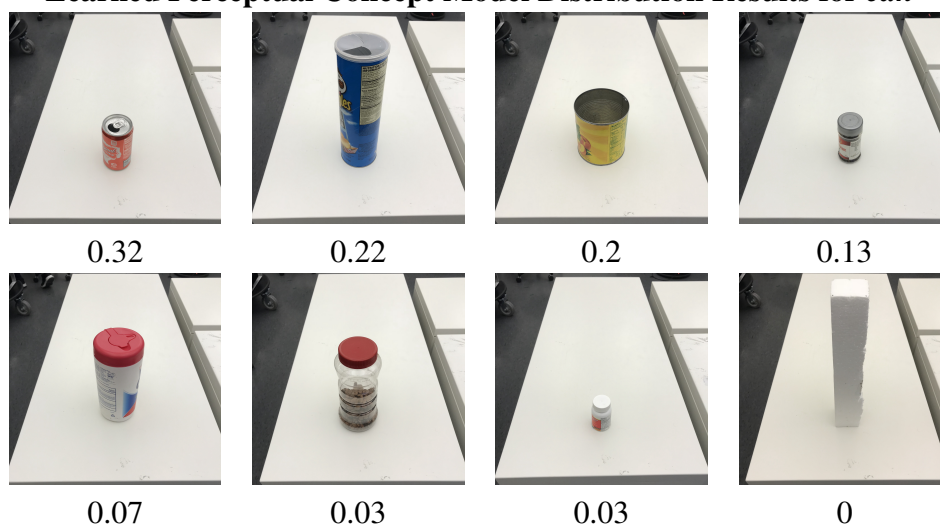


Figure 8.18: The perceptual concept model learned for *can* after training from conversations with human users. The numbers below each object in the active test set represent their (rounded) portion of the distribution of probability based on outputs from the *can* concept model. The objects are roughly ordered by *can*-ness after training, with the soda can object ranking highest.

Physical Robot Implementation

For our embodied demonstration, we use the BWIBot (Khandelwal et al., 2014, 2017), equipped with a Kinova MICO arm (for pointing to and manipulating objects), an Xtion ASUS Pro camera (for viewing objects on tabletop surfaces), a Hokuyo lidar (for navigation and obstacle avoidance), a Blue Snowball microphone, and a speaker (for conducting the dialog). An Alienware computer executes all necessary computation. The robot’s mobile Segway base is reinforced with two additional 12V Li-Ion batteries to power the base, arm, computer, and sensors for up to 6 hours of continuous operation.

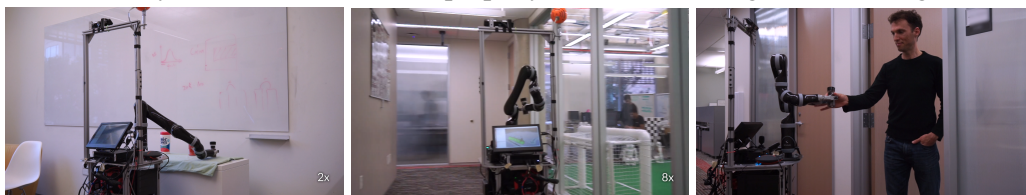
We implement the dialogue interaction atop of the BWIBot software stack (Khandelwal et al., 2017)— which provides automated task planning and autonomous navigation. For this demonstration, speech transcripts are provided by the Google Speech API (<https://cloud.google.com/speech/>) and speech synthesis is performed with the Festival Speech Synthesis System (<http://www.cstr.ed.ac.uk/projects/festival/>). Tabletop perception, required for both the dialog interaction and the execution of the resulting command, is implemented with plane fitting and Euclidean clustering. The robot approaches 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 is realized with a heuristic grasping approach. Candidate grasps are 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 centroid of the object. The robot executes handovers by presenting the item and waiting for its joint efforts to exceed a threshold. The symbolic planning domain here is specific to navigation actions, such as going to a location and opening a door, and plan execution is handled by the existing BWIBots system.

In the demonstration, depicted in Figure 8.19, the dialog begins near a set of objects in a room that can be subjects of opportunistic active learning questions (https://youtu.be/PbOfteZ_CJc). A human user instructs the robot to *Move a rattling container from the lounge by the conference room to Bob’s office.*

- 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 user gives the command verbally. (b) The robot learns the new *rattling* property. (c) The robot autonomously navigates to the target.



(d) The robot decides the relevant item and grasps it. (e) The robot autonomously navigates to the destination. (f) The robot hands over the item.

Figure 8.19: A demonstration of the learning agent embodied on the physical BWI-Bot platform.

The dialog agent has been initialized using predicates learned from Amazon Mechanical Turk interactions, but does not know the word *rattling*. The agent checks its word embedding to see if it can relate the word to a known predicate. After the human rejects the agent’s synonymy and antonymy inquiries, it queries the user for positive and negative examples of rattling items from the nearby objects. After this exchange, the robot confirms that the user intended to issue an object relocation command and that it grounded the destination location correctly. Leveraging its previous exploration of the queried objects, the agent trains a new perceptual classifier for the rattling predicate and correctly identifies the requested item from among several candidates in the target room.

The grounded command type, object, source room, and target room are passed to task level control. The task is decomposed into three parts: navigation to the source room, retrieval of the object, navigation to the target room for a hand-over. In this demonstration, navigation to the source room is 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 perceives its surroundings and executes a table approach and object grasp step. Once the robot has the object in its gripper, it makes a navigation plan with the goal of facing the target room’s door. Once at the target door, the robot hands over the object to *Bob*, concluding the execution of the task.

Conclusion

In this chapter, we presented a holistic system for jointly improving semantic parsing and grounded perception on a robotic system for interpreting natural-language commands through human-robot dialog. This agent uses clarification questions during dialog to refine its understanding of user intent, then aligns answers to those questions to underlying denotations to improve its semantic parser after inferring the latent semantic meaning connecting the answers to their denotations. Further, the agent engages users with questions about local objects and their physical properties, with questions selected through an opportunistic active

learning strategy that improves performance on the current command (relevant perceptual words) and downstream tasks (off-topic words).

We show, via a large-scale Mechanical Turk experiment, that users are better able to communicate tasks to the agent after this learning procedure acquires new information from conversations with other users. Further, we find that users rate the overall system more usable for real-world tasks after conversation-based learning. Finally, we embody this learning agent, initialized from Mechanical Turk conversations, in a physical robot platform, and demonstrate its learning abilities for the non-visual word *rattling* unseen during training on a novel test command.

Given this learning agent that combines core ideas from this thesis, there are immediate future directions for folding in more ideas presented in the preceding chapters into a complete system. In the next chapter, we discuss these immediate possibilities as well as broader future research applications for human-robot dialog systems that learn in a continual way from human users.

Chapter 9

Future Work

This thesis broadly addresses topics at the intersection of natural language processing, robotics, vision, and dialog. Below, we overview future directions for integrating ideas presented in this thesis to build a grounded dialog agent that can improve its component parts through dialogs with human users. We then discuss broader future directions for applications of human-robot dialog to accomplish other high-level goals outside of command understanding.

Improving Language Understanding and Perception Modules

The preceding chapter discusses our holistic dialog agent that performs semantic parser learning from conversations and engages the user in perception-relevant sub-dialogs to improve its concept models. There are several possible improvements on such an agent. Some of these directions are outside the scope of this thesis, while others are natural integrations of the methods presented here.

Improving Black-box Speech Recognition through Parsing and Perception

Off-the-shelf speech recognition systems can have high word-error rates unless trained extensively on individual user voices. Even small word-error rates can render utterances unparseable. Existing work the author contributed to investigates using a semantic parser and in-domain language model to re-rank hypothesis transcriptions of audio utterances given to a black-box speech recognition system (Corona et al., 2017).

In that work, transcription hypotheses are re-ranked based on their felicity according to an in-domain language model and a trained semantic parser. The intuition behind this method is that if a transcription contains out-of-vocabulary words or syntactic constructions the semantic parser doesn't recognize, that transcription

is less likely to be correct in the domain. By using a state-of-the-art black-box speech recognition system, high transcription accuracy is possible even without models of individual user speech (in that work, we use the Google Speech API). By adding in-domain constraints through parsing, that transcription accuracy is improved.

Further work in this direction could re-rank transcriptions not just considering semantic information, but also local perceptual information. For example, if a person says *bring me the short bottle* and the top transcriptions include both that and *bring me the sport bottle*, grounding the resulting parses against the local objects could help clarify that *short bottle* is more likely, if an object matching that description exists in the environment (assuming there is not, in fact, an athletics sport bottle on the table as well). The grounding algorithm for working with both perception and knowledge base information, presented in Chapter 8, provides a starting point for this direction of future work, and already uses the Google Speech API for speech transcription on the robot platform.

Leveraging Accommodation

Most natural language understanding algorithms focus on understanding what humans say. However, the overarching goal of NLU is to communicate effectively with a human partner. Effective communication can also arise from more interpretable input utterances from the human partner. In particular, if a user adapts her speech so that the system can better understand it, effective communication is still achieved. In a vanilla dialog system, the user has no way of knowing what words or syntactic constructions the system understands best.

Accommodation is a conversational phenomenon in which interlocutors converge to shared referring expressions, lexical and syntactic choices, cadence, volume, and other vocal variations (Lakin et al., 2003; Gravano et al., 2015; Lubold et al., 2015). Past work by the author explored the connection between user accommodation and learning in a tutoring dialog system (Thomason et al., 2013).

Future work could leverage accommodation to improve natural language un-

derstanding. When responding to a human user, rather than using template-based conversational responses, a dialog agent could consider a range of possible utterances and rank them based on how well the semantic parser understands them. Such an approach is conceptually similar to previous work in which a neural sequence-to-sequence model was used to generate SQL queries from user utterances, and incorrect queries were flagged by skilled users familiar with SQL for further annotation by domain experts (Iyer et al., 2017). The proposed strategy could replace skilled workers with unskilled users by creating an additional model that translates the logical form (in that case, SQL) back into language and offering the paraphrase as a signal for whether the answer is correct.

Previous work has used a similar strategy to influence lexical choices (Lopes et al., 2013). Through accommodation, we can expect human users to adopt the lexical and syntactic choices of the dialog agent as the conversation proceeds. By selecting responses that the semantic parser understands, the system tacitly encourages the user to make lexical choices the parser better understands.

Learning Dialog Policy

The dialog agents that employ clarification and opportunistic active learning, presented in Chapters 3, 5, and 8 use, hand-coded dialog policies to choose questions to ask the user. These policies discretize continuous belief states regarding semantic role slots and confidences about whether perceptual predicates apply to objects, then map these discrete states to questions.

A policy based on a Partially Observable Markov Decision Process-based policy (Young et al., 2013) that considers a continuous belief state about the user’s intention was implemented and evaluated by recent work the author contributed to (Padmakumar et al., 2017). That work demonstrated that a learned POMDP-based policy led to shorter clarification dialogs with users in a setting similar to that presented in Chapter 3, where only semantic parsing was considered (i.e. no perception). Additionally, that work compared two strategies for jointly retraining both the dialog policy and the underlying semantic parser, and established that retraining

them jointly in batches was more effective than holistic retraining, since changes to the parser affect the structure of a good dialog policy.

Further work in this direction could incorporate perceptual grounding, as was done in Chapter 8. In this case, the partial observability of the world would include not only the latent intentions of the user, but the applicability of perceptual concepts to objects in the environment. Incremental retraining would need to consider changes not only to the underlying parser that affects the dialog policy, but additionally to changes in the underlying perceptual concept models as training data becomes available through conversations with users.

Synset Induction for Multi-modal Grounded Predicates

In our experiments with multi-modal, grounded linguistic semantics (Chapters 4 and 5), we discovered that people use some polysemous words (e.g. *light*) as well as effectively synonymous words (e.g. *round* and *cylindrical*) when describing objects. By applying a synset induction algorithm based on the one presented in Chapter 6 to multi-modal predicates, we could tease apart polysemous word senses and strengthen perceptual classifiers by combining synonymous predicates' data.

This kind of learning would be helpful in a deployed system partly because it could learn domain-specific polysemy and synonymy based on data. For example, in the office domain, the command *Fetch me a pen* may mean a robot should bring a writing pen or that it should bring a whiteboard pen. The polysemy step should be able to separate these domain senses of *pen*. The synonymy step should subsequently merge the whiteboard-relevant sense with a sense of *marker*.

In the synset induction algorithm presented in Chapter 6, every observation is associated with a unique label. Each pair of an image and the text of webpages it appeared on was associated with a single noun phrase, such as *kiwi* or *Chinese grapefruit*. However, in robot perception, an observation can be associated with many labels. For example, the same object may be described as *blue*, *cylindrical*, and *bottle*. A baseline for performing synset induction on this multi-modal, multi-label data using the existing algorithms in this thesis is to simply duplicate each

object as an instance for every label. However, because perceptual contexts offer more than the two modalities in Chapter 6 (vision and language), there are more interesting ways to frame the problem. For example, considering the intuitions about context relevance guiding behaviors in Chapter 7, one might partition a polysemous word like *light* into senses by observing that the *lift* behavior is discriminatory for one subset of *light* objects while the *look* behavior is discriminatory for the complementary set.

For synonymy, another simple starting approach is to greedily merge concept words as synonymous if a classifier trained with labels from both words outperformed each individually. Known label overlap and lexical embeddings could be used to guide the search for plausible synonyms. For polysemy, a comparable simple baseline would be to perform 2-means clustering within each concept's positive labels in object space, examining whether the resulting cluster divergence exceeded some threshold beyond which multiple word senses were plausible. This approach would be computationally expensive but could be done along with parser training as an *overnight* task.

Classifier and Feature Extraction Approaches for Perception

The perception modules in this thesis are implemented as linear combinations of decisions from support vector machines (SVMs) operating in each perceptual context. SVMs draw a decision boundary in a feature space given labeled positive and negative examples.

In the future, we could instead use positive-unlabeled classifier methods (Liu et al., 2003; Elkan and Noto, 2008), removing the need for explicit negative examples. Some of these methods include continual learning similar to that presented in our work, like the positive labels an agent receives over time from conversational feedback (Chang et al., 2016). Other methods bootstrap attribute classification in deep models using co-training between multiple representations of objects, starting with only positive examples (e.g. sets of words that describe objects) (Cheng et al., 2016).

Another next step would be to leverage neural feature representations during perceptual grounding. We currently use the penultimate layer of the VGG (Simonyan and Zisserman, 2014) network as a sensorimotor context space for *looking* at objects. We could similarly use auto-encoders over object representations (Burchfiel and Konidaris, 2017) to provide a reduced feature vector representing a *found*, rather than hand-crafted, feature space at every sensorimotor context. For example, the features learned through an encoding for the matrix of haptic motor feedback during the lifting behavior through time may prove more useful than the temporal-binning done presently as a feature extraction method.

Feature extraction for most language grounding work operates at the whole-object level. However, in our experiments (e.g. in Chapter 5), users describe components of objects as well, such as *a container with a red cap*. Object sub-segmentation into components, such as the *cap* of a container or the *handle* of a mug, would enable feature extraction at the whole-object as well as component level. In this case, some perceptual predicates, like *cap* and *handle*, could identify components, and adjectives operating on those nouns in the description (e.g. *red*) could be grounded against the relevant component’s features.

Guided Open-world Object Exploration

Throughout this thesis, language grounding has been performed against a fixed knowledge base (Chapter 3), explored objects (Chapters 4 and 5), or both (Chapter 8). However, in a deployed setting, a robot will need to explore new objects in the environment, as simulated in Chapter 7 by revealing object observations only when the simulated agent chose to explore them when learning a new perceptual concept word.

One direction for future work on an embodied platform is to perform exploratory behaviors in an autonomous fashion. To reduce exploration costs, methods like those explored in Chapter 7 could be employed. Moreover, for applications like the demonstration in Chapter 8 of the robot learning the word *rattling*, a robot would need to identify novel objects in the source room and explore them solely

with the discriminatory behavior estimated for *rattling* from examples gathered using opportunistic active learning with the human. In this case, that could mean picking up and dropping each object to get discriminative audio feedback signal for the *rattling* property.

Learning Across Robot Platforms

As more robots are deployed in workplaces, disjoint corpora of human-robot conversations will become available. There may be room for leveraging transfer learning between similarly-deployed robots (for example, in different hospitals) to increase the amount of human-robot language data available. This could enable learning more data-hungry, but less brittle, neural parsing methods (Dong and Lapata, 2016). Similarly, as different robots interact with different sets of objects, learned manifold alignments between object feature representations from different platforms (for example, different robot arms and cameras) may allow transfer of object experiences *between* distinct robot platforms (Tuia and Camps-Valls, 2016).

Human-Robot Dialog in Other Applications

This thesis has focused on human-robot dialog for command understanding. We have presented algorithmic solutions that leverage the presence of a human interlocutor. Human-robot teams, in general, are capable of accomplishing more than a robot can alone. Below, we outline directions for future work on tasks outside of commanding a robot, for which the presence of a human interlocutor both changes the space of the problem and offers novel ways to solve it.

Learning from Demonstrations with Dialog

When learning from demonstrations, systems are trained to recreate actions after observing a human perform them. For example, after watching videos of people making sandwiches, a robot should be able to make a sandwich. At a high level, and given recent advances in both NLP and robotics, future research could incor-

porate conversation, where humans and robots collaborate on learning a task. This conversational learning would benefit from incorporating common ground— modeling both the robot’s understanding and expected human understanding of actions and questions posed (Knepper et al., 2017). In this setting, robots must be able to: perceive the scene and actions the human is taking, understand natural language from the human and connect it to that scene, formulate natural language questions to clarify understanding, and perform actions incrementally while describing them during question asking.

A step towards achieving these is a change from discrete to continuous representations of knowledge and actions. High-level planning and traditional language understanding can share a common ontology. For example, *grasp*, *knife*, and *jelly* can be composed to express counter-top actions. These discrete symbols limit the behaviors a robot can perform and the perceptual concepts it can reason with. By contrast, continuous representations, such as word embeddings, can give locality information about a wide vocabulary of concepts. Embedding ontological symbols in a continuous space will allow a robot to share information between similar symbols while preserving differences in their execution—for example, knowing that *hold* and *grasp* involve the same end position for a hand, while the latter requires closing the fingers first. Work towards this end has begun for example for zero-shot recognition of previously unseen human gestures by embedding gestures in a continuous space (Thomason and Knepper, 2016).

These continuous representations can be bootstrapped from large, unstructured text and tuned incrementally during language-augmented learning from demonstration. This change could be a first step towards a conversational training paradigm for learning from demonstrations where natural language understanding and generation are integral. In this setting, where perception and planning to recreate actions are integral, language understanding may take the form of signals that influence plan structure. Language generation, for communicating back to the human, then becomes an option during planning to resolve misunderstandings about how to complete a demonstrated task (refining both perception- and linguistic-level ambiguity).

Explainable AI with Dialogs

Enabling robots to explain their decisions will help humans understand why those decisions were made, and enable targeted feedback to modify these decisions in the future. Explainable artificial intelligence is an area of emerging interest in the wake of black-box deep learning methods that achieve state-of-the-art, though opaque, performance in computer vision tasks. Advances in neural language models, for example, allow co-generating model decisions and explanations (Hendricks et al., 2016). At a high level, bringing explainability into robotics creates challenges and unique opportunities versus explaining decisions in vision applications.

If a robot gets the wrong mug from the kitchen, it should be able to offer why (maybe it falsely believes the mug to be green), allowing the human user to guard against future, similar mistakes by correcting it (*Actually, that mug is purple*). However, in order to use deep models at all, data sparsity issues (limited sentence/semantic form pairs for language understanding, exploratory behaviors on objects for concept recognition, etc.) must be addressed, and adding model interpretability will exacerbate this need. Encoding intuitions from traditional semantic parsing, such as the tree structure of semantic forms, can reduce the data required to train deep models (Dong and Lapata, 2016). Additionally, perturbing available input information, such as sentences (Alvarez-Melis and Jaakkola, 2017), and objects (possibly based on low-level object representations like those induced by (Burchfiel and Konidaris, 2017)) could expand the available training data analogous to the perturbation strategies used in deep visual recognition (Krizhevsky et al., 2012).

In a grounded setting, a robot perceiving a cluttered table could establish a prior when understanding a command from a human in the same environment (i.e. the human is probably talking about something in the nearby clutter). How to incorporate these structural and perceptual priors together when training explainable language understanding models for robots is an open area for further research.

Chapter 10

Conclusions

The work in this thesis is situated at the intersection of natural language processing, robotics, vision, and dialog. We draw attention to the importance, and difficulty, of robust command understanding, and present methods for using human-robot conversation as a learning signal for this task. To summarize each contribution of this thesis briefly:

1. We describe a method for using clarification dialog to both interpret the current command a human is trying to communicate and learn to improve future command understanding (Chapter 3). We show that this learning can reduce conversation lengths with users for multi-argument commands.
2. We demonstrate the effectiveness of language grounding in multi-modal space, connecting words like *heavy* and *empty* to physical properties of objects, going beyond traditional language grounding work in visual space alone (Chapter 4). We introduce a human-robot *I Spy* game, through which a robot gets positive and negative labels between words and physical objects from human players.
3. We introduce an opportunistic active learning strategy for a robot to decide on maximally-useful questions about objects and their word labels to pose to a human interlocutor during an object retrieval task (Chapter 5). We find that humans are undeterred by a robot agent that asks them off-topic questions about words they themselves did not use to describe objects.
4. We introduce a multimodal, unsupervised method for discovering word senses (detecting polysemy) and subsequently clustering those senses (detecting synonymy) to form synonymous sets of senses (Chapter 6). We find that these unsupervised synsets quantitatively match supervised versions and are qualitatively rated as more sensible by humans when the clustering methods operate in a joint textual and visual space.

5. We also introduce a strategy for guiding a robot’s exploratory behavior policy, reducing the number of behaviors (and the time taken) to explore new objects when acquiring a concept model for a novel word (Chapter 7). We find that large-scale word embeddings and human annotations of relevant behaviors improve concept model accuracy when a robot is restricted to a fixed number of exploratory behaviors or a time budget for those behaviors.
6. Finally, we join two themes of the thesis into a grounded dialog agent that improves both its semantic parsing and lexical grounding components jointly from conversations with humans (Chapter 8). This joint learning allows the agent to understand more complex commands, acquire new syntactic structures and words, and improve concept models of object properties over time.

These contributions begin to address the low-resource nature of many problems in human-robot dialog, and grounded language understanding in general, by introducing algorithms and strategies for data collection that can be deployed with low initial supervision in a new domain. We discussed both immediate directions for further work in command understanding and orthogonal directions for human-robot collaboration leveraging dialog for continual learning (Chapter 9). Many more unexplored possibilities are forthcoming as robots inhabit more human environments, and we look forward to increased community focus on human-robot dialog as researchers continue making progress in this cross-disciplinary area.

References

- E. Bruni A. Lazaridou and M. Baroni. Is this a wampimuk? cross-modal mapping between distributional semantics and the visual world. In *Proceedings of ACL 2014 (52nd Annual Meeting of the Association for Computational Linguistics)*, pages 1403–1414, 2014.
- Aitor Aldoma, Zoltan-Csaba Marton, Federico Tombari, Walter Wohlking, Christian Potthast, Bernhard Zeisl, Radu Bogdan Rusu, Suat Gedikli, and Markus Vincze. Point cloud library. *IEEE Robotics & Automation Magazine*, 1070(9932/12), 2012.
- Muhannad Alomari, Paul Duckworth, Nils Bore, Majd Hawasly, David C. Hogg, and Anthony G. Cohn. Grounding of human environments and activities for autonomous robots. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, August 2017.
- Muhannad Alomari, Paul Duckworth, David C. Hogg, and Anthony G. Cohn. Natural language acquisition and grounding for embodied robotic systems. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 4349–4356, February 2017.
- David Alvarez-Melis and Tommi S. Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, September 2017.
- Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. *arXiv preprint arXiv:1711.07280*, 1711, 2017.
- Takaya Araki, Tomoaki Nakamura, Takayuki Nagai, Kotaro Funakoshi, Mikio Nakano, and Naoto Iwahashi. Online object categorization using multimodal information autonomously acquired by a mobile robot. *Advanced Robotics*, 26(17):1995–2020, 2012.

- Yoav Artzi and Luke Zettlemoyer. Bootstrapping semantic parsers from conversations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 421–432, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.
- Yoav Artzi and Luke Zettlemoyer. UW SPF: The University of Washington Semantic Parsing Framework, 2013.
- Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *Transactions of the Association for Computational Linguistics*, 1(1):49–62, 2013.
- Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *arXiv preprint arXiv:1705.09406*, 1705, 2017.
- Kobus Barnard and Matthew Johnson. Word sense disambiguation with pictures. *Artificial Intelligence*, 167:13–30, 2005.
- Daniel Paul Barrett, Scott Alan Bronikowski, Haonan Yu, and Jeffrey Mark Siskind. Robot language learning, generation, and comprehension. *CoRR*, abs/1508.06161, 2015.
- Emanuele Bastianelli, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. Textual inference and meaning representation in human robot interaction. In *Joint Symposium on Semantic Processing*, Trento, Italy, 2013.
- Emanuele Bastianelli, Danilo Croce, Andrea Vanzo, Roberto Basili, and Daniele Nardi. A discriminative approach to grounded spoken language understanding in interactive robotics. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2747–2753, July 2016.
- J. Berant, A. Chou, R. Frostig, and P. Liang. Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2013.
- Yonatan Bisk, Deniz Yuret, and Daniel Marcu. Natural language communication with robots. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 751–761, San Diego, CA, June 2016.

- Stefan Bordag. Word sense induction: Triplet-based clustering and automatic evaluation. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 137–144, 2006.
- Simon Brodeur, Ethan Perez, Ankesh Anand, Florian Golemo, Luca Celotti, Florian Strub, Jean Rouat, Hugo Larochelle, and Aaron Courville. Home: A household multimodal environment. *arXiv preprint arXiv:1711.11017*, 1711, 2017.
- Elia Bruni, Nam Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47, 2014.
- Benjamin Burchfiel and George Konidaris. Generalized 3d object representations using bayesian eigenobjects. In *Proceedings of Robotics: Science and Systems (RSS)*, 2017.
- M. Cakmak and A. L. Thomaz. Designing robot learners that ask good questions. In *Human-Robot Interaction (HRI), 2012 7th ACM/IEEE International Conference on*, pages 17–24, March 2012.
- M. Cakmak, C. Chao, and A. L. Thomaz. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development*, 2(2):108–118, June 2010.
- Rehj Cantrell, Kartik Talamadupula, Paul Schermerhorn, J. Benton, Subbarao Kambhampati, and Matthias Scheutz. Tell me when and why to do it!: Runtime planner model updates via natural language instruction. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 471–478, New York, NY, USA, 2012. ACM.
- Shiyu Chang, Yang Zhang, Jiliang Tang, Dawei Yin, Yi Chang, Mark A. Hasegawa-Johnson, and Thomas S. Huang. Positive-unlabeled learning in streaming networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, 2016.
- Xinlei Chen, Alan Ritter, Abhinav Gupta, and Tom Mitchell. Sense Discovery via Co-Clustering on Images and Text. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.
- Yanhua Cheng, Xin Zhao, Rui Cai, Zhiwei Li, Kaiqi Huang, and Yong Rui. Semi-supervised multimodal deep learning for rgb-d object recognition. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3345–3351, July 2016.

- Gordon Christie, Ankit Laddha, Aishwarya Agrawal, Stanislaw Antol, Yash Goyal, Kevin Kochersberger, and Dhruv Batra. Resolving language and vision ambiguities together: Joint segmentation & prepositional attachment resolution in captioned scenes. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.
- Vivian Chu, Ian McMahon, Lorenzo Riano, Craig G McDonald, Qin He, Jorge Martinez Perez-Tejada, Michael Arrigo, Naomi Fitter, John C Nappo, Trevor Darrell, et al. Using robotic exploratory procedures to learn the meaning of haptic adjectives. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3048–3055. IEEE, 2013.
- Rodolfo Corona, Jesse Thomason, and Raymond J. Mooney. Improving black-box speech recognition using semantic parsing. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP-17)*, November 2017.
- Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *arXiv preprint*, 2017.
- Abhishek Das, Satwik Kottur, José M.F. Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.
- Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Computer Vision and Pattern Recognition*, 2009.
- Thomas Deselaers and Vittorio Ferrari. Visual and semantic similarity in imagenet. In *Computer Vision and Pattern Recognition*, pages 1777–178, June 2011.
- Antonio Di Marco and Roberto Navigli. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754, 2013.
- Haris Dindo and Daniele Zambuto. A probabilistic approach to learning a visually grounded language model through human-robot interaction. In *International*

- Conference on Intelligent Robots and Systems*, pages 760–796, Taipei, Taiwan, 2010. IEEE.
- Li Dong and Mirella Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 33–43, 2016.
- Kais Dukes. Semeval-2014 task 6: Supervised semantic parsing of robotic spatial commands. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval)*, pages 45–53, 2014.
- Kathleen Eberhard, Hannele Nicholson, Sandra Kübler, Susan Gunderson, and Matthias Scheutz. The Indiana “Cooperative Remote Search Task” (CReST) Corpus. In *Proceedings of the Seventh Language Resources and Evaluation Conference (LREC)*, 2010.
- C. Elkan and K. Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008)*, pages 213–220, 2008.
- Rui Fang, Malcolm Doering, and Joyce Y. Chai. Collaborative models for referring expression generation towards situated dialogue. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, pages 1544–1550, 2014.
- Christiane D. Fellbaum. *WordNet: An Electronic Lexical Database*. MITP, Cambridge, MA, 1998.
- Nicholas FitzGerald, Yoav Artzi, and Luke Zettlemoyer. Learning distributions over logical forms for referring expression generation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.
- Maxwell Forbes and Yejin Choi. Verb physics: Relative physical knowledge of actions and objects. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, August 2017.
- Zhenyong Fu, Tao Xiang, Elyor Kodirov, and Shaogang Gong. Zero-shot object recognition by semantic manifold distance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2644, 2015.

- Nancy Fulda, Daniel Ricks, Ben Murdoch, and David Wingate. What can you do with a rock? affordance extraction via word embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1039–1045, August 2017.
- Q. Gao, M. Doering, S. Yang, and J. Y. Chai. Physical causality of action verbs in grounded language understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, August 2016.
- Yang Gao, Lisa Anne Hendricks, Katherine J Kuchenbecker, and Trevor Darrell. Deep learning for tactile understanding from visual and haptic data. In *International Conference on Robotics and Automation (ICRA)*, pages 536–543. IEEE, 2016.
- Matt Gardner and Jayant Krishnamurthy. Open-vocabulary semantic parsing with both distributional statistics and formal knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 3195–3201, February 2017.
- Guglielmo Gemignani, Emanuele Bastianelli, and Daniele Nardi. Teaching robots parametrized executable plans through spoken interaction. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, May 2015.
- Agustin Gravano, Štefan Benuš, Rivka Levitan, and Julia Hirschberg. Backward mimicry and forward influence in prosodic contour choice in standard american english. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.
- H. Paul Grice. Logic and conversation. In Peter Cole and Jerry Morgan, editors, *Syntax and Semantics 3: Speech Acts*, pages 41–58. Academic Press, New York, 1975.
- Sergio Guadarrama, Erik Rodner, Kate Saenko, and Trevor Darrell. Understanding object descriptions in robotics by open-vocabulary object retrieval and detection. *International Journal of Robotics Research (IJRR)*, 35(1-3):265–280, 2015.
- S. Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.
- Luheng He, Julian Michael, Mike Lewis, and Luke Zettlemoyer. Human-in-the-loop parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2337–2342. Association for Computational Linguistics, 2016.

- Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generative visual explanations. In *Proceedings of the 14th European Conference on Computer Vision (ECCV)*, 2016.
- Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
- Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. In *Proceedings of the 2017 Conference of the Association for Computational Linguistics (ACL)*, 2017.
- Robin Jia and Percy Liang. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 12–22, 2016.
- Olga Kanishcheva and Galia Angelova. *About Sense Disambiguation of Image Tags in Large Annotated Image Collections*, pages 133–149. Springer International Publishing, 2016.
- Piyush Khandelwal, Fangkai Yang, Matteo Leonetti, Vladimir Lifschitz, and Peter Stone. Planning in Action Language \mathcal{BC} while Learning Action Costs for Mobile Robots. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2014.
- Piyush Khandelwal, Shiqi Zhang, Jivko Sinapov, Matteo Leonetti, Jesse Thomason, Fangkai Yang, Ilaria Gori, Maxwell Svetlik, Priyanka Khante, Vladimir Lifschitz, J. K. Aggarwal, Raymond Mooney, and Peter Stone. Bwibots: A platform for bridging the gap between ai and human–robot interaction research. *The International Journal of Robotics Research (IJRR)*, 36, February 2017.
- Douwe Kiela and Stephen Clark. Multi- and cross-modal semantics beyond vision: Grounding in auditory perception. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2461–2470, Lisbon, Portugal, 2015.
- Douwe Kiela, Laura Rimell, Ivan Vulić, and Stephen Clark. Exploiting image generality for lexical entailment detection. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 119–124, Beijing, China, July 2015.

- Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics (ACL) - Volume 1*, pages 423–430, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.
- Ross A. Knepper, Christoforos I. Mavrogiannis, Julia Proft, and Claire Liang. Implicit communication in a joint action. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 283–292, 2017.
- Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, and Karl Moritz Hermann. Semantic parsing with semi-supervised sequential autoencoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, November 2016. Association for Computational Linguistics.
- Elyor Kodirov, Tao Xiang, Zhenyong Fu, and Shaogang Gong. Unsupervised domain adaptation for zero-shot learning. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In *Proceedings of the 5th ACM/IEEE International Conference on Human-robot Interaction, HRI '10*, pages 259–266, 2010.
- Thomas Kollar, Jayant Krishnamurthy, and Grant Strimel. Toward interactive grounded language acquisition. In *Robotics: Science and Systems*, 2013.
- Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 2017 Conference of the Association for Computational Linguistics (ACL)*, 2017.
- Satwik Kottur, Ramakrishna Vedantam, José M. F. Moura, and Devi Parikh. Visual word2vec (vis-w2v): Learning visually grounded word embeddings using abstract scenes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2016.
- Evan Krause, Michael Zillich, Thomas Williams, and Matthias Scheutz. Learning to recognize novel objects in one shot through human-robot interactions in natural language dialogues. In *Proceedings of Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

- Jayant Krishnamurthy and Thomas Kollar. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics*, 1:193–206, 2013.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012.
- Joseph B Kruskal and Myron Wish. *Multidimensional scaling*, volume 11. Sage, 1978.
- H. Kuehne, A. B. Arslan, and T. Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of Computer Vision and Pattern Recognition Conference (CVPR)*, 2014.
- Johannes Kulick, Marc Toussaint, Tobias Lang, and Manuel Lopes. Active learning for teaching a robot grounded relational symbols. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, pages 1451–1457. AAAI Press, 2013.
- Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1223–1233, 2010.
- Tom Kwiatkowski, Eunsol Choi, Yoav Artzi, and Luke Zettlemoyer. Scaling semantic parsers with on-the-fly ontology matching. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1545–1556, 2013.
- Jessica L. Lakin, Valerie E. Jefferis, Clara Michelle Cheng, and Tanya L. Chartrand. The chameleon effect as social glue: Evidence for the evolutionary significance of nonconscious mimicry. *Journal of Nonverbal Behavior*, 27(3):145–162, 2003.
- Angeliki Lazaridou, Nghia The Pham, and Marco Baroni. Combining language and vision with a multimodal skipgram model. In *The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2015.

- Joohyung Lee, Vladimir Lifschitz, and Fangkai Yang. Action Language *BC*: A Preliminary Report. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2013.
- Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Global neural ccg parsing with optimality guarantees. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2366–2376, Austin, Texas, November 2016. Association for Computational Linguistics.
- David D Lewis and William A Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.
- Mike Lewis, Kenton Lee, and Luke Zettlemoyer. Lstm ccg parsing. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2016.
- Percy Liang and Cristopher Potts. Bringing machine learning and compositional semantics together. *Annual Review of Linguistics*, 1(1):355–376, 2015.
- P. Liang, M. I. Jordan, and D. Klein. Learning semantic correspondences with less supervision. In *Association for Computational Linguistics and International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 91–99, 2009.
- P. Liang, M. I. Jordan, and D. Klein. Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*, pages 590–599, 2011.
- Changsong Liu and Joyce Yue Chai. Learning to mediate perceptual differences in situated human-robot dialogue. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pages 2288–2294, 2015.
- Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip Yu. Building text classifiers using positive and unlabeled examples. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM-03)*, 2003.
- Changson Liu, Lanbo She, Rui Fang, and Joyce Y. Chai. Probabilistic labeling for efficient referential grounding based on collaborative discourse. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 13–18, Baltimore, Maryland, USA, 2014.

- Nicolas Loeff, Cecilia Ovesdotter Alm, and David A. Forsyth. Discriminating image senses by clustering with multimodal features. In *Proceedings of the COLING/ACL on Main Conference Poster Sessions, COLING-ACL '06*, pages 547–554, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.
- José Lopes, Maxine Eskenazi, and Isabel Trancoso. Automated two-way entrainment to improve spoken dialog system performance. In *International Conference on Acoustics Speech and Signal Processing (ICASSP)*. IEEE, 2013.
- Dongcai Lu and Xiaoping Chen. Towards an architecture combining grounding and planning for human-robot interaction. In *RoboCup*, pages 214–225, 2015.
- Nichola Lubold, Erin Walker, and Heather Pon-Barry. Relating entrainment, grounding, and topic of discussion in collaborative learning dialogues. In *Proceedings of the 11th International Conference on Computer Supported Collaborative Learning (CSCL)*, 2015.
- Aurelien Lucchi and Jason Weston. Joint image and word sense discrimination for image retrieval. In *Computer Vision–ECCV*, pages 130–143. Springer, 2012.
- Ingo Lutkebohle, Julia Peltason, Lars Schillingmann, Britta Wrede, Sven Wachsmuth, Christof Elbrechter, and Robert Haschke. The curious robot-structuring interactive robot learning. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 4156–4162. IEEE, 2009.
- Dermot Lynott and Louise Connell. Modality exclusivity norms for 423 object properties. *Behavior Research Methods*, 41(2):558–564, 2009.
- Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems*, pages 13–18, Montréal, Canada, 2014.
- Suresh Manandhar, Ioannis P. Klapaftis, Dmitriy Dligach, and Sameer S. Pradhan. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10*, pages 63–68, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.
- Junhua Mao, Xu Wei, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan L. Yuille. Learning like a child: Fast novel visual concept learning from sentence descriptions of images. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

- Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK, 2012.
- Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *International Symposium on Experimental Robotics (ISER)*, 2012.
- Cynthia Matuszek, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *Experimental Robotics*, pages 403–415. Springer International Publishing, 2013.
- Cynthia Matuszek, Liefeng Bo, Luke Zettlemoyer, and Dieter Fox. Learning from unscripted deictic gesture and language for human-robot interactions. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence*, Québec City, Québec, Canada, 2014.
- Diana McCarthy and Roberto Navigli. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 48–53. Association for Computational Linguistics, 2007.
- Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *Proceedings of AAAI*, 2016.
- Oren Melamud, Omer Levy, and Ido Dagan. A simple word embedding model for lexical substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 1–7, Denver, Colorado, June 2015. Association for Computational Linguistics.
- Çetin Meriçli, Steven D. Klee, Jack Paparian, and Manuela Veloso. An interactive approach for situated task specification through verbal instructions. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, pages 1069–1076, 2014.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*, pages 3111–3119, Lake Tahoe, Nevada, 2013.

- Dipendra Misra and Yoav Artzi. Neural shift-reduce ccg semantic parsing. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1775–1786. Association for Computational Linguistics, 2016.
- Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me dave: Context-sensitive grounding of natural language to mobile manipulation instructions. In *Robotics: Science and Systems, RSS*, 2014.
- Ishan Misra, Abhinav Gupta, and Martial Hebert. From Red Wine to Red Tomato: Composition with Context. In *CVPR*, 2017.
- Shiwali Mohan, Aaron H Mininger, James R Kirk, and John E Laird. Acquiring grounded representations of words with situated interactive instruction. In *Advances in Cognitive Systems*, 2012.
- Shiwali Mohan, Aaron H. Mininger, and John E. Laird. Towards an indexical model of situated language comprehension for real-world cognitive agents. In *Proceedings of the 2nd Annual Conference on Advances in Cognitive Systems*, Baltimore, Maryland, USA, 2013.
- Roberto Navigli and Daniele Vannella. Semeval-2013 task 11: Word sense induction and disambiguation within an end-user application. In *In Second Joint Conference on Lexical and Computational Semantics*, 2013.
- Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10, 2009.
- Aishwarya Padmakumar, Jesse Thomason, and Raymond J. Mooney. Integrated learning of dialog strategies and semantic parsing. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 547–557, April 2017.
- Natalie Parde, Adam Hair, Michalis Papakostas, Konstantinos Tsiakas, Maria Dagioglou, Vangelis Karkaletsis, and Rodney D. Nielsen. Grounding the meaning of words through vision and interactive gameplay. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 1895–1901, Buenos Aires, Argentina, 2015.
- Rohan Paul, Jacob Arkin, Nicholas Roy, and Thomas M. Howard. Efficient grounding of abstract spatial concepts for natural language interaction with robot manipulators. In *Proceedings of Robotics: Science and Systems*, Ann Arbor, Michigan, June 2016.

- Rohan Paul, Andrei Barbu, Sue Felshin, Boris Katz, and Nicholas Roy. Temporal grounding graphs for language understanding with accrued visual-linguistic context. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4506–4514, 2017.
- Ted Pedersen and Rebecca Bruce. Distinguishing word senses in untagged text. In *Proceedings of the Second Conference on Empirical Methods in Natural Language*, pages 197–207, 1997.
- Ian Perera and James F. Allen. Sall-e: Situated agent for language learning. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pages 1241–1247, Bellevue, Washington, USA, 2013.
- Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source robot operating system. In *Open Source Software in Robotics Workshop at the IEEE International Conference on Robotics and Automation (ICRA)*, 2009.
- Joseph Reisinger and Raymond J. Mooney. Multi-prototype vector-space models of word meaning. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT-10)*, pages 109–117, 2010.
- Lazlo Ring, Dina Utami, Stefan Olafsson, and Timothy Bickmore. Increasing engagement with virtual agents using automatic camera motion. In *International Conference on Intelligent Virtual Agents*, pages 29–39. Springer, 2016.
- Andrew Rosenberg and Julia Hirschberg. V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, 2007.
- Deb Roy and Alex Pentland. Learning words from sights and sounds: a computational model. *COGSCI*, 26(1):113–146, 2002.
- Deb Roy. Learning visually grounded words and syntax of natural spoken language. *Evolution of Communication*, 4(1), 2001.
- Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE, 2009.

- Kate Saenko and Trevor Darrell. Unsupervised learning of visual sense models for polysemous words. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1393–1400. 2008.
- Hinrich Schutze. Automatic word sense discrimination. *Computational Linguistics*, 24(1):97–123, 1998.
- Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics, 2008.
- Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 52(55-66):11, 2010.
- Manali Sharma and Mustafa Bilgic. Evidence-based uncertainty sampling for active learning. *Data Mining and Knowledge Discovery*, 31(1):164–202, 2017.
- Lanbo She, Shaohua Yang, Yu Cheng, Yunyi Jia, Joyce Y. Chai, and Ning Xi. Back to the blocks world: Learning new actions through situated human-robot dialogue. In *Proceedings of 15th SIGDIAL Meeting on Discourse and Dialogue*, 2014.
- Carina Silberer and Mirella Lapata. Grounded models of semantic representation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1423–1433, Jeju Island, Korea, July 2012. Association for Computational Linguistics.
- Carina Silberer and Mirella Lapata. Grounded meaning representations with autoencoders. In *In Proceedings of the Association for Computational Linguistics (ACL)*, 2014.
- Carina Silberer, Vittorio Ferrari, and Mirella Lapata. Visually grounded meaning representations. *IEEE transactions on pattern analysis and machine intelligence*, 39(11):2284–2297, 2017.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

- Jivko Sinapov, Connor Schenck, Kerrick Staley, Vladimir Sukhoy, and Alexander Stoytchev. Grounding semantic categories in behavioral interactions: Experiments with 100 objects. *Robotics and Autonomous Systems*, 62(5):632–645, 2014.
- Jivko Sinapov, Connor Schenck, and Alexander Stoytchev. Learning relational object categories using behavioral exploration and multimodal perception. In *IEEE International Conference on Robotics and Automation*, 2014.
- Jivko Sinapov, Priyanka Khante, Maxwell Svetlik, and Peter Stone. Learning to order objects using haptic and proprioceptive exploratory behaviors. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.
- Danijel Skočaj, Alen Vrečko, Marko Mahnič, Miroslav Janíček, Geert-Jan M Kruijff, Marc Hanheide, Nick Hawes, Jeremy L Wyatt, Thomas Keller, Kai Zhou, et al. An integrated system for interactive continuous learning of categorical knowledge. *Journal of Experimental & Theoretical Artificial Intelligence*, 28(5):823–848, 2016.
- Michael Spranger and Luc Steels. Co-acquisition of syntax and semantics—an investigation of spatial language. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 1909–1915, Buenos Aires, Argentina, 2015.
- Mark Steedman and Jason Baldridge. Combinatory categorial grammar. In Robert Borsley and Kersti Borjars, editors, *Non-Transformational Syntax: Formal and Explicit Models of Grammar*. Wiley-Blackwell, 2011.
- Pei-Hao Su, Milica Gašić, and Steve Young. Reward estimation for dialogue policy optimisation. *Computer Speech and Language*, 2018.
- Yuyin Sun, Liefeng Bo, and Dieter Fox. Attribute based object identification. In *International Conference on Robotics and Automation*, pages 2096–2103, Karlsruhe, Germany, 2013. IEEE.
- Paul Taylor, Alan W Black, and Richard Caley. The architecture of the festival speech synthesis system. In *In the Third ESCA Workshop in Speech Synthesis*, pages 147–151, 1998.

- Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R. Walter, Ashis Gopal Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2011.
- Stefanie Tellex, Ross Knepper, Adrian Li, Daniela Rus, and Nicholas Roy. Asking for help using inverse semantics. In *Proceedings of Robotics: Science and Systems (RSS)*, Berkeley, California, 2014.
- Wil Thomason and Ross A. Knepper. Recognizing unfamiliar gestures for human-robot interaction through zero-shot learning. In *International Symposium on Experimental Robotics (ISER)*, 2016.
- Jesse Thomason and Raymond J. Mooney. Multi-modal word synset induction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI-17)*, pages 4116–4122, August 2017.
- Jesse Thomason, Huy Nguyen, and Diane Litman. Prosodic entrainment and tutoring dialogue success. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED)*, pages 750–753, July 2013.
- Jesse Thomason, Shiqi Zhang, Raymond Mooney, and Peter Stone. Learning to interpret natural language commands through human-robot dialog. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1923–1929, July 2015.
- Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond Mooney. Learning multi-modal grounded linguistic semantics by playing “I spy”. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3477–3483, July 2016.
- Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Justin Hart, Peter Stone, and Raymond J. Mooney. Opportunistic active learning for grounding natural language descriptions. In *Proceedings of the 1st Annual Conference on Robot Learning (CoRL-17)*, volume 78, pages 67–76. Proceedings of Machine Learning Research, November 2017.
- Jesse Thomason, Jivko Sinapov, Raymond Mooney, and Peter Stone. Guiding exploratory behaviors for multi-modal grounding of linguistic descriptions. In *Proceedings of the 32nd Conference on Artificial Intelligence (AAAI-18)*, February 2018.

- Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.
- Mycal Tucker, Derya Aksaray, Rohan Paul, Gregory J. Stein, and Nicholas Roy. Learning unknown groundings for natural language interaction with mobile robots. In *International Symposium of Robotics Research, ISRR-17*, 2017.
- Devis Tuia and Gustau Camps-Valls. Kernel manifold alignment for domain adaptation. *PLoS ONE*, 11, 2016.
- Ashwin K Vijayakumar, Ramakrishna Vedantam, and Devi Parikh. Soundword2vec: Learning word representations grounded in sounds. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 920–925, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.
- Adam Vogel, Karthik Raghunathan, and Dan Jurafsky. Eye spy: Improving vision through dialog. In *Association for the Advancement of Artificial Intelligence*, pages 175–176, 2010.
- Matthew Walter, Sachithra Hemachandra, Bianca Homberg, Stefanie Tellex, and Seth Teller. Learning semantic maps from natural language descriptions. In *Proceedings of Robotics: Science and Systems (RSS)*, Berlin, Germany, 2013.
- Sida I Wang, Percy Liang, and Christopher D Manning. Learning language games through interaction. In *Proceedings of the 2017 Conference of the Association for Computational Linguistics (ACL)*, 2017.
- David Whitney, Miles Eldon, John Oberlin, and Stefanie Tellex. Interpreting Multimodal Referring Expressions in Real Time. In *International Conference on Robotics and Automation*, 2016.
- David Whitney, Eric Rosen, James MacGlashan, Lawson L. S. Wong, and Stefanie Tellex. Reducing errors in object-fetching interactions through social feedback. In *IEEE International Conference on Robotics and Automation (ICRA)*, June 2017.
- Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning - the good, the bad and the ugly. In *Proceedings of the 30th IEEE Conference on Computer Vision and Pattern Recognition (CVPR-17)*, July 2017.

- Yezhou Yang, Cornelia Fermüller, Yiannis Aloimonos, and Anupam Guha. A cognitive system for understanding human manipulation actions. *Advances in Cognitive Systems*, 3:67–86, 2014.
- David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL95*, pages 189–196, 1995.
- Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.
- Daniel Younger. Recognition and parsing of context-free languages in time n^3 . *Information and Control*, 10:189–208, 1967.
- Onur Yürüten, Erol Şahin, and Sinan Kalkan. The learning of adjectives and nouns from affordance and appearance features. *Adaptive Behavior*, 21(6):437–451, 2013.
- Shiqi Zhang, Jivko Sinapov, Suhua Wei, and Peter Stone. Robot behavioral exploration and multimodal perception using pomdps. In *AAAI Spring Symposium on Interactive Multi-sensory Object Perception for Embodied Agents*, March 2017.
- Larry Zitnick and Devi Parikh. Bringing semantics into focus using visual abstraction. In *Computer Vision and Pattern Recognition (CVPR)*, December 2013.