The Dissertation Committee for Joo Hyun Kim
certifies that this is the approved version of the following dissertation:

Grounded Language Learning Models for Ambiguous Supervision

Committee:

Raymond J. Mooney, Supervisor

Jason Baldridge

Dana Ballard

Percy Liang

Peter Stone

Grounded Language Learning Models for Ambiguous Supervision

by

Joo Hyun Kim, B.S.; M.S.C.S.

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

December 2013

Dedicated to my loving wife.

Acknowledgments

First of all, I would like to thank my advisor Raymond Mooney. Over the past six years at UT, he has provided a great guidance for my research. I still can't forget the moment that he has given me the opportunity to work together in the Machine Learning Research Group. That was one of the best days in my life at UT. Ever since then, he has shown me endless support and trust. While I was struggling in my research, his insightful comments and advice based on his vast experiences have always helped me to find a way to a successful approach. I also deeply admire his great passion about the research, which has greatly affected my attitude toward the life and the research. I will always miss him and the good times we have had in our weekly meetings.

I want to thank all my committee members—Jason Baldridge, Dana Ballard, Percy Liang, and Peter Stone—for their helpful comments and suggestions that helped improve this thesis. I want to give my special gratitude to Percy Liang for his precious time to visit Austin to attend my thesis defense.

I also would like to thank all the members of the Machine Learning Research Group whom I have worked with. I especially thank David Chen for his insightful ideas and advice that incredibly helped my works in this thesis. I also want to thank Sonal Gupta for my first collaborative publication which extended our class project together. Finally, I want to thank the rest

of the group: Rohit Kate, Lilyana Mihalkova, Joe Reisinger, Tuyen Huynh, Parag Singla, Sindhu Raghavan, Yinon Bentor, Bishal Barman, Dan Garette, Lu Guo, Ayan Acharya, Hyeonseo Ku, Tanvi Motwani, Karl Pichotta, Islam Beltagy, Niveda Krishnamoorthy, Girish Malkarnenkar, Gemma Boleda, Cuong Chau, Shruti Bhosale, Amelia Harrison, and Subhashini Venugopalan for attending my practice talks and giving me helpful feedback and advice to improve my research and presentation.

I also want to thank the members of UTCS department staff for their kind help for the past six years. Especially, Katherine Utz, Stacy Miller, Gloria Ramirez, and Lydia Griffith have promptly responded to me for any help for the administrative issues and have been incredibly resourceful.

I am also grateful to my Korean friends in Austin whom I made friendship with from the department and the church. They were like my second family while my wife and I have been here, away from our families.

I want to thank the UTCS department and the National Science Foundation (NSF) for supporting the fund for my Ph.D research. The research presented in this thesis was supported by the NSF grants IIS-0712097 and IIS-1016312. I also thank Samsung Scholarship for the financial support during my first 5 Ph.D years at UT.

Finally, I would like to thank my family including my parents, my younger brother Minwoo, and my wife Ahra. It would have not been possible for me to complete the six-year journey of Ph.D work without their support

and love. I especially thank Ahra for being my wife and understanding all the long years that I have spent as a student with her greatest support.

Joo Hyun Kim

The University of Texas at Austin
December 2013

Grounded Language Learning Models for Ambiguous Supervision

Publication No. _____

Joo Hyun Kim, Ph.D.
The University of Texas at Austin, 2013

Supervisor: Raymond J. Mooney

Communicating with natural language interfaces is a long-standing, ultimate goal for artificial intelligence (AI) agents to pursue, eventually. One core issue toward this goal is "grounded" language learning, a process of learning the semantics of natural language with respect to relevant perceptual inputs. In order to ground the meanings of language in a real world situation, computational systems are trained with data in the form of natural language sentences paired with relevant but ambiguous perceptual contexts. With such ambiguous supervision, it is required to resolve the ambiguity between a natural language (NL) sentence and a corresponding set of possible logical meaning representations (MR).

In this thesis, we focus on devising effective models for simultaneously disambiguating such supervision and learning the underlying semantics of language to map NL sentences into proper logical MRs. We present probabilistic

generative models for learning such correspondences along with a reranking model to improve the performance further.

First, we present a probabilistic generative model that learns the mappings from NL sentences into logical forms where the true meaning of each NL sentence is one of a handful of candidate logical MRs. It simultaneously disambiguates the meaning of each sentence in the training data and learns to probabilistically map an NL sentence to its corresponding MR form depicted in a single tree structure. We perform evaluations on the RoboCup sportscasting corpus, proving that our model is more effective than those proposed by previous researchers.

Next, we describe two PCFG induction models for grounded language learning that extend the previous grounded language learning model of Börschinger, Jones, and Johnson (2011). Börschinger et al.'s approach works well in situations of limited ambiguity, such as in the sportscasting task. However, it does not scale well to highly ambiguous situations when there are large sets of potential meaning possibilities for each sentence, such as in the navigation instruction following task first studied by Chen and Mooney (2011). The two models we present overcome such limitations by employing a learned semantic lexicon as a basic correspondence unit between NL and MR for PCFG rule generation.

Finally, we present a method of adapting discriminative reranking to grounded language learning in order to improve the performance of our proposed generative models. Although such generative models are easy to imple-

ment and are intuitive, it is not always the case that generative models perform best, since they are maximizing the joint probability of data and model, rather than directly maximizing conditional probability. Because we do not have gold-standard references for training a secondary conditional reranker, we incorporate weak supervision of evaluations against the perceptual world during the process of improving model performance.

All these approaches are evaluated on the two publicly available domains that have been actively used in many other grounded language learning studies. Our methods demonstrate consistently improved performance over those of previous studies in the domains with different languages; this proves that our methods are language-independent and can be generally applied to other grounded learning problems as well. Further possible applications of the presented approaches include summarized machine translation tasks and learning from real perception data assisted by computer vision and robotics.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Understanding and learning the semantics of natural language is one of the long-standing, ultimate goals of artificial intelligence (AI) and natural language processing (NLP) research. It is a core ability of computers to communicate with humans in as natural a way as humans do among themselves, as opposed to using structured and digitized methods such as mouse clicks and keyboard inputs. This is the ultimate objective for intelligent systems to pursue.

"Language grounding," a process of mapping natural language to relevant aspects of a surrounding perceptual environment, is one approach to this goal. A human child "grounds" language in perceptual contexts via repetitive exposure to the co-occurrence of language and perception. Recent research supports the idea that the human language learning process also happens in a *statistical* manner (Saffran, Johnson, Aslin, & Newport, 1999; Saffran, 2003). Ideally, language grounding systems should be able to mimic the language learning process of humans.

A number of researchers have attempted to model the grounded language learning of humans (Bailey, Feldman, Narayanan, & Lakoff, 1997; Roy,

2002; Barnard, Duygulu, Forsyth, de Freitas, Blei, & Jordan, 2003; Yu & Ballard, 2004; Gold & Scassellati, 2007; Fleischman & Roy, 2008; Branavan, Chen, Zettlemoyer, & Barzilay, 2009; Liang, Jordan, & Klein, 2009; Vogel & Jurafsky, 2010; Feng & Lapata, 2010; Branavan, Silver, & Barzilay, 2011; Tellex, Kollar, Dickerson, Walter, Banerjee, Teller, & Roy, 2011). All these previous studies are related to one or more fields of AI research including computer vision, robotics, natural language processing, cognitive science, and psychology. Although these studies confront the problem from varying perspectives, their common objective is to connect underlying meanings of natural language to surrounding raw perceptions that are naturally observed.

Many other previous approaches try to understand the semantics of language in a way that finds the direct relevance of language to real world perceptions in order to perform actual tasks. In contrast in the present study, we concentrate more on language learning itself, while minimizing other issues concerning real perceptions that might involve computer vision or robotics. Thus, we simplify the problem by abstracting real perceptions into machine-interpretable logical forms using off-the-shelf automated systems. That way, we need only need be concerned about how language is connected to the components of logical forms. This is advantageous in that we can avoid the excessive complexity that could potentially occur with noisy raw sensory or visual data and free-form natural language. It also simplifies the entire problem by dividing it in two: language understanding solved with natural language processing, and perception understanding handled by computer vision, cognitive

science, and/or robotics.

Semantic parsing is an area of research that investigates how to translate and interpret meanings of complete natural language (NL) sentences into formal, logical meaning representations (MR) (Zelle & Mooney, 1996; Zettlemoyer & Collins, 2005; Kate & Mooney, 2006; Wong & Mooney, 2006, 2007b; Zettlemoyer & Collins, 2007; Lu, Ng, Lee, & Zettlemoyer, 2008; Zettlemoyer & Collins, 2009). Conventional semantic parsing approaches require fully annotated corpora where one NL sentence is paired with one translated complete logical form. Typically, they are trained in a supervised manner with a few hundred to a few thousand one-to-one annotated training example pairs. Although such conventional methods have been proven to work well in several domains, it is a non-trivial task to extend these methods to large-scale systems. We need the assistance of human experts to create the necessary parallel corpora, and we especially need the specialists who have knowledge in the fields of both natural and formal languages. Therefore, the entire process of such annotation is inevitably very time-consuming and difficult to accomplish.

Instead, inspired by how a human child learns natural language, the present study focuses on more relaxed, natural settings of supervision. The training data usually consists of each NL sentence along with multiple, potentially relevant logical forms describing the current perceptual states. However, our goal is not to mimic precisely a human language learning mechanism. Rather, we seek to learn the semantics of language more naturally by getting ambiguously supervised data, which are usually easier to obtain. Normally,

the data can be collected in such a way that humans first produce NL utterances describing a certain phenomenon/action/state while all the relevant surrounding perceptions are recorded separately through an automated process without costly additional human annotation. In this sense, the language learning systems used in the present study "ground" language naturally with surrounding perceptions.

In this thesis, we explore several approaches to solving the problems of grounded language learning. In the typical setting of such problems, we face referential ambiguity in that an NL description may refer to one or more potentially relevant perceptions formalized by a large set of MRs, and this is the major challenge of grounded language learning. We investigate generative methods that integrate and describe NL segments and logical components in a single hierarchical structure, as well as resolving such ambiguity in a probabilistic framework. In addition, these generative models also learn how to map novel NL sentences into proper logical forms by simultaneously disambiguating ambiguous supervision.

We evaluate our methods in two different domains with different levels of ambiguity. First, we describe a simultaneous alignment and semantic parsing model that solves the previous task of learning how to sportscast in virtual RoboCup 2D soccer games (Chen & Mooney, 2008). The training data consists of NL commentary on the recorded videos of games as well as automatically extracted logical forms representing abstracted events happening concurrently. Thus, the training data have inherent ambiguity in that

each NL commentary has zero or one true meaning out of several candidates for meanings. Chen and Mooney (2008) first attempted to solve this challenge via a hard Expectation-Maximization (EM) algorithm that is developed by Kate and Mooney (2007) for artificially created data. This approach is likely to suffer from information loss, because in each retraining iteration, the model learns parameters from only the most probable MR match for each NL sentence. By contrast, the generative model proposed in the present study probabilistically selects the correct alignment as well as subsequent components of logical forms and natural language words and retains probabilistic counts for such relationships. Our approach is capable of disambiguating the match between language and meanings while also learning a complete semantic parser for mapping sentences to logical forms. Evaluation results on the RoboCup sportscasting domain show that our approach outperforms previous results on the NL–MR alignment task and language generation and also produces competitive performance on semantic parsing.

Next, we present two unsupervised *probabilistic context-free grammar* (PCFG) induction models evaluated on the navigation task previously investigated by Chen and Mooney (2011). The navigation task involves a much higher level of ambiguity. In this task, each instruction is paired with a formal *landmarks plan* that includes a full description of the observed actions and world-states that result when someone follows this instruction. The main challenge here is that the instruction refers to only a subset of this full description, which inevitably results in exponentially many potential alignments

between each NL instruction and its correct logical form. Our model is a novel enhancement of an existing grounded language learning approach using unsupervised induction of PCFGs (Börschinger et al., 2011). Our model uses semantic lexemes as the basic building blocks for PCFG rules to avoid a combinatorial explosion in the number of matchings between components of NL words and logical forms. The semantic lexicon confines the PCFG rule set to a tractable size compared to Börschinger et al.'s approach, while still exploiting full probabilistic predictions. We present two versions of such a PCFG induction model, one of which follows the hierarchy of semantic concepts while the other uses a more simplified unigram generation of concepts. Experimental results show our approaches are better than those of previous studies both in partial parsing accuracy and in end-to-end execution results.

Furthermore, we introduce a *discriminative reranking* (Collins, 2000) approach to improve the performance of the proposed generative models. Discriminative reranking is a common machine learning method for improving the accuracy of a generative model significantly with the help of an additional conditional model on top of the generative model. A reranker uses the global features of complete parses to identify correct interpretations in order to train a discriminative classifier, which finally produces improved results on a novel test set. Reranking has been successfully employed to improve various tasks of natural language processing, including syntactic parsing (Collins, 2002b), semantic parsing (Ge & Mooney, 2006; Lu et al., 2008), semantic role labeling (Toutanova, Haghighi, & Manning, 2005), and named entity recog-

nition (Collins, 2002c). While conventional reranking requires gold-standard interpretations (e.g., parse trees) to train the discriminative classifier, it is non-trivial for grounded language learning since normally it does not provide gold-standard interpretations as training data. Only the ambiguous perceptual context of the NL utterance is given. The navigation task provides the observed sequence of actions taken by a human when following an instruction as supervision. Therefore, it is impossible to directly apply conventional discriminative reranking to such problems. We show how to adapt reranking to work with such weak supervision. Instead of using gold-standard annotations to determine the correct interpretations, we simply use the reference as evaluations of candidate interpretations of navigation instructions that are executed in the perceptual world, observing how well they reach the intended destination.

1.1 Thesis Contributions

This thesis makes two primary contributions to the area of grounded language learning. First, we suggest several probabilistic generative models that solve grounded language learning problems which tackle two different levels of NL–MR referential ambiguity. Compared to the previous methods that have been evaluated on the same tasks, our methods employ probabilistic approaches that incorporate a more intuitive, effective hierarchy of semantic concepts. In particular, our methods do not suffer from the possible information loss that may have befallen previous methods due to pipelines of process-

ing stages. In addition, our methods are able to disambiguate true matchings out of a potentially large set of relevant MRs, as well as to perform accurate semantic interpretation of natural language utterances. Experimental results prove that our methods are more effective when compared to previous best results.

Second, we show how discriminative reranking approaches can be extended and applied on top of generative models for grounded language learning problems. Discriminative reranking improves the performance of generative models with a secondary conditional model, and has been proven to be effective in many NLP tasks in the past. However, it is non-trivial to apply discriminative reranking directly to grounded language learning problems without gold-standard references for each training example. For the first time, we propose that evaluating the candidate parses from the outputs of a generative model against the world-state eliminates the need for gold-standard reference interpretations. Even though our suggested method uses a simple perceptron model, the general methodology of evaluating against the world to get weak supervision can be broadly applied to many other available discriminative reranking models. In addition, we prove that such adaptation is effective in boosting the performance of the original model by a large margin and even better than the standard reranking model with gold-standards in end-to-end evaluations.

1.2 Thesis Outline

The remainder of this thesis is organized as follows:

- Chapter 2 reviews previous research that this thesis directly builds upon and explains how it is incorporated in the models we present.
- Chapter 3 presents a review of the sportscasting task and a generative model that resolves 1-to-N ambiguity in a hierarchical framework.
- Chapter 4 explains the navigation task and also describes our PCFG induction models that tackle highly ambiguous supervision. Two variations are presented and evaluated: one with a hierarchy of semantic concepts and the other with unigram generation of the concepts.
- Chapter 5 describes how to adapt discriminative reranking to grounded language learning where gold-standard references are not provided.
- Chapter 6 reviews related work in grounded language learning, learning from ambiguous supervision, conventional semantic parsing, and natural language generation.
- Chapter 7 discusses directions for future research
- Chapter 8 offers our final conclusion.

It should be noted that some of the research discussed in Chapter 3, Chapter 4, and Chapter 5 has been already presented in our previous publications (Kim & Mooney, 2010, 2012, 2013).

Chapter 2

Background

In this chapter, we will cover previous systems or models that our proposed methods are built upon. First, we describe supervised semantic parsing methods that the models presented in this study for the sportscasting task (see Chapter 3) are based on. Our model is built on the generative semantic parsing model of Lu et al. (2008). After learning a probabilistic alignment and parsing model, we also used the WASP and WASP^{-1} systems to produce additional parsing and generation results. More specifically, since our current model is incapable of effectively generating NL sentences from MR logical forms, in order to demonstrate how our matching results can aid NL generation, we use WASP^{-1} to learn a generator. This follows the experimental scheme of Chen, Kim, and Mooney (2010), who had demonstrated that the improved NL–MR matching of Liang et al. (2009) results in better overall parsing and generation.

Next, we will review two grounded language learning methods that learn from ambiguous supervision. IGSL (Iterative Generation Strategy Learning) was first proposed by Chen and Mooney (2008) as a method of estimating the probabilities of "content selection" for natural language generation. It de-

cides "what to say" out of an ambiguous set of possible sportscasting events to describe. Our generative model in Chapter 3 incorporates IGSL to estimate the prior probability of each event-type generating a natural-language comment and to help initialize the model for a better starting point of EM. Then, we describe the grounded language learning approach of Börschinger et al. (2011), which, in turn, was inspired by a series of previous techniques (Lu et al., 2008; Liang et al., 2009; Kim & Mooney, 2010) based on the idea of constructing correspondences between NL and MR in a single probabilistic generative framework. Specifically, Börschinger et al.'s approach automatically constructs a PCFG rule set that generates NL sentences from MRs, which indicates how atomic MR constituents are probabilistically related to NL words. This approach is able to handle not only conventional supervised semantic parsing problems, but also grounded language learning problems with limited ambiguity. It has been shown to be very effective in the sportscasting domain, but because the generative process finds probabilistic connection between each MR component with NL phrases, the model easily suffers from exponential complexity with respect to the size of the MR language. Our model introduced in Chapter 4 extends this approach by applying it to more complex problems with the aid of a statistically learned semantic lexicon.

Finally, we will briefly review about conventional discriminative reranking approaches. These are effective tools for improving the performance of a generative model proven in various tasks, but they require gold-standard annotation for training that is not naturally provided in grounded language learning

tasks. Chapter 5 proposes a way to get around this limitation of conventional reranking methods and modify them to work in grounded learning settings.

2.1 Semantic Parsing Approaches

Semantic parsing is a process of translating and interpreting the semantics of full natural language (NL) sentences into formal, logical meaning representations (MR). In conventional settings, one-to-one, fully translated NL–MR pairs are needed for training supervised semantic parsing approaches. Our model proposed in Chapter 3 is built upon such supervised semantic parsing models, but extends the supervised models in order to handle more relaxed, ambiguous supervision.

2.1.1 Generative Hybrid Tree Model for Semantic Parsing

Lu et al. (2008) proposed a generative semantic parsing model using a hybrid tree framework. A *hybrid tree* is defined over a pair $(\mathbf{w}, \mathbf{m})$ of an NL sentence and its corresponding MR. The tree describes a correspondence of NL word segments and MR components following the grammatical structure of the MR. In a hybrid tree, MR production rules constitute the internal nodes, while NL words (or phrases) constitute the leaves. A sample hybrid tree from the English RoboCup data is shown in Figure 2.1.

A generative process based on hybrid trees is defined as follows: starting from a root semantic category, the model generates a production of the MR grammar, and then subsequently generates a mixed hybrid pattern of NL words

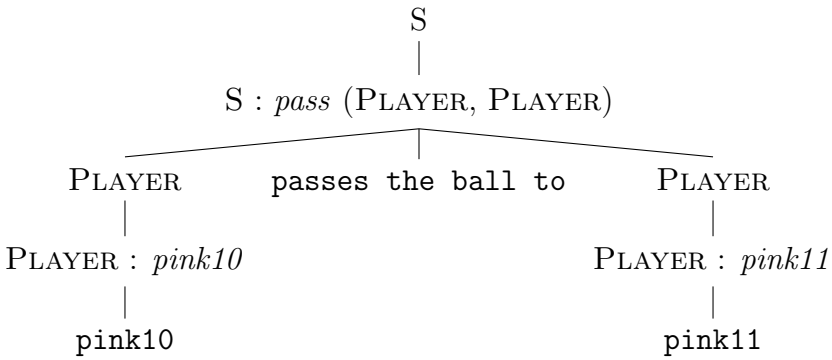


Figure 2.1: Sample hybrid tree of NL/MR pair from the English sportscasting dataset: PINK10 PASSES THE BALL TO PINK11 / *pass*(*pink10*, *pink11*)

and child semantic categories. This process continues until all the leaves in the hybrid tree become NL words. The generation assumes a Markov process, implying that each step is only dependent on its parent step.

Lu et al.'s (2008) generative parsing model estimates the joint probability $P(\mathcal{T}, \mathbf{w}, \mathbf{m})$, the probability of generating a hybrid tree $\mathcal{T}$ with NL $\mathbf{w}$, and MR $\mathbf{m}$. This probability is calculated by the whole product of the probabilities of all the generation steps in the tree. The data likelihood of the pair $(\mathbf{w}, \mathbf{m})$ given by the learned model becomes the sum of $P(\mathcal{T}, \mathbf{w}, \mathbf{m})$ over all the possible hybrid trees, because a hybrid tree for an NL $\mathbf{w}$ and an MR $\mathbf{m}$ is not unique.

The model runs in conventional, fully supervised settings. In order to learn from ambiguous supervision, we extend this model to include an additional generative process for selecting the subset of available MRs used to generate NL sentences. Our model, thus, has a capability of performing align-

ment of correct NL–MR pairs at the same time as finding the correct MR for a novel NL sentence.

2.1.2 WASP and WASP^{-1}

WASP (Word-Alignment-based Semantic Parsing) (Wong & Mooney, 2006) is a semantic parsing system that uses statistical machine translation (SMT) techniques (Brown, Cocke, Della Pietra, Della Pietra, Jelinek, Lafferty, Mercer, & Roossin, 1990; Yamada & Knight, 2001; Chiang, 2005) in order to learn semantic parsers. SMT methods learn machine translation models that are trained on parallel corpora composed of one-to-one human-annotated translations of two or more natural languages. SMT techniques have been shown to be very effective compared to other previous hand-engineered approaches, and have become dominant in recent decades. The main idea of WASP is to utilize such SMT techniques in order to translate between natural language (NL) and logical meaning representation language (MRL) instead of translating between two different NLs.

WASP is mainly composed of two stages. First, GIZA++ (Och & Ney, 2003; Brown, Della Pietra, Della Pietra, & Mercer, 1993), a statistical word alignment model, constructs a bilingual (NL–MR) lexicon that finds the most probable relationship between NL phrases and MRL grammar rules. Certain MRL tokens, such as parentheses, commas, or colons, are inherently meaningless, and are used in order to maintain and visualize the syntactic formalism of the MRL. Thus, it is not an ideal way to find direct co-occurrence relationships

between NL phrases and MR component tokens. Instead, the Wasp system uses GIZA++ to construct mappings between NL phrases and MRL grammar production rules used.

Then, the WASP system induces a probabilistic synchronous context-free grammar (SCFG) for generating corresponding NL–MR pairs (Aho & Ullman, 1972). An SCFG rule contains two strings on its right-hand side, which represents the fact that two language components are generated by a single rule at the same time, in our case, one in NL and the other in MRL. The bilingual lexicon obtained is used to construct a set of SCFG production rules. NL sentences and corresponding MRs are simultaneously generated from the SCFG derivations. Then, a maximum-entropy model is trained to produce a probabilistic parser to learn the weights of SCFG rules. When parsing a new NL sentence into the desired MR, the most probable SCFG derivation is obtained by a probabilistic chart parser, which subsequently generates the corresponding MR.

SCFG is symmetric with respect to the two languages it generates, and thus the same trained model can be used for both semantic parsing (translating NL to MR) and natural language generation (translating MR to NL) tasks by reversing the direction of input and output languages. A noisy-channel model (Brown et al., 1990) is used for the natural language generation system, WASP^{-1} (Wong & Mooney, 2007a), which learns an n-gram language model for the NL part to play the role of formal grammar for the NL side.

2.2 Learning from Ambiguous Supervision

The 1-to-1 supervision of NL–MR annotations for semantic parsing tasks requires too much human intervention and leads to a very high cost for constructing a corpus. However, we can obtain a training corpus much more easily when we extract the surrounding perceptual context as MRs along with a given NL sentence. Inevitably, this kind of data will have 1-to-N ambiguous supervision, which needs to be resolved first in order to learn a correct semantic parsing model. Our proposed approaches tackle this main challenge of ambiguous supervision. Our model in Chapter 3 uses a previous content selection model for initializing its parameters. In addition, our model in Chapter 4 extends a previous approach, a PCFG induction model that tackles ambiguous learning problems so that our model is able to learn accurate semantic parsers from a much higher level of ambiguous supervision.

2.2.1 Iterative Generation Strategy Learning (IGSL)

Chen and Mooney (2008) introduced Iterative Generation Strategy Learning (IGSL) for determining which event types a human commentator is more likely to describe in natural language. This is sometimes called *strategic generation*, or *content selection*, a process of choosing *what to say*; as opposed to *tactical generation*, which determines *how to say it*. IGSL uses a method analogous to EM to train on ambiguously supervised data and iteratively improves probability estimates of commenting on each event type, specifying how likely each MR predicate is to elicit an NL comment.

Every event type is initialized with uniform probability counts. Then, the IGSL algorithm alternates between two processes:

1. Calculating the expected probability of an NL–MR matching based on the currently learned probability estimates.

2. Updating the probability of each event type based on the expected match counts.

IGSL has been shown to be quite effective at predicting which events in a RoboCup game a human would comment upon. In our model in Chapter 3, we use IGSL probability scores as initial priors for our event selection model.

2.2.2 Unsupervised PCFG Induction for Grounded Language Learning

Börschinger et al. (2011) introduced an unsupervised PCFG induction model for grounded language learning. It automatically constructs a PCFG that generates natural language (NL) sentences from formal meaning representations (MRs). The nonterminals in the grammar correspond to complete MRs and MR constituents, while NL phrases and words are expressed as terminals. The generative process of PCFG describes how a composite MR generates its MR constituents. Then, each constituent eventually generates NL words.

First, the nonterminal for a composite MR generates each of its MR constituents. Since we do not know the order in which each constituent will generate NL words, every possible permutation of the constituents must be

included in order to consider all the possibilities. Second, the nonterminal for an MR constituent generates $Phrase_x$, representing a sequence of NL words connected to the constituent x. $Phrase_x$ is then used to generate a sequence of $Word_x$, which subsequently produces NL words, which simulates a unigram Markov process of generating multiple NL words from $Phrase_x$. By training the Inside-Outside algorithm on the produced PCFG rules, the system learns the probabilistic relationships between NL words, MR constituents, and complete MRs by getting the most probable weights for the rules. Figure 2.2 shows a derivation tree of this framework for a sample NL–MR pair and the PCFG rules that are constructed for it. When parsing a novel sentence into the most probable parse tree using a probabilistic chart parser, we are able to get the most likely MR interpretation for a given NL sentence by reading the top nonterminal containing the full MR.

However, this approach has several clear limitations. First, it only works for finite MR languages, and the produced PCFG becomes intractably large even for finite but moderately complex MRs. The main reason stems from the assumption that the overall structure of the MRL is simple enough so that every constituent of an MR and all its possible permutations can be encoded in a reasonable number of PCFG rules for building correspondences with NL. This is not true in more general situations where the MRs represent a wider range of surrounding perceptual environments that may often over-describe the NL. In addition, this approach assumes every MR component is responsible for having a semantic connection with at least one or more NL words. In addition,

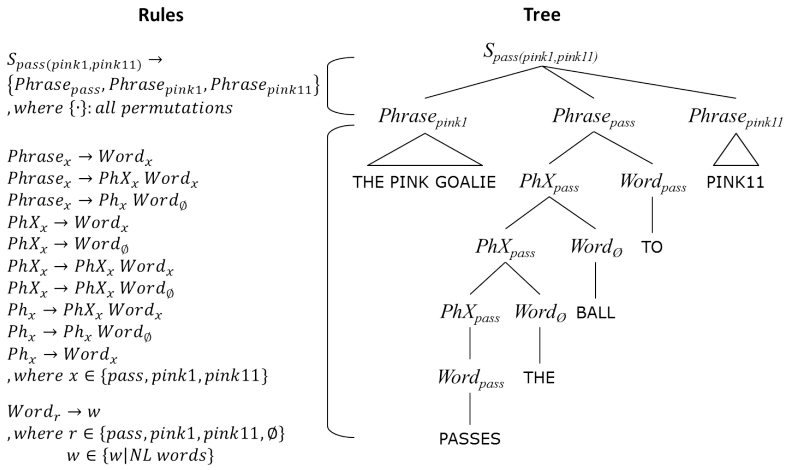


Figure 2.2: Derivation tree for the NL/MR pair: THE PINK GOALIE PASSES THE BALL TO PINK11 / $pass(pink1, pink11)$. The left side shows PCFG rules that are added for each stage (complete MR to MR constituents and subsequently generated NL words).

all permutation orders of MR components need to be considered because we do not know in advance which components are connected to which parts of NL sentences. If the number of constituents per MR increases, the resulting PCFG size can increase exponentially. In Chapter 4, we present two enhanced models that extend this current approach by incorporating a learned semantic lexicon to build direct correspondences between NL words and semantic lexeme MRs (constituting semantic concepts). This results in the smallest semantic unit being semantic lexeme MRs. Therefore, our approach constrains the space of productions and thereby makes the number of production rules tractable for complex MRs, and even has the capability of handling MR grammars that define an infinite language. Another limitation of Börschinger et al.'s (2011) approach stems from parsing novel NL sentences into desired MR outputs.

This approach is only able to produce MRs previously seen during training as the parse result of a novel NL sentence, because the parsing step only requires reading off the top nonterminal of the PCFG parse tree. In contrast, our approach has the ability to produce novel MRs when parsing test sentences by composing related semantic lexeme MRs. This makes our approach work well with a much higher level of ambiguous supervision where an NL sentence refers only to some subset of a matching MR representation, which implies an exponential number of possibilities for the true matching.

2.3 Discriminative Reranking

Discriminative reranking is a common machine learning technique to improve the results of generative models. It has been shown to be effective for various natural language processing tasks including syntactic parsing (Collins, 2000, 2002b; Collins & Koo, 2005; Charniak & Johnson, 2005; Huang, 2008), semantic parsing (Lu et al., 2008; Ge & Mooney, 2006), part-of-speech tagging (Collins, 2002a), semantic role labeling (Toutanova et al., 2005), named entity recognition (Collins, 2002c), machine translation (Shen, Sarkar, & Och, 2004; Fraser & Marcu, 2006) and surface realization in language generation (White & Rajkumar, 2009; Konstas & Lapata, 2012).

In order to enhance the performances of a generative model, a secondary conditional model is trained on the k-best candidate outputs obtained from the baseline generative model with a gold-standard interpretation provided for each training example. The conditional model evaluates and compares the

quality of candidates against the gold-standard during the training phase, and thus finally optimizes the parameters to rerank novel candidates from test data. By using global features of candidate interpretations, the trained discriminative reranker can significantly improve the accuracy of the baseline generative model. For our experiments, we use an averaged perceptron (Collins, 2000), which has been shown to be effective in a wide range of previous natural language processing research.

Although such conventional discriminative reranking approaches require the gold-standard interpretations for training, typical grounded language learning problems are not equipped with a single gold-standard for each training example. To our knowledge, there has been no previous attempt to apply discriminative reranking to grounded language learning problems. In Chapter 5, we describe how discriminative reranking can be adapted to solve grounded language learning problems without gold-standards, particularly using weak supervision of evaluation feedback of candidate outputs against the perceptual world.

Chapter 3

Generative Alignment and Semantic Parsing for Limited Ambiguity

In this chapter, we present a probabilistic generative model for learning semantic parsers trained on supervision of limited ambiguity where a NL sentence is paired with multiple candidates of logical MRs naturally obtained from the surrounding world state (Kim & Mooney, 2010). This model disambiguates the underlying meaning of each sentence while simultaneously learning a semantic parser that maps NL sentences into MR logical forms. Our method is evaluated on the previously introduced problem of RoboCup Sportscasting (Chen & Mooney, 2008; Chen et al., 2010). Compared to the approaches of Chen and Mooney (2008) and Chen et al. (2010), our model produces successful and more effective matching disambiguation and semantic parsing from the ambiguous training corpus whose parameters are estimated by a fully probabilistic model. In addition, in contrast to a previous generative model for semantic alignment by Liang et al. (2009), it also supports full semantic parsing. Experimental results on the sportscasting corpora in both English and Korean indicate that our approach produces more accurate semantic alignments than existing methods and also produces competitive semantic parsers and improved natural language generators.

3.1 Chapter Overview

Chen and Mooney (2008) first introduced the problem of learning to sportscast by simply observing natural language commentary on simulated RoboCup robot soccer games. However, the 1-to-N NL–MR ambiguity of the training data caused by the manner in which the data are collected poses a serious challenge to learning accurate semantic parsers or language generators. We first need to learn the correct semantic alignment between NL and MR, since the correct alignment of the training data is unknown.

The original approach of Chen and Mooney (2008) to this task retrains existing supervised semantic parser learners iteratively in a manner similar to EM training on the disambiguated NL–MR training example pairs produced by the previous iteration. However, it suffers from possible information loss since it does not run on a well-defined probabilistic model. On the other hand, Liang et al. (2009) proposed a probabilistic generative alignment model for ambiguous supervision. Despite its improved performance, the model is only capable of semantic alignment between NL–MR and does *not* learn either a semantic parser or a language generator. In addition, Liang et al. assume a bag-of-words model for natural languages and do not incorporate linguistic syntax which includes additional cues to be exploited.

Our generative model overcomes some of the limitations of these previous methods and provides simultaneous semantic alignment and semantic parsing for ambiguous supervision using the Hybrid tree model proposed by Lu et al. (2008), which generates NL and MR components in a single tree

structure. Experimental results on the sportscasting data show that our approach outperforms all the previous results on the NL–MR matching (semantic alignment) and language generation task and also achieves competitive performance on the semantic parsing task.

3.2 Sportscasting Data and Task

The RoboCup sportscasting data (Chen & Mooney, 2008; Chen et al., 2010) were collected by asking humans to commentate the four final games (2001 to 2004) of the RoboCup simulation soccer league. Commentaries were collected in both English and Korean by the corresponding native speakers of the languages. Table 3.1 shows overall statistics about the data. The detailed statistics for each game appear in Table A.1 of Appendix A. The human commentators only saw the visual progress of the soccer games and were provided with an annotation tool to record their NL commentary along with timestamps. On the other hand, game events are collected by a rule-based system that automatically extracts them from the simulator traces. The extracted events mainly involve actions with the ball, such as kicking, passing, turnovers, or goals, but also include other game information describing current game status, such as kickoffs, offsides, or corner kicks. The events are represented as atomic formulas in predicate logic recorded with timestamps. These logical components constitute full meaning representations consisting of one predicate and up to two arguments that most commonly represent players.

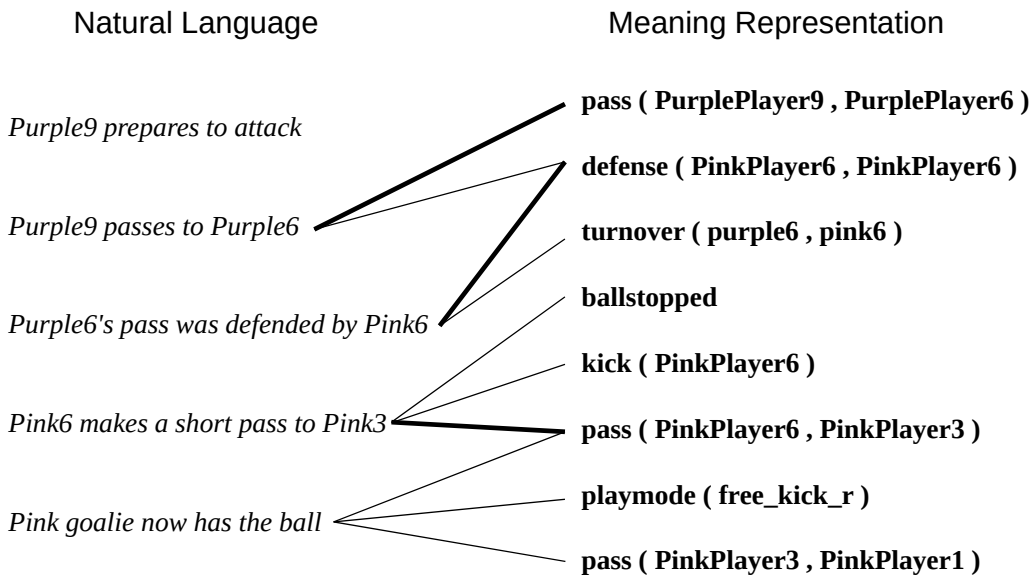
Since NL and MR data are collected separately, there are only weak

	English	Korean
# of NL comments	2036	1999
# of words	11742	7941
Average words per NL comment	5.77	3.97
# of extracted MR events	10657	10657
# of NLs with matching MRs	1868	1913
# of MRs with matching NLs	4670	4610
Average number of MRs per NL	2.50	2.41

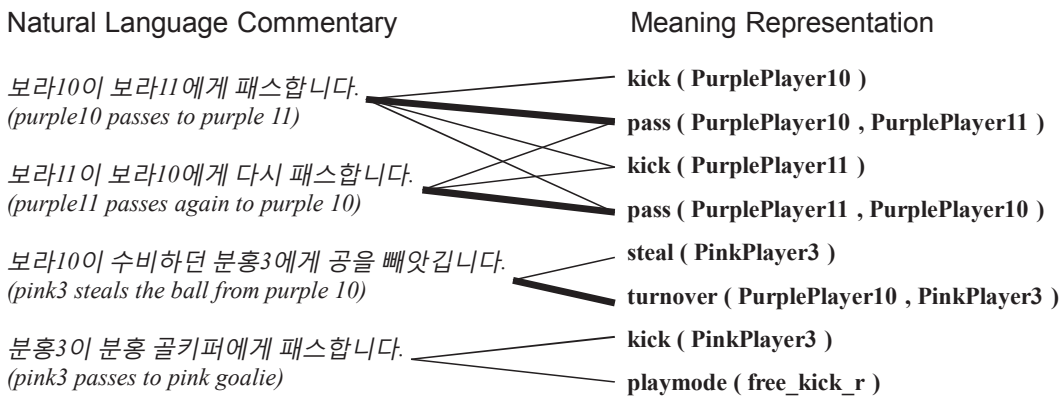
Table 3.1: Statistics for RoboCup sportscasting data

connections between the two. The only assumption we can make is that each NL commentary has the possibility of connecting with MR events that occur close together based on the timestamps. Since sportscasting commentaries are made after the events, each NL commentary sentence is paired with automatically extracted MRs of ongoing simulation events that occurred in the previous 5 seconds (an average of 2.5 events).

Figure 3.1 shows a sample trace from the English and Korean data. As shown, each NL commentary sentence normally has several candidate MR matches that occurred within the 5-second window, indicated by the edges between NL and MR. Bold edges denote gold-standard alignment manually constructed solely for evaluation purposes. It is not guaranteed, however, that every NL has a correct matching MR, because sometimes there are unrecognized or undetected events and sometimes there are NL commentaries that describe high-level concepts about the game (e.g., the pink team is sloppy today) that cannot be normally captured by automatically extracted MR events. Such ambiguity brings the additional challenge that a given NL sentence may



(a) Sample trace of ambiguous English training data



(b) Sample trace of ambiguous Korean training data

Figure 3.1: Sample traces of sportscasting data. Each outgoing edge from the NL comments indicates that the comment and connected meaning representations are possible translations of each other. The bold links indicate correct matches between the comments and the meaning representations. Note that some NL commentaries do not have correct matching MRs.

not have a correct matching MR.

3.3 Generative Model for Semantic Alignment and Language Grounding

Our model is based Lu et al.'s (2008) generative semantic parsing model using a hybrid-tree framework, to which we have added the capability of selecting which MR out of all the candidates, as described below. A *hybrid tree* is defined over a pair of an NL sentence and a complete MR $(\mathbf{w}, \mathbf{m})$ to describe hierarchical correspondences between each NL word and MR components. A hybrid tree constitutes a generative process of how NL words are produced along with MR production rule structure. In contrast, our model estimates $P(\mathbf{w}|\mathbf{s})$, where $\mathbf{w}$ is an NL sentence and $\mathbf{s}$ is a world state consisting of several candidate MRs matched to $\mathbf{w}$. In this setting, our approach is intended to support both determining the most likely match between an NL and its MR, **and** semantic parsing, that is, finding the most probable mapping from a given NL sentence to an MR logical form.

Our generative model consists of two stages:

- Event selection: $P(\mathbf{e}|\mathbf{s})$ chooses the event $\mathbf{e}$ in the world state $\mathbf{s}$ to be described.
- Natural language generation: $P(\mathbf{w}|\mathbf{e})$ models the probability of generating natural-language sentence $\mathbf{w}$ from the MR specified by event $\mathbf{e}$.

A sample generative process is shown in Figure 3.2.

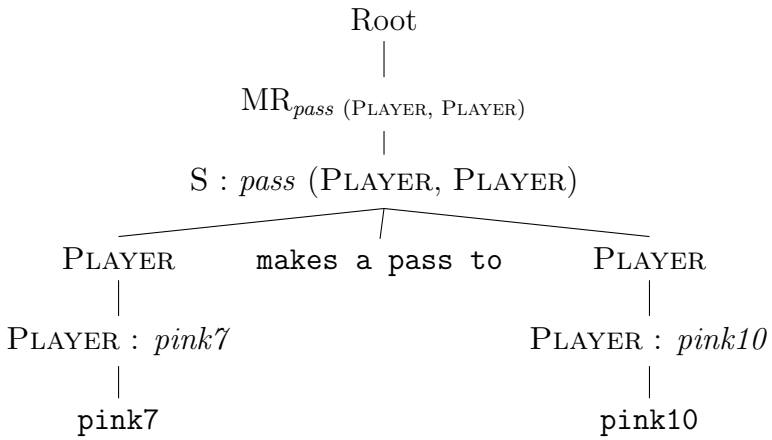


Figure 3.2: Sample generative process of our model from the root nonterminal to the selected MR, and finally to the hybrid tree. The NL is PINK7 MAKES A PASS TO PINK10, and the chosen MR is *pass*(*pink7*, *pink10*) out of multiple potential MRs. Note that there is an additional layer of selecting MR *pass*(*pink7*, *pink10*) in order to generate the corresponding hybrid tree.

3.3.1 Event Selection

The event selection model specifies the probability distribution for picking an event that is likely to be commented upon among the multiple candidate MRs appearing in the world state $\mathbf{s}$. The probability of selecting an event is assumed to depend only on its event type as given by the predicate of its MR. For example, the MR $pass(pink10, pink11)$ has the event type *pass* and arguments $pink10$ and $pink11$. The probability of picking an event e of type t_e is $p(t_e)$. If there are multiple type t events in $\mathbf{s}$, then a type t event is selected uniformly from the set $\mathbf{s}(t)$ of events of type t in $\mathbf{s}$. Thus, the probability of picking an event is given by the following:

$$P(\mathbf{e}|\mathbf{s}) = p(t_e)\frac{1}{|\mathbf{s}(t_e)|} \tag{3.1}$$

This model is similar to the *record choice* model of Liang et al. (2009), but it only models *salience* to the extent that some event types are more likely than others. Our model does not consider the order of event types (*coherence*) because the RoboCup sportscasting data only have at most one true MR for a given NL sentence.

3.3.2 Natural Language Generation

The natural-language generation model defines the probability distribution of NL sentences given an MR specified by the previously selected event in the event selection model. We use Lu et al.'s (2008) generative semantic parsing model for this step:

$$P(\mathbf{w}|\mathbf{e}) = \sum_{\forall \mathcal{T} \ over \ (\mathbf{w},\mathbf{m})} P(\mathcal{T}, \mathbf{w}|\mathbf{m}) \tag{3.2}$$

where $\mathbf{m}$ is the MR defined by event $\mathbf{e}$ and $\mathcal{T}$ is a hybrid tree defined over the NL–MR pair $(\mathbf{w}, \mathbf{m})$.

The probability $P(\mathcal{T}, \mathbf{w}|\mathbf{m})$ is given by the generative semantic parsing model (Lu et al., 2008) with the inside probability of the NL–MR pair $(\mathbf{w}, \mathbf{m})$. The likelihood of a sentence $\mathbf{w}$ is then the sum over all possible hybrid trees defined by the NL–MR pair $(\mathbf{w}, \mathbf{m})$. Out of Lu et al.'s three proposed models (unigram, bigram, and mixgram), we used the bigram model, which estimates its inside probability by checking whether an NL word or a semantic category is

dependent upon the previously generated one. In our experiments, the bigram model always performed the best on all tasks.

The natural language generation model replaces the role of the *field choice* model and *word choice* model of Liang et al. (2009) in semantic alignment tasks. It also considers the order of predicates and arguments in an MR as well as the orderly generation of NL words and phrases, since the model constructs an ordered hybrid tree structure that generates NL words, MR semantic categories, and MR grammar rules.

3.4 Learning

Standard EM methods are used to train our generative model. The process is similar to that used by Lu et al. (2008), an inside-outside style algorithm that generates a hybrid tree from the NL–MR pair $(\mathbf{w}, \mathbf{m})$, but our model additionally considers expected counts under the posterior $P(\mathbf{e}|\mathbf{w}, \mathbf{s}; \theta)$ in the E-step and normalizes the counts in the M-step. Training time takes about 30 minutes for sportscasts of three training games in either the English or Korean dataset.

However, the experiments show that the EM method tends to fall into local optima when estimating the event-type selection probabilities, $p(t)$, thus hurting the overall performance. To resolve this issue, we initialized the parameters of our model with the corresponding strategic generation values learned by the IGSL algorithm (Chen & Mooney, 2008). IGSL priors serve as a good starting point for training EM, particularly for our event selection model.

IGSL has already been shown to be very effective at predicting which event types are likely to be described in sportscasting data.

The generative semantic parsing model of Lu et al. (2008) is trained through several stages to provide the best performing results. The bigram model we used in our model was trained on the basis of parameters previously learned for the IBM Model 1 (Brown et al., 1993) and the unigram model. We followed a similar multi-stage learning strategy. Our best model, which uses the bigram model, was trained on the previously learned parameters from our model with the IBM Model 1 and the unigram model. The multiple learning stages led to the model's being vulnerable to getting stuck in local optima when running EM across these multiple steps. We also tried using random restarts with several initializations, but IGSL priors provided the best results in the evaluations.

3.5 Experimental Evaluation

For evaluation, we followed the same evaluation schemes as in Chen and Mooney (2008) covering three tasks: NL–MR matching (semantic alignment), semantic parsing, and natural language generation (surface realization). RoboCup sportscasting data contains 4 separate games, and we performed leave-one-game-out (4-fold) cross validations using 3 games for training and the remaining 1 game for testing to evaluate semantic parsing and natural language generation. Since the matching (semantic alignment) task is essentially that of disambiguating the training data, the performance of matching

Systems	Matching	Semantic Parsing	Language Generation	Ambiguous Training
Our model	√	√	-	√
Liang et al. (2009)	√	-	-	√
Lu et al. (2008)	-	√	-	-
WASP^{-1} (Wong & Mooney, 2007a)	-	√	√	-

Table 3.2: Overview of various systems and models used in the experiments. Each column indicates the capability on various tasks.

is evaluated on the training data.

The accuracy of matching and semantic parsing is measured using the F-measure, which is the harmonic mean of precision and recall. We evaluated the natural language generation using the BLEU score (Papineni, Roukos, Ward, & Zhu, 2002) between the generated sentences and the reference NL sentences in the test set. Systems we compared include those of Chen and Mooney (2008) and Chen et al. (2010), and that of Liang et al. (2009) for the semantic alignment results only.

In Table 3.2, we present the various systems and models used in the experimental evaluations and their capabilities for the tasks. Our model is capable of learning semantic parsers while disambiguating the correct matching of training data.

	English	Korean
Chen and Mooney (2008)	0.681	0.753
Liang et al. (2009)	0.757	0.694
Chen et al. (2010)	0.793	0.841
Our model	0.832	0.800
Our model with IGSL prior initializations	**0.885**	**0.895**

Table 3.3: NL–MR Matching Results (F-measure).

3.5.1 NL–MR Matching (Semantic Alignment)

The matching or semantic alignment task measures how well the system finds the correct NL–MR alignment out of ambiguous examples consisting of an NL sentence and multiple potential MRs. As described above, training examples in the RoboCup sportscasting data have up to one correct matching. Our model outputs the most probable matching as an NL **w** and an MR **m** if and only if **m** is the most probable parse of **w** according to the learned semantic parser. Thus, our model does not force every NL to match to an MR. Some NL sentences whose most probable parse is not one of the candidate MRs are left unmatched. Matching output is evaluated against the manually constructed gold-standard matches, which are never used during training.

Evaluation results for the English and Korean datasets are shown in Table 3.3. Since the Korean data were not yet available for use by either Chen and Mooney (2008) or Liang et al. (2009), we cited those results from Chen et al. (2010). Our best approach outperforms all previous methods by large margins when using IGSL priors. In particular, our model also outperforms the generative alignment model of Liang et al. (2009), implying that the extra

	English	Korean
Chen and Mooney (2008)	0.702	0.720
Chen et al. (2010)	0.803	**0.812**
Our learned parser	0.742	0.764
Lu et al. (2008) initialized with our model's matching	**0.810**	0.794
Lu et al. (2008) initialized with Liang et al. (2009)	0.790	0.690
WASP initialized with our model's matching	0.786	0.808
WASP initialized with Liang et al. (2009)	0.803	0.740

Table 3.4: Semantic Parsing Results (F-measure).

linguistic information and MR grammatical structure result in a more effective model than a Markov model with a bag-of-words model.

3.5.2 Semantic Parsing

Semantic parsing is evaluated by how accurately the systems map novel NL sentences into their proper corresponding MRs in the test data. Table 3.4 presents the results. We compare to the best results presented in the cited papers: WASPER-GEN for Chen and Mooney (2008), WASPER with Liang et al.'s (2009) matching initialization for English and WASPER-GEN-IGSL-METEOR with Liang et al.'s initialization for Korean for Chen et al. (2010). Semantic parsing results with our directly learned parser from the ambiguous training data are presented, as well as supervised parsers (both WASP and Lu et al.'s) trained on the NL–MR matching output by our model. All our semantic parsing results used IGSL initialization, which resulted in the best performances. For additional comparisons, Lu et al.'s parser and WASP trained on Liang et al.'s NL–MR matchings are also shown.

Our initial learned semantic parser performs better than that of Chen and Mooney (2008), but worse than that of Chen et al. (2010). Training WASP and Lu et al.'s (2008) parsers on our highly accurate NL–MR matchings improved the results over Liang et al.'s (2009) matchings. It is also noteworthy that retraining on the hardened one-to-one supervision of the most probable NL–MR matches gives better performance than the parser directly trained using EM. The uncertainty caused by incorrect NL–MR matchings resided as probabilistic counts in our generative model seems to affect the overall parsing performance.

Comparing with the corresponding results for training WASP and Lu et al.'s (2008) supervised parser on the NL–MR matchings produced by Liang et al.'s (2009) alignment method, it is clear that our matchings produce more accurate semantic parsers except when training WASP in English. This result means that the improved matching leads to a better semantic parsing system in general.

3.5.3 Natural Language Generation (Surface Realization)

The natural language generation (tactical generation) or surface realization task evaluates how well a system generates accurate NL sentences from novel test MRs. Since our semantic parsing model does not support natural language generation, which is the reverse task of semantic parsing, we trained the publicly available WASP^{-1} system (Wong & Mooney, 2007a) on our disambiguated NL–MR matches. Since we were using WASP^{-1}, we can directly

	English	Korean
Chen and Mooney (2008)	0.4560	0.5575
Chen et al. (2010)	0.4599	0.6796
WASP^{-1} trained on matching of Liang et al. (2009)	0.4580	0.5828
WASP^{-1} trained on our matching outputs	**0.4727**	**0.7148**

Table 3.5: Natural language generation (surface realization) results (BLEU score).

compare our results with those of Chen and Mooney (2008) and Chen et al. (2010).

Table 3.5 shows the natural language generation results of our model and the best reported results from the cited papers: WASPER-GEN for Chen and Mooney (2008), WASPER trained with Liang et al.'s (2009) matching for the English results of Chen et al. (2010), and WASPER-GEN with Liang et al.'s initialization for the Korean dataset. In this experiment, our generation results are also based on our best matching results with IGSL initialization, which provides the best results overall. WASP^{-1} trained on our NL–MR matching results performed the best. It should also be noted that WASP^{-1} trained with our matchings performs better than WASP^{-1} trained with Liang et al.'s matchings.

3.6 Discussion

Overall, our model performs particularly well at the matching task. However, improved matching does not transfer to notably better semantic parsing results, seeing as there is a 10% improvement for matching compared

to a 1-point improvement on the semantic parsing task.

This seems to be due to the nature of the noise in the matching results. Although Liang et al.'s (2009) alignment model gives a much lower F-measure, it provides cleaner matching and contains fewer noisy, misleading NL–MR pairs. On the other hand, even though our model performs much better in matching, it predicts some misleading matches when the gold-standard match does not exist, inevitably resulting in worse semantic parsers due to the noisy probabilistic counts coming from such training pairs. For example, an NL sentence `pinkG intercepts` is matched to an MR *ballstopped* due to incomplete event detection, and this NL sentence does not actually have a correct match in the gold-standard. This sentence is covered by our model, and it becomes harder for the semantic parser to learn to map to the true MR for this sentence, *block(pink1)*. By contrast, Liang et al.'s (2009) model does not try to match this sentence to any MR, which leads to less noise when training the semantic parser.

Compared to the model of Liang et al. (2009), our more accurate matchings provide a clear improvement in both semantic parsing and natural language generation, although the improvement in semantic parsing is not dramatic. The only exception is semantic parsing in the English data using WASP, which seems to be due to some misleading noise in our alignments explained above. WASP seems to be affected more than Lu et al.'s (2008) system by such extraneous noise. However, in natural language generation, this extraneous noise does not lead to worse performance, and our approach always gives

the best results. As discussed by Chen and Mooney (2008) and Chen et al. (2010), this difference seems to stem from natural language generation's being somewhat easier than semantic parsing in the sense that semantic parsing needs to learn to map a variety of synonymous NL sentences to the same MR, whereas surface realization only needs to learn one way to produce a correct NL description of an MR.

3.7 Chapter Summary

In this chapter, we have presented a generative model capable of probabilistically aligning natural-language sentences to their correct meaning representations given the ambiguous supervision provided by a grounded language learning scenario. Our model is also capable of simultaneously learning to semantically parse NL sentences into their corresponding meaning representations. Experimental results in the RoboCup sportscasting domain show that the NL–MR matchings inferred by our model are significantly more accurate than the results produced by all previous methods using the same data. Our approach also learns competitive semantic parsers and improved language generators compared to previous methods. Specifically, we showed that our alignments provide a better foundation for learning accurate semantic parsers and tactical generators than those of Liang et al. (2009), whose generative model is limited by a simple bag-of-words assumption and does not utilize any linguistic syntax structure.

Chapter 4

Unsupervised PCFG Induction for Grounded Language Learning with High Ambiguity

In the previous chapter, we reviewed the RoboCup sportscasting task, where the training data contain pairs of an NL sentence with a handful of possible MRs. Our proposed generative model showed that it is capable of disambiguating the 1-to-N potential matchings and simultaneously parsing natural language (NL) sentences to proper meaning representations (MRs). In this chapter, we will deal with more complex ambiguity. The navigation task (Chen & Mooney, 2011) contains training data with a large set of potential meanings for each sentence, where only a subset of meaning components are relevant. To solve this task, we will present our enhanced model based on Börschinger et al. (2011), which learns a semantic parser on ambiguous supervision by transforming grounded language learning into unsupervised *probabilistic context-free grammar* (PCFG) induction. Their model works well in the sportscasting task, where there is limited ambiguity, but it cannot be generally applied to more complex problems with higher ambiguity. Our novel enhancement uses a semantic lexicon as the basic unit to make correspondences between NL substrings and MR components, which also allows handling highly ambiguous situations without additional computational complexity. Experi-

mental results on the navigation task demonstrate the effectiveness of our approach.

4.1 Chapter Overview

First, we discuss our unsupervised PCFG induction models for learning the semantics of language when the training data is highly ambiguous (Kim & Mooney, 2012). In particular, we focus on the navigation task (Chen & Mooney, 2011) where the goal is to interpret natural language instructions in virtual environments so that an agent can perform the desired actions. The navigation task requires the system to disambiguate the training data in which each instructional sentence is paired with a formal *landmarks plan* (represented in a large graph structure) that includes a full description of the observed actions and world-states that are encountered while following the instruction. The major challenge stems from the fact that the NL instruction refers to only a subgraph of the formal landmarks plan. This inevitably leads to a combinatorial number of possible meanings when finding a true match for a given sentence.

To resolve this problem, we present two versions of novel enhancements of the unsupervised PCFG induction method for grounded language learning introduced by Börschinger et al. (2011). Börschinger et al.'s approach works for limited ambiguity settings where there is up to one true meaning out of a small set of contextual meanings for an NL sentence, such as the sportscasting task. Their approach first constructs a large set of production rules from

the ambiguous training set of an NL sentence paired with multiple MRs, and then optimizes the weights of the PCFG grammar using EM. Parsing a novel sentence with this learned grammar produces a parse tree containing the formal MR parse in the top nonterminal. Although this approach is effective for simple ambiguous supervision such as the sportscasting data, applying it to problems with highly ambiguous supervision, such as the navigation task, leads to a prohibitively large number of PCFG production rules. For instance, there are a number of training examples in the navigation data containing more than 20 actions for a single NL instruction sentence, which produces more than 20! ($> 10^{18}$) PCFG production rules to train on, considering that the model should produce at least every permutation of actions encoded in PCFG rules.

To overcome this difficulty, our approaches enhance Börschinger et al.'s (2011) model by using semantic lexemes as the basic building block when constructing PCFG production rules. Whereas Börschinger et al. used each MR constituent to generate NL words probabilistically, our approaches use lexemes (pairs of an MR graph and an NL phrase) which form meaningful semantic concepts. The advantage of this enhancement is that we directly connect semantic concepts to corresponding NL words during training.

Our first approach builds upon the intuition that the semantic concepts represented by lexeme MRs form hierarchical structures analogous to the syntactic hierarchy in syntax parsing (Kim & Mooney, 2012). Even though this approach outperforms previous methods in the same corpus, the performance

is still limited due to the additional complexity caused by the hierarchy of lexemes and the inevitable permutation rules introduced. Our second approach takes a simpler method without using the lexeme hierarchy. Instead, the new approach generates relevant lexemes by a unigram Markov process so that the permutation rules are not necessary. Further, the generated lexemes are probabilistically matched to NL substrings to complete language groundings.

Our two approaches are able to solve some of the limitations of Börschinger et al.'s (2011) model in that the number of PCFG production rules remains tractable, since semantic lexemes as basic units encode MRs in compact representations for complicated MR languages. Moreover, our models can also produce novel final MR parses that were never seen during training, whereas Börschinger et al.'s model cannot.

We describe our two PCFG approaches in Sections 4.3 and 4.4. The experimental results show that our two approaches perform better than the previous methods, and our second, and simpler, approach achieves even better performance than our hierarchy generation approach while also reducing training time.

4.2 Navigation Task and Dataset

To evaluate our model, we employ the task and data introduced by Chen and Mooney (2011), where the goal is to interpret and follow NL navigation instructions in a virtual world by simply observing how humans follow them. Figure 4.1 shows a sample execution path in a particular virtual

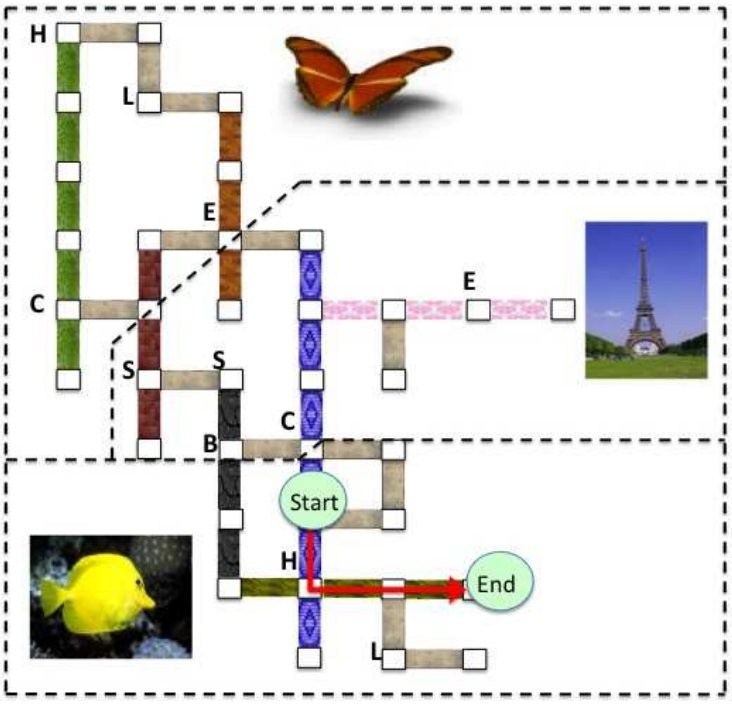


Figure 4.1: Sample virtual world from Chen and Mooney (2011) of interconnecting hallways with different floor and wall patterns and objects indicated by letters (e.g., "H" for hatrack).

world. In other words, the goal of the task is to train a system that converts instructions into runnable MR navigation plans and execute them in the virtual environment. Formally speaking, given the training data of the form $\{(e_1, a_1, w_1), \ldots, (e_n, a_n, w_n)\}$, where e_i is an NL instruction, a_i is an observed action sequence, and w_i is the current world state (describing patterns of floors and walls, positions of any objects, etc.), we want to produce the correct actions a_j for a novel (e_j, w_j).

In order to learn, the task requires us to infer the intended formal plan p_i (the MR for a sentence in this task) that produced the action sequence a_i

from the instruction e_i. However, there is a large number of possibilities when choosing a formal plan for any given action sequence. For a simple example, there are several ways to describe the actions of going two steps toward a sofa and then turning right. In a straightforward manner, we can describe the actions as `Travel`(`steps` : `2`), `Turn`(`RIGHT`) by just describing the atomic actions, or `Travel`(), `Verify`(`front` : `SOFA`), `Turn`(`RIGHT`) using notable objects along the way. Chen and Mooney (2011) called the former a *basic plan* and the latter a *landmarks plan*. They focused more on the landmarks plan since it carries more information for understanding the semantics of instructions and is closer to how humans actually describe navigational directions in the real world. Also, they showed that landmarks plans led to better overall performance evaluation with their proposed system.

Their system first constructs a formal landmarks plan, c_i, for each a_i, which is a graph representing the context consisting of a full description of every action in the sequence and the world-state that is encountered while following the actions. The hard part is that the correct plan MR, p_i, is assumed to be a subgraph of c_i, which implies that there is an exponential number of possibilities to choose a correct MR from. The landmarks and correct plans for a sample instruction are shown in Figure 4.2.

To circumvent this combinatorial problem, Chen and Mooney (2011) never explicitly enumerate the combinatorial possibilities of potential meanings for each sentence. Instead, their system first learns a semantic lexicon that maps NL words and short phrases to small MRs (subgraphs) formally repre-

Instruction: "at the easel, go left and then take a right onto the blue path at the corner"

Landmarks plan:
Travel (steps: 1) ,
Verify (**at: EASEL** , side: CONCRETE HALLWAY) ,
Turn (LEFT) ,
Verify (front: CONCRETE HALLWAY) ,
Travel (steps: 1) ,
Verify (side: BLUE HALLWAY , **front: WALL**) ,
Turn (RIGHT) ,
Verify (back: WALL , **front: BLUE HALLWAY** , front: CHAIR , front: HATRACK , left: WALL , right: EASEL)

Figure 4.2: Sample instruction with its landmarks plan. Bold components are the true plan.

senting agent actions and outstanding objects appearing in the virtual world. The lexicon is learned by finding co-occurrence of NL words and phrases with specific actions and objects in the simulated virtual world while following the corresponding NL instruction. The learning process is called Graph Intersection Lexicon Learnng (GILL) (Chen & Mooney, 2011; Chen, 2012b) and is similar to other "cross-situational" approaches of learning word meanings (Siskind, 1996; Thompson & Mooney, 2003). From the training data (e_i, c_i), the algorithm first collects all navigation plans c_js representing the entire context, co-occurring with an n-gram w as candidate meanings for w. This initial candidate meaning set is expanded while repeatedly taking intersections between the candidate meanings, where the intersections can be obtained by taking the largest common subgraphs. The resulting candidate set is ranked by the following scoring metric for an n-gram w and an MR graph m:

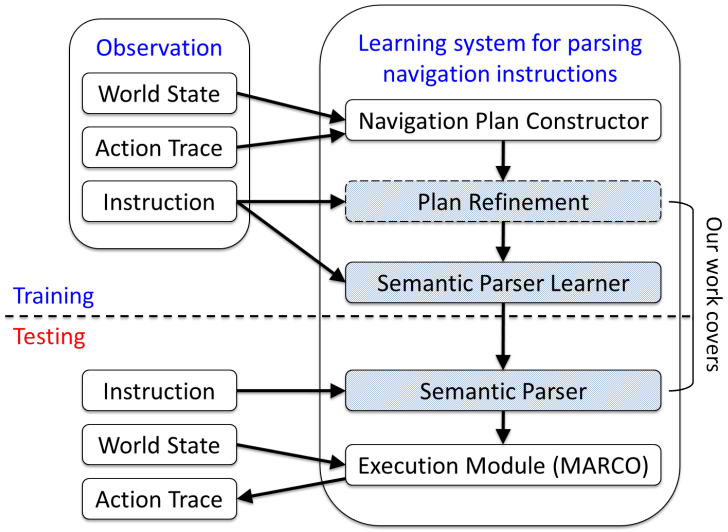


Figure 4.3: An overview of Chen and Mooney's (2011) system. Our approaches replace the roles of the plan refinement component and the semantic parser.

$$Score(w, m) = p(m|w) - p(m|\neg w)$$

which measures how much more likely an MR m appears when w is present compared to when it is not.

After obtaining a lexicon, the *plan refinement* step estimates p_i from context c_i by greedily selecting high-scoring *lexemes* (i.e, lexicon entries of (w_j, m_j)) whose phrases (w_j) cover the instruction e_i and introduce components (m_j) from the landmarks plan c_i. The refined plans are then used to train a semantic parser learner as a supervised training set (e_i, p_i). The trained semantic parser can parse a novel instruction into a formal plan, which is finally executed for end-to-end evaluation. Figure 4.3 illustrates the overall system.

As this figure indicates, our new PCFG induction methods replace the roles of the plan refinement step and the semantic parser in Chen and Mooney's (2011) system. The two systems presented in this chapter are unified systems that simultaneously disambiguate the training data and learn a semantic parser in a single probabilistic framework. We use the landmarks plans and the learned lexicon produced by GILL (Chen & Mooney, 2011) as ambiguous inputs to our system.

4.3 Hierarchy Generation PCFG Approach

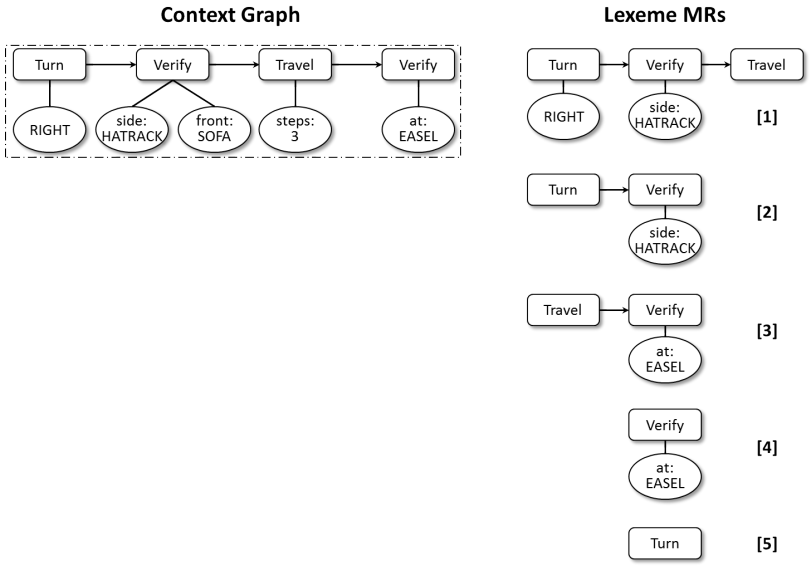
Like Börschinger et al. (2011), our first PCFG approach learns a semantic parser directly from ambiguous supervision: specifically, NL instructions paired with the complete landmarks plans as context in the navigation data. Our method incorporates semantic lexemes obtained from GILL as basic building blocks to find correspondences between NL words and semantic concepts represented by the MRs in the lexemes, instead of building connections for each MR constituent as level, as with Börschinger et al.'s (2011) method. We utilize the hierarchical subgraph relationships between the MRs in the semantic lexicon to produce a smaller, more focused set of PCFG rules. The intuition behind this is analogous to the hierarchical relations between syntactic categories in syntax parsing. In syntax parsing, high level categories, such as S, VP, or NP, refer to bigger concepts that are further divided into smaller concepts, such as V, N, or Det, therefore forming a hierarchical structure. Inspired by this notion, we introduce a directed acyclic graph called the *Lex-*

eme Hierarchy Graph (LHG) which represents the hierarchical relationships between lexemes. Since complex lexeme MRs represent complicated combined semantic concepts and simple MRs represent simple concepts, it is natural to construct a hierarchy between the lexeme MRs. The LHGs for all training examples are used to construct production rules for PCFG, which are then parametrized using EM. Finally, novel sentences are semantically parsed by computing their most-probable parses using the trained PCFG and extracting an MR from the resulting parse tree.

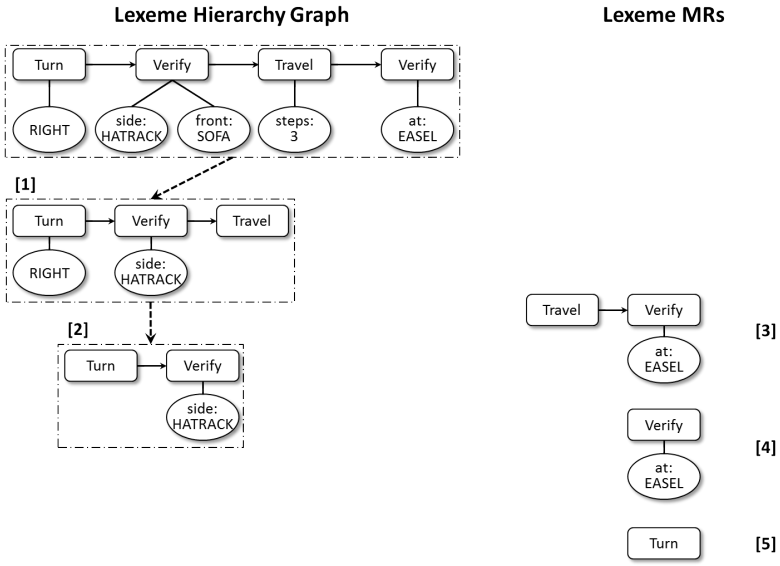
4.3.1 Constructing a Lexeme Hierarchy Graph

An LHG represents the hierarchy of semantic concepts relevant to a particular training instance by encoding the subgraph relations between the MRs of relevant lexemes. Algorithm 1 shows pseudo-code of LHG construction for the training instance (e_i, c_i). First, we obtain all relevant lexemes (w_j^i, m_j^i) in the lexicon L, where the MR m_j^i is a subgraph of the context c_i (denoted as $m_j^i \subset c_i$). These lexemes are sorted in descending order based on their MR sizes (i.e, number of component nodes in m_j^i). Next, lexemes are inserted, in order, into the MR hierarchy graph starting with the root node of the context c_i. The MR of an added child should be a subgraph of the MR of its parent. Figures 4.4 and 4.5 illustrate a sample construction of an LHG.

In this step, a lexeme is only added as a child of current leaf nodes. The lexeme is added multiple times if multiple leaves are supergraphs of this lexeme. Thus, it frequently happens that one lexeme is the subgraph of multiple

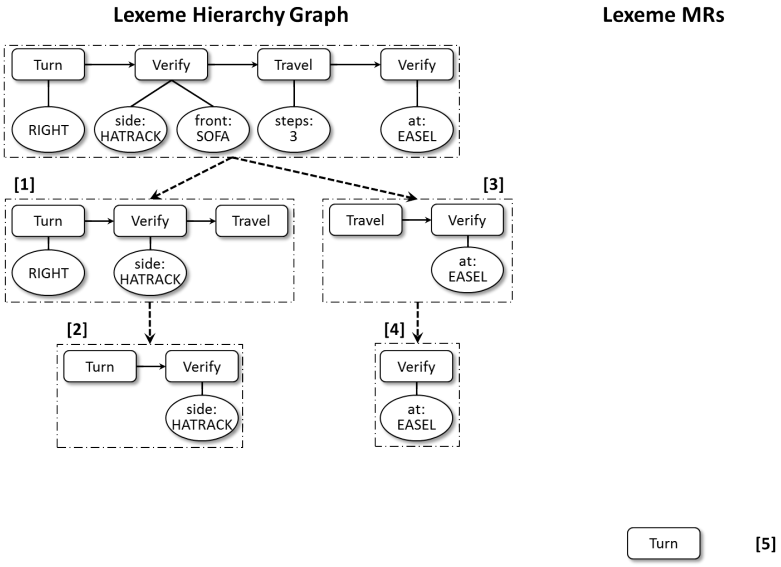


(a) All relevant lexemes are obtained for the training example and ordered by the number of nodes in their MR.

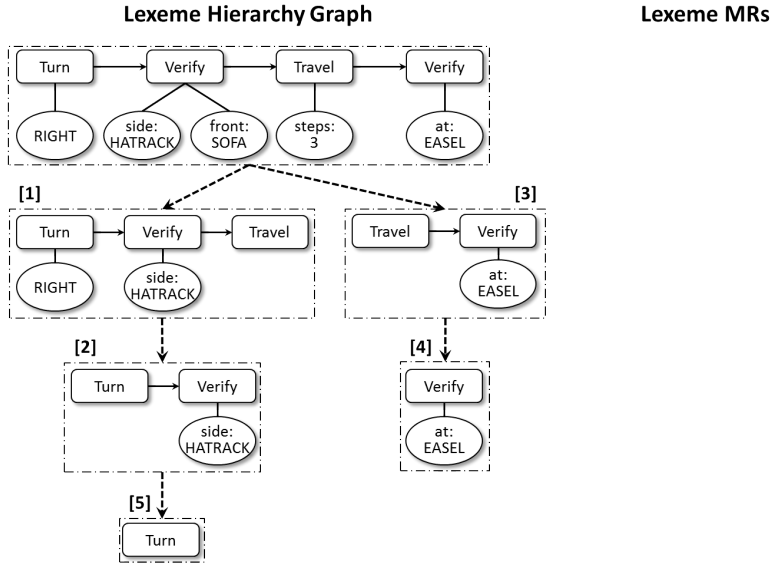


(b) Lexeme MR [1] is added as a child of the top node. MR [2] is a subgraph of [1], so it is added as its child.

Figure 4.4: Sample LHG construction for the context `Turn`(`RIGHT`), `Verify`(`side` : `HATRACK`, `front` : `SOFA`), `Travel`(`steps` : 3), `Verify`(`at` : `EASEL`).



(a) MR [3] is not a subgraph of [1] or [2], so it is added as a child of the root. MR [4] is added under [3].



(b) Finally, MR [5] is recursively filtered down and finds its correct place under [2].

Figure 4.5: Sample LHG construction for the context `Turn(RIGHT)`, `Verify(side : HATRACK, front : SOFA)`, `Travel(steps : 3)`, `Verify(at : EASEL)`, continued from Figure 4.4.

Algorithm 1 LEXEME HIERARCHY GRAPH (LHG)

Input: Training instance (e_i, c_i), Lexicon L
Output: Lexeme hierarchy graph for (e_i, c_i)

Find relevant lexemes $(w_1^i, m_1^i), \ldots, (w_n^i, m_n^i)$ s.t. $m_j^i \subset c_i$
Create a starting node T; $MR(T) \leftarrow c_i$
for all m_j^i in the descending order of size **do**
    Create a node T_j^i; $MR(T_j^i) \leftarrow m_j^i$
    PLACELEXEME(T_j^i,T)
end for

procedure PLACELEXEME(T',T)
    for all children T_j of T **do**
        if $MR(T') \subset MR(T_j)$ **then**
            PLACELEXEME(T',T_j)
        end if
    end for
    if T' was not placed under any child T_j **then**
        Add T' as child of T
    end if
end procedure

other mutually disjoint lexemes. Since the subsequent processes would only duplicate a lower level hierarchy over multiple times, the LHG is maintained as a *directed acyclic graph* (DAG) instead of a tree so that one particular lexeme only appears once per the LHG of the training example. The steps are repeated until all lexemes are added.

The initial LHG may contain nodes with too many children, which may result in too many PCFG rules, because we add a PCFG production rule for every possible k-permutation of the children of each node (see Section 4.3.2). Thus, we introduce *pseudo-lexeme* nodes to reduce the branching factor by

Algorithm 2 Adding Pseudo Lexemes to LHG

Input: LHG with root T
Output: LHG with pseudo lexemes added
procedure ReconstructLHG(T)
  repeat
    $((T_i, T_j), m') \leftarrow$ MostSimilarPair(T)
    Add child T' of T; $MR(T') \leftarrow m'$
    Move T_i and T_j to be children of T'
  until There are no more pairs to combine
  for all non-leaf children T_k of T **do**
    ReconstructLHG(T_k)
  end for
end procedure

procedure MostSimilarPair(T)
  for all pairs (T_i, T_j) of children of T **do**
    $m' \leftarrow$ smallest graph s.t. $MR(T_i) \subset m'$,
      $MR(T_j) \subset m'$, $m' \subset MR(T)$
    $score \leftarrow Sim(MR(T_i), MR(T_j), m')$
    if $maxScore < score$ **then**
      $maxPair \leftarrow (T_i, T_j)$
      $maxScore \leftarrow score$
    end if
  end for
  return $(maxPair, m')$
end procedure

repeatedly combining the two most similar children of each node. Pseudocode for this procedure is shown in Algorithm 2. The MR for a pseudo-lexeme is the minimal graph, m', which is a supergraph of both of the lexeme MRs that it combines. The pair of most similar children, (m_i, m_j), is calculated by the ratio of how many nodes in m_i and m_j overlap with m' and is described as follows:

$$Sim(m_i, m_j, m') = \frac{|m_i| + |m_j|}{2\,|m'|}$$

where $|m|$ is the number of nodes in the MR m. Adding pseudo-lexemes has another advantage: they can be intuitively thought of as higher-level semantic concepts composed of two or more concepts. Moreover, the pseudo-lexemes will likely occur in other training examples as well, allowing for more flexible interpretations. For example, let us consider the rule $A \Rightarrow BCD$ from an LHG, and we introduce pseudo-lexeme E, so that we build two rules, $A \Rightarrow BE$ and $E \Rightarrow CD$. It is likely that E occurs in another rule in other training examples, such as $E \Rightarrow FG$. Then, we can increase the model's expressiveness by having rules such as $A \Rightarrow^* BFG$, providing more flexibility when parsing a novel NL sentence.

4.3.2 Composing PCFG Rules

The next step is to compose PCFG rules from the LHGs. The process is summarized in Figure 4.6. We basically follow the scheme of Börschinger et al. (2011), but instead of generating NL words from each atomic MR, words are generated from each lexeme MR, and smaller lexeme MRs are generated from

more complex ones as given by the LHGs. A nonterminal S_m is generated for the MR, m, of each LHG node. Then, for every LHG node, T, with MR, m, we add rules of the form $S_m \rightarrow S_{m_i}...S_{m_j}$, where the RHS is some k-permutation of the nonterminals for the MRs of the children of node T. Although Börschinger et al. made sure every MR constituent generates at least one NL word, we must generate every possible ordered subset of the children nonterminals, because we do not know which subgraph of the whole context c_i is responsible for generating the NL words in the sentence. In summary, complex semantic concepts are described as an ordered hierarchy tree of smaller concepts that are eventually described by NL phrases.

The rest of the process more closely follows Börschinger et al.'s (2011) approach. Every lexeme MR, m, [1] generates a rule $S_m \rightarrow Phrase_m$, and every $Phrase_m$ generates a sequence of NL words, including one or more "content words" ($Word_m$) for expressing m and zero or more "extraneous" words ($Word_\emptyset$). While Börschinger et al. let $Word_m$ generate any NL words in the vocabulary weighted by EM, we restrict each $Word_m$ to produce only the NL phrases or words associated with m in the lexicon. This helps reduce the PCFG to a tractable size and also decreases unnecessary ambiguity caused by the possible connections between lexemes and all words in the vocabulary. $Word_\emptyset$ has rules for every word, including unknown ones, and thus is responsible for generating uncovered words.

[1] Pseudo-lexemes only generate words by generating child lexemes.

$$Root \rightarrow S_c, \qquad \forall c \in contexts$$

$$\forall non\text{-}leaf\ node\ and\ its\ MR\ m$$
$$S_m \rightarrow \{S_{m_1}, \ldots, S_{m_n}\},$$
$$where\ m_1, \ldots, m_n\text{: children lexeme MR of } m,$$
$$\{\cdot\}\text{: all } k\text{-permutations for } k = 1, \ldots, n$$

$$\forall lexeme\ MR\ m$$
$$S_m \rightarrow Phrase_m$$

$Phrase_m \rightarrow Word_m$	$PhX_m \rightarrow PhX_m\ Word_m$
$Phrase_m \rightarrow PhX_m\ Word_m$	$PhX_m \rightarrow PhX_m\ Word_\emptyset$
$Phrase_m \rightarrow Ph_m\ Word_\emptyset$	$Ph_m \rightarrow PhX_m\ Word_m$
$PhX_m \rightarrow Word_m$	$Ph_m \rightarrow Ph_m\ Word_\emptyset$
$PhX_m \rightarrow Word_\emptyset$	$Ph_m \rightarrow Word_m$

$$Word_m \rightarrow s, \qquad \forall s\ \text{s.t.}\ (s, m) \in lexicon\ L$$
$$Word_m \rightarrow w, \qquad \forall word\ w \in s\ \text{s.t.}\ (s, m) \in lexicon\ L$$

$$Word_\emptyset \rightarrow w, \qquad \forall word\ w \in NLs$$

Figure 4.6: Summary of the rule generation process for the Hierarchy Generation PCFG approach based on LHGs. *NLs* refer to the set of NL words in the corpus. Lexeme MR rules follow the schemata of Börschinger et al. (2011), and allow every lexeme MR to generate at least one NL word through a unigram Markov process. Note that pseudo-lexeme nodes do not produce NL words.

4.3.3 Parsing Novel NL Sentences

To learn the parameters of the resulting PCFG, we use the Inside-Outside algorithm.[2] Next, we use the standard probabilistic CKY algorithm to produce the most probable parses for novel NL sentences (Jurafsky & Martin, 2000). A simplified version of a sample parse tree from the Hierarchy

[2]We used the implementation available at `http://web.science.mq.edu.au/~mjohnson/Software.htm`, which was also used by Börschinger et al. (2011).

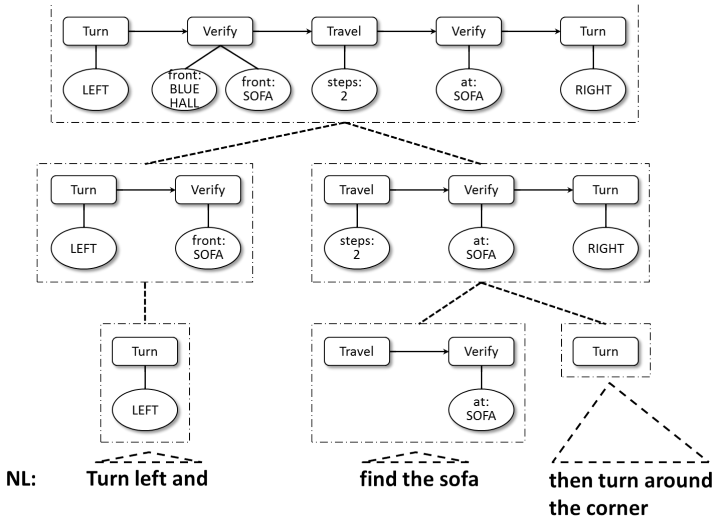


Figure 4.7: Simplified parse for the sentence *"Turn left and find the sofa then turn around the corner"* for the Hierarchy Generation Model. Nonterminals show the MR graph, where additional nonterminals for generating NL words are omitted.

Generation PCFG Model is shown in Figure 4.7.

Börschinger et al. (2011) simply read the MR, m, for a sentence off the top S_m nonterminal of the most probable parse tree. Therefore, their model is able to produce only the MRs seen during training. In contrast, our method produces the output MR parse by composing the appropriate subset of lexeme MRs that are actually responsible for generating NL words. Thus, our system is able to produce novel MRs as long as they are some subgraphs of the complete context (c_i) that appeared in the training data.

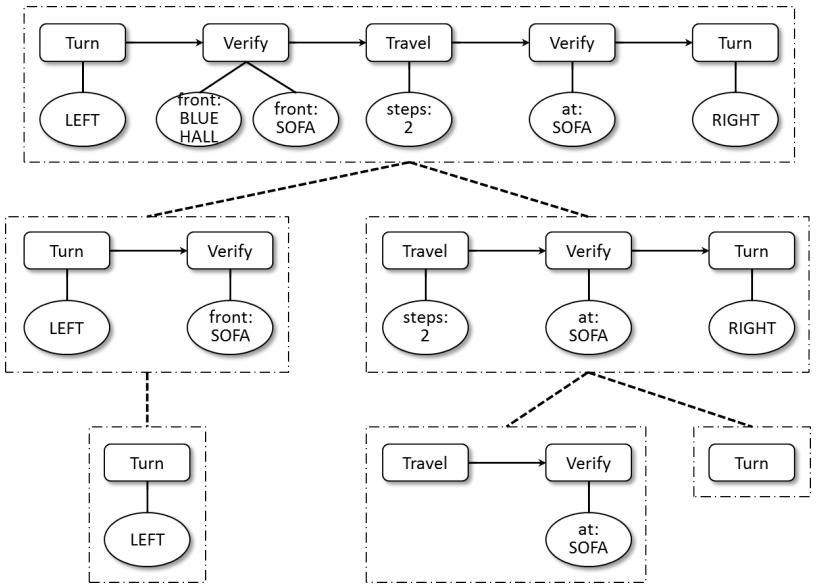
First, the parse tree is pruned to remove all the subtrees with the root of $Phrase_x$, producing the tree with only S_m nodes. The pruned subtrees are only concerned about generating NL words, so we can figure out which lexeme

Algorithm 3 CONSTRUCT PARSED MR RESULT

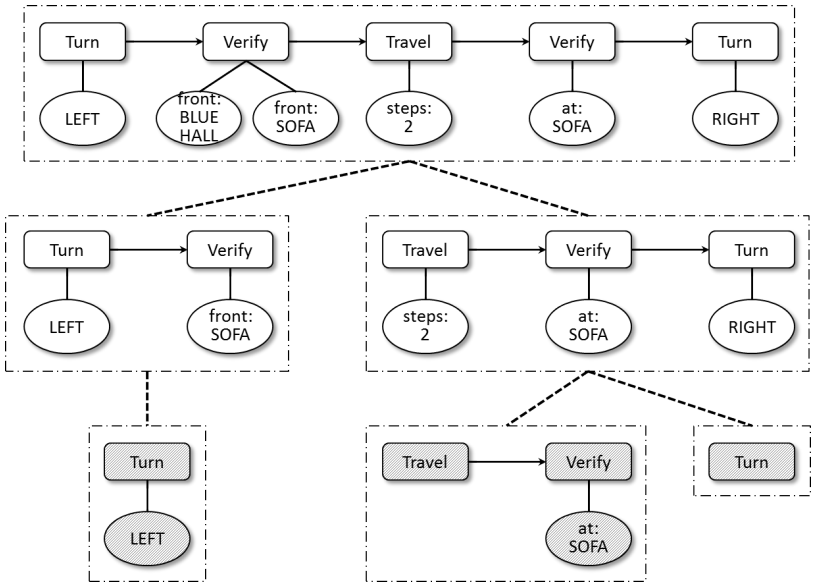
Input: Parse tree T for input NL, e, with all $Phrase_x$ subtrees removed.
Output: Semantic parse MR, m, for e
procedure OBTAINPARSEDOUTPUT(T)
  if T is a leaf **then**
    return $MR(T)$ with all its nodes marked
  end if
  for all children T_i of T **do**
    $m_i \leftarrow$ OBTAINPARSEDOUTPUT(T_i)
    Mark the nodes in $MR(T)$ corresponding
      to the marked nodes in m_i
  end for
  if T is not the root **then**
    return $MR(T)$
  end if
  return $MR(T)$ with unmarked nodes removed
end procedure

MRs are involved in generating the target NL sentence. The leaves S_m in the pruned tree show lexeme MRs m that are responsible for generating the NL sentence. These lexeme MR components are combined so that they conform to the parse tree structure to produce the final MR parse.

Algorithm 3 shows the pseudo-code for producing the MR parse from the pruned parse tree. Figures 4.8 and 4.9 are a sample trace. The algorithm recursively traverses the parse tree. When a leaf-node is reached, it marks all of the nodes in its MR. After traversing all of its children, a node in the MR for the current parse-tree node is marked if, and only if, its corresponding node in any of the children's MRs was marked. Removing all of the unmarked nodes from the root MR results in the final MR we want.

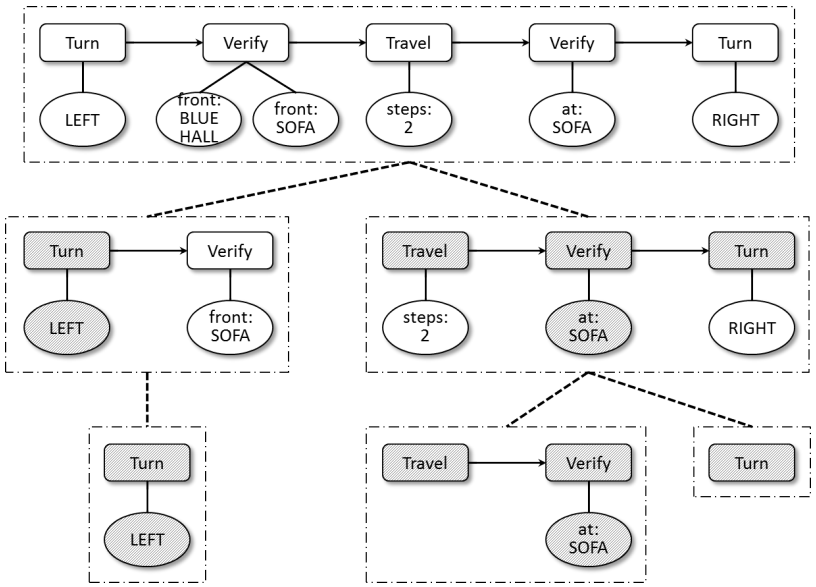


(a) Pruned parse tree showing only MRs for S_m nodes.

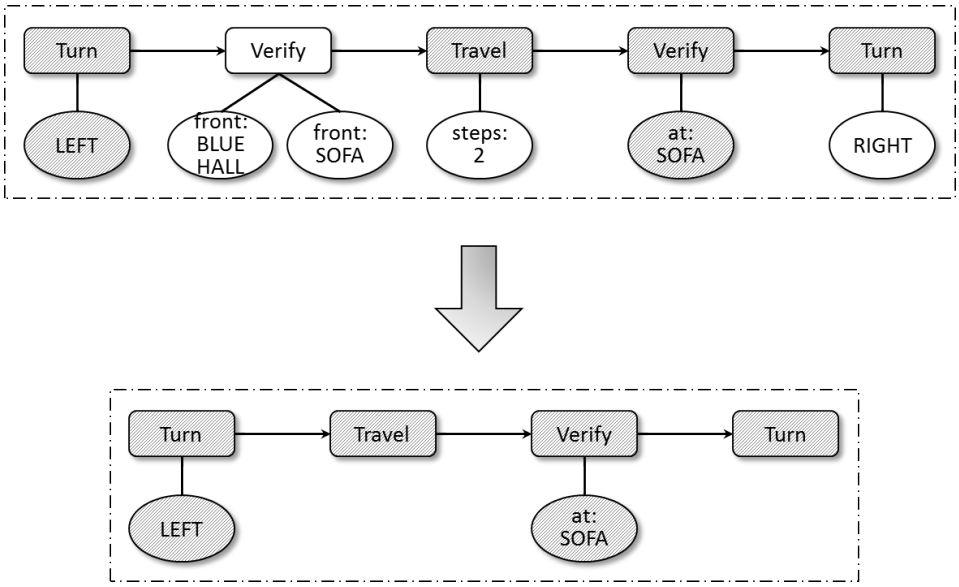


(b) Leaf nodes have all their elements marked.

Figure 4.8: Sample construction of a derived MR output from a pruned parse tree for the Hierarchy Generation PCFG approach.



(a) Upper level nodes are marked according to leaf-node markings.



(b) Removing all unmarked elements for the root node leads to the final MR output.

Figure 4.9: Sample construction of a derived MR output from a pruned parse tree for the Hierarchy Generation PCFG approach, continued from Figure 4.8.

4.4 Unigram Generation PCFG Approach

The key idea of the Hierarchy Generation PCFG approach described in section 4.3 is to encode in the model how complex semantic concepts generate smaller ones in order, which finally make correspondences with NL groundings. Even though the process follows a natural intuition, this PCFG approach could suffer from an exploding number of grammar rules if we do not introduce pseudo-lexemes from initial LHG structures before creating final PCFG rules. This is mainly because we have to consider all the permutations of how semantic lexemes are generated in order to know which lexeme MR is connected to which NL substrings. On the other hand, even though introducing pseudo-lexemes is inevitable in order to maintain the feasibility of the Hierarchy Generation model, pseudo-lexemes also increase the total number of nonterminals in the resulting PCFG, which results in more latent variables to consider during the EM training.

In this section, we introduce a simpler approach that does not rely on a pre-computed LHG for each example when generating a final PCFG rule set. This new PCFG approach uses a unigram Markov process to generate related semantic lexemes one by one, starting from nonterminals representing the context MR, each of which further generates corresponding NL words. This approach has an advantage over the previous LHG-based hierarchical approach, because it does not require that extra PCFG rules be introduced for permutations of nonterminals, and it produces a simpler PCFG rule set overall. Experimental results show that the new, simpler approach performs

better than the more complex hierarchical approach, and it also runs faster during training due to the smaller number of generated PCFG rules.

4.4.1 Composing PCFG Rules

The first step in composing PCFG rules is to learn a semantic lexicon from the training data, which follows the same scheme as in the previous PCFG approach (Section 4.3). Then, for each training example, we list every semantic lexeme relevant to this example. Formally, for a training example pair of an NL instruction and a context MR (landmarks plan) (e_i, c_i), we obtain all the semantic lexemes (w_j^i, m_j^i) from the lexicon L such that the MR m_j^i is a subgraph of the context c_i and w_j^i appears in the NL sentence e_i.

Then, without calculating inter-lexeme subgraph relationships, we can compose PCFG rules in a straightforward manner. The rule generation process is summarized in Figure 4.10. The basics of the scheme are the same as before. Each semantic lexeme is responsible for generating related NL words appearing in the learned lexicon L, following the same unigram Markov generation of NL words as in the Hierarchy Generation PCFG approach. The only difference is that, instead of hierarchical lexeme generation, each relevant lexeme is generated one by one from the context MR, simulating a unigram Markov process.

This rule generation process does not have to consider the permutation orders of lexemes explicitly. The means of generating lexemes from the context MR is flat; thus, the unigram selection of lexemes in the second and third lines

$Root \rightarrow S_c, \qquad \forall c \in contexts$

$\forall lexeme\ MR\ m$
$S_c \rightarrow L_m\ S_c$
$S_c \rightarrow L_m$

$L_m \rightarrow Phrase_m$
$Phrase_m \rightarrow Word_m$ $\qquad$ $PhX_m \rightarrow PhX_m\ Word_m$
$Phrase_m \rightarrow PhX_m\ Word_m$ $\qquad$ $PhX_m \rightarrow PhX_m\ Word_\emptyset$
$Phrase_m \rightarrow Ph_m\ Word_\emptyset$ $\qquad$ $Ph_m \rightarrow PhX_m\ Word_m$
$PhX_m \rightarrow Word_m$ $\qquad$ $Ph_m \rightarrow Ph_m\ Word_\emptyset$
$PhX_m \rightarrow Word_\emptyset$ $\qquad$ $Ph_m \rightarrow Word_m$
$Word_m \rightarrow s, \qquad \forall s\ \text{s.t.}\ (s, m) \in lexicon\ L$
$Word_m \rightarrow w, \qquad \forall word\ w \in s\ \text{s.t.}\ (s, m) \in lexicon\ L$

$Word_\emptyset \rightarrow w, \qquad \forall word\ w \in NLs$

Figure 4.10: Summary of the rule generation process of the Unigram Generation PCFG approach. *NLs* refer to the set of NL words in the corpus. Lexeme MR rules are just the same as the Hierarchy Generation PCFG approach (Section 4.3). Every lexeme MR should generate at least one relevant NL word through a unigram Markov process. The second and third lines cover the unigram Markov process of generating each relevant lexeme MR from the context MR.

of Figure 4.10 already considers all possible permutations.

4.4.2 Parsing Novel NL Sentences

When parsing a new NL sentence in the simplified model, we follow an approach similar to that of the Hierarchy Generation approach using LHGs, but in a simpler way. Again, we use the Inside-Outside algorithm first, to get weight parameters for the resulting PCFG rule set, and then the prob-

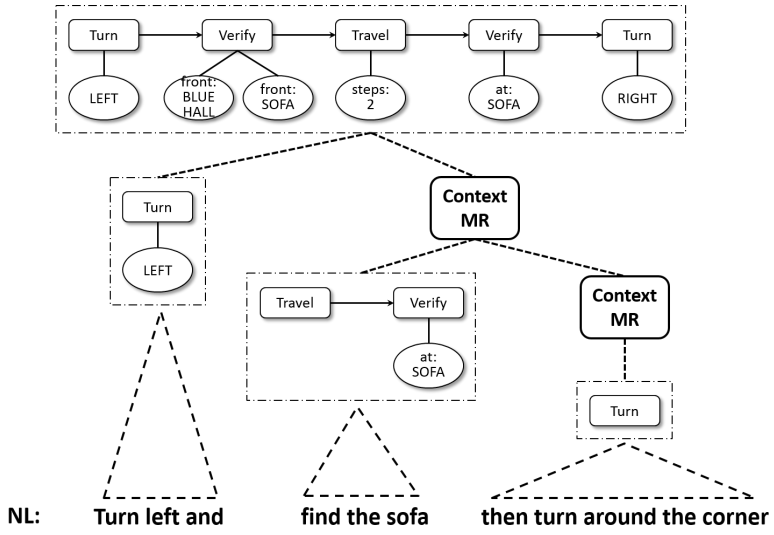


Figure 4.11: Simplified parse for the sentence *"Turn left and find the sofa then turn around the corner"* for the Unigram Generation Model. Nonterminals show the MR graph, where additional nonterminals for generating NL words are omitted. The node "Context MR" refers to the same nonterminal of the root node that represents the context MR.

abilistic CKY algorithm predicts the most probable parse tree for test NL sentences. Whereas the previous approach generates parse trees with hierarchical structures containing relevant lexemes for a given NL sentence, this approach produces a flat lexeme structure in its resulting parse trees. A simplified version of a sample parse tree for the Unigram Generation PCFG model appears in Figure 4.11. Thus, parsing and obtaining a properly derived MR for a test sentence becomes simple. All we need to do is find internal L_m nonterminal nodes appearing in the parse trees. A simple tree traversal of the parse tree to find L_m nodes will suffice. All the subtree structures below the nonterminal $Phrase_m$ can be discarded because they are, again, only used for

NL generation and are not relevant to determining the relevant lexemes used. Once we know relevant lexeme nodes of L_ms and context MR S_c appearing in the root of the parse tree, then we can compose the desired final MR by marking context MR c of corresponding nodes appearing in each lexeme MR m.

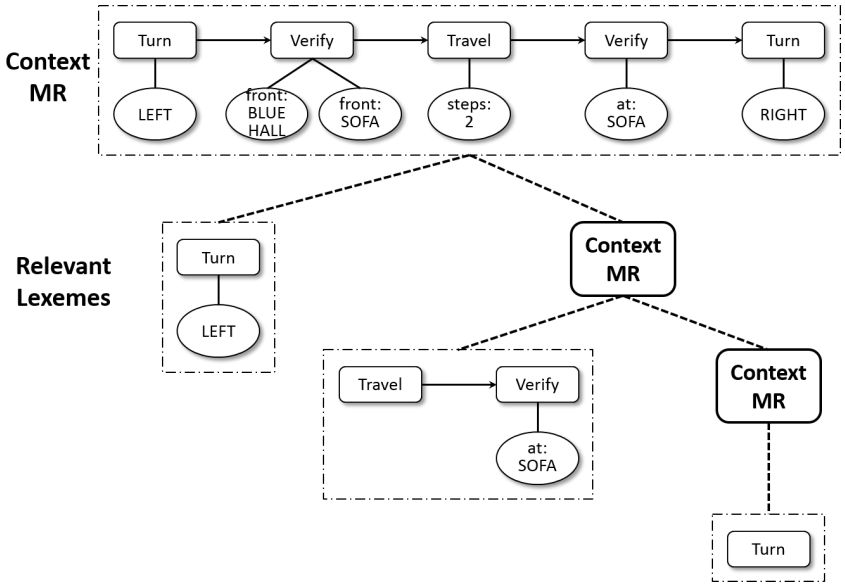
Algorithm 4 shows the pseudo-code for this process. Figures 4.12 and 4.13 are a sample trace. The algorithm simply goes over all the relevant lexemes and marks the corresponding nodes appearing in the context MR. Finally, removing the unmarked nodes from the root context MR results in the final derived MR.

4.5 Experimental Evaluation

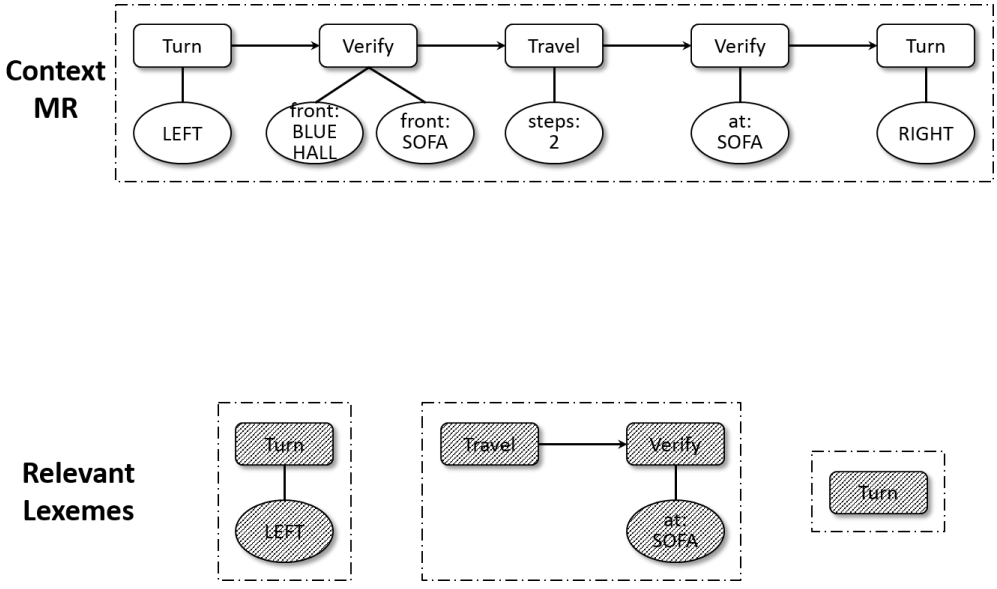
4.5.1 Data

To evaluate our methods, we used the English instructions and follower data originally collected by MacMahon et al. (2006).[3] These data contain 706 route instructions for three virtual worlds called Grid, L, and Jelly. The instructions were produced by six instructors for 126 unique starting and ending location pairs in the three worlds. Each navigation instruction comes with 1 to 15 human followers' traces with an average of 10.4 actions per instruction, and the followers are separate from the instruction annotators. Each instruction consists of an average of 5.0 sentences, each containing an average of 7.8

[3]Data and relevant code are available at `http://www.cs.utexas.edu/users/ml/clamp/navigation/`

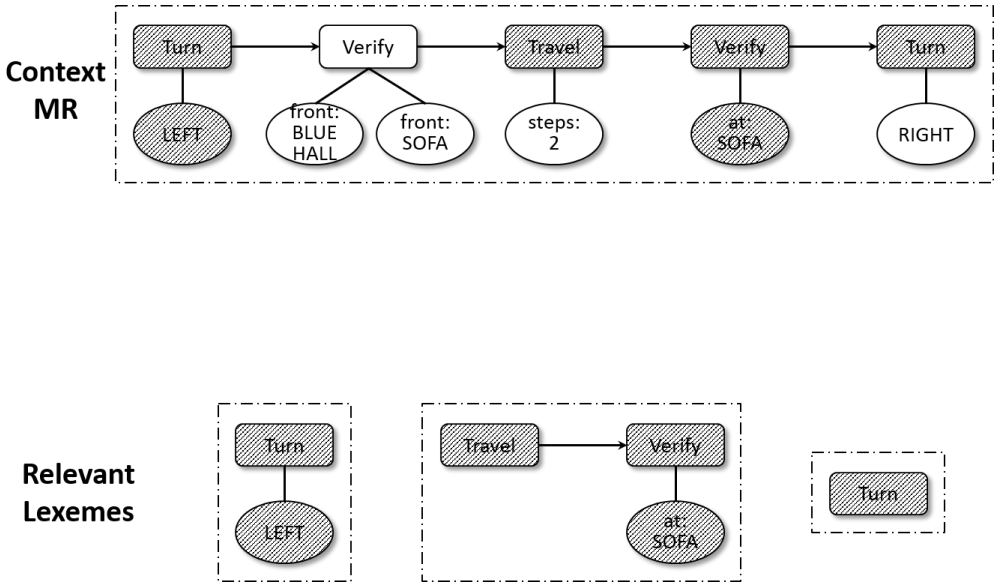


(a) Pruned parse tree showing only MRs for S_m and L_m nodes, referring to context MR and relevant lexeme MRs, respectively.

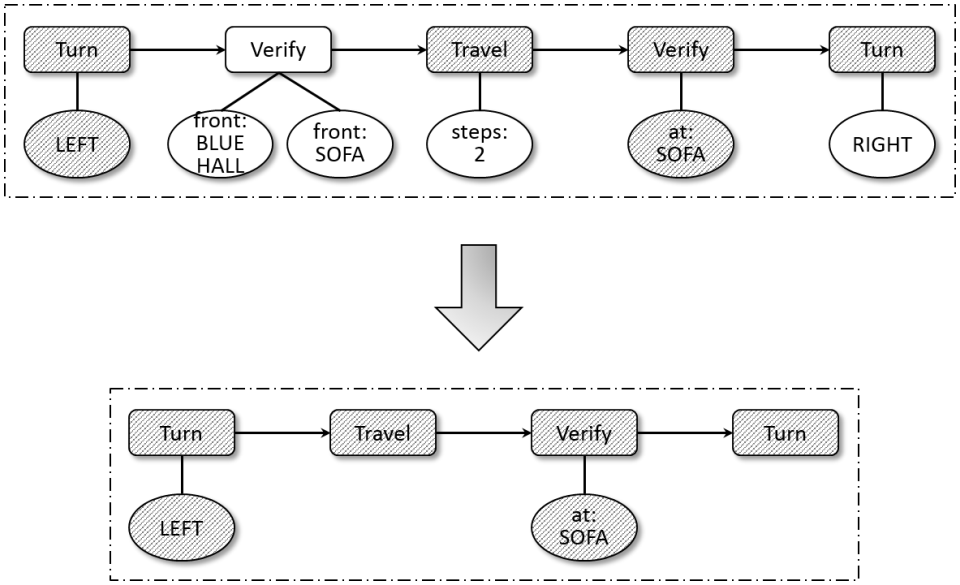


(b) First, relevant lexeme MRs are extracted from the parse tree.

Figure 4.12: Sample construction of a derived MR output from a pruned parse tree for the Unigram Generation PCFG approach.



(a) Context MR is marked according to the corresponding nodes in lexeme MRs.



(b) Removing all unmarked elements for the root node of the context MR leads to the final derived MR.

Figure 4.13: Sample construction of a derived MR output from a pruned parse tree for the Unigram Generation PCFG approach, continued from Figure 4.12.

Algorithm 4 CONSTRUCT PARSED MR RESULT

Input: Parse tree T for input NL, e, with all $Phrase_x$ subtrees removed.
Output: Semantic parse MR, m, for e
procedure OBTAINPARSEDOUTPUT(T)
  $Lexemes \leftarrow$ OBTAINRELEVANTLEXEMES(T)
  for all Lexeme m_i of $Lexemes$ **do**
    Mark the nodes in $MR(T)$ corresponding
      to the nodes in m_i
  end for
  return $MR(T)$ with unmarked nodes removed
end procedure
procedure OBTAINRELEVANTLEXEMES(T)
  if T is a leaf **then**
    return a singleton set containing $MR(T)$
  end if
  $Result \leftarrow$ a singleton set containing $MR($RIGHTCHILD(T))
  if RIGHTCHILD(T) exists **then**
    $T_R \leftarrow$ RIGHTCHILD(T)
    Add all elements of OBTAINRELEVANTLEXEMES(T_R) to $Result$
  end if
  return $Result$
end procedure

words.

In addition, Chen and Mooney (2011) constructed an additional single-sentence corpus by matching each sentence with the majority of human followers' actions in order to ease the training process. This single-sentence version is used for training our models, but both versions of the data are used for testing. There are manually annotated "gold standard" plans only for evaluation purposes. Detailed statistics are shown in Table 4.1.

In addition to the English data, Chen (2012a) introduced a Chinese

	Original (Paragraph)	Single-sentence
# instructions	706	3236
Vocabulary size	660	629
Avg. # sentences	5.0 (2.8)	1.0 (0)
Avg. # words	37.6 (21.1)	7.8 (5.1)
Avg. # actions	10.4 (5.7)	2.1 (2.4)

Table 4.1: Statistics about the navigation corpus originally collected by MacMahon et al. (2006) and the single-sentence processed version by Chen and Mooney (2011). Average values are shown, as well as standard deviations, in parentheses.

translation version of the corpus. As in previous studies on the navigation corpus (Chen & Mooney, 2011; Chen, 2012a), our system is language-independent and capable of learning to interpret any language semantics within the same framework without modification. These additional Chinese data were annotated by a single native Mandarin Chinese speaker and are translations of each sentence of the English navigation instructions.

One major issue with Chinese data is that unlike English or Korean, Chinese does not have space word boundaries. In order to handle this, Chen (2012a) presented two versions of the Chinese data. The first version treats each Chinese character as a word by putting a space between each character. It seems an unreasonable assumption to make, but whereas English has only 26 characters in total and each character does not usually form a meaningful semantic unit, the number of Chinese characters is much larger and each Chinese character constitutes a single semantic unit, because Chinese characters are ideograms. Another version is constructed by using an existing tool for

		Original (Paragraph)	Single-sentence
# instructions		706	3236
Segmented	Vocabulary size	661	508
	Avg. # words	31.6 (18.1)	6.9 (4.9)
Character	Vocabulary size	448	328
	Avg. # words	48.9 (28.3)	10.6 (7.3)

Table 4.2: Word statistics about the Chinese translation version of the navigation corpus (Chen, 2012a). Both word-segmented ("Segmented") and character-segmented ("Character") versions are presented.

segmenting Chinese characters into words. Chen (2012a) provided this version of the corpus processed by the Stanford Chinese Word Segmenter (Chang, Galley, & Manning, 2008). Word statistics of both versions of Chinese data are presented in Table 4.2.

4.5.2 Methodology and Results

For evaluation, we followed the same methodology as Chen and Mooney (2011), performing "leave one environment out" cross-validation (i.e, training on two environments and testing on the third). We present direct comparisons with the best reported results of Chen and Mooney (2011) and Chen (2012b).[4] Regarding the semantic lexicon, all our proposed methods use the Graph Intersection Lexicon Learnng (GILL) introduced by Chen and Mooney (2011) and later named by Chen (2012b). Some of the best cited results for Chen (2012b) used a new lexicon learning algorithm called Subgraph Generation

[4]The experimental results of Chen (2012b) subsume those of Chen (2012a)

Methods	Precision	Recall	F1
Hierarchy Generation PCFG model	87.58	65.41	74.81
Unigram Generation PCFG model	86.10	∗68.79	∗**76.44**
Chen and Mooney (2011)	∗90.16	55.41	68.59
Chen (2012b)	88.36	57.03	69.31
Chen (2012b) with additional data	88.11	56.57	68.90

Table 4.3: Test accuracy for semantic parsing on English data; '∗' denotes statistically significant difference compared to the second best results ($p < .05$).

Online Lexicon Learning (SGOLL). We performed a Wilcoxon signed-rank test for statistical significance, and '∗' denotes significant differences ($p < .05$) compared to the second best results in the tables.

4.5.2.1 Semantic Parsing Results

Semantic parsing evaluates how accurately the model learns to map novel NL sentences in the test environment into correct MRs. Partial semantic-parsing accuracy (Chen & Mooney, 2011) assigns partial credit if two MRs have the same predicate, and additional credit for each matching argument. Precision (accuracy of the system output against the gold standard), recall (the gold standard against the system output), and F1 (harmonic mean between precision and recall) are evaluated, and every metric considers partial credit for approximately correct MRs.

Table 4.3 shows a direct comparison of our proposed two PCFG models described in Section 4.3 and Section 4.4. We also make comparisons with the best results presented in the following cited papers: the refined landmarks plan

for Chen and Mooney (2011) and the expanded CFG with SGOLL lexicon learning for Chen (2012b) (see Section 4.7). We also cite the results with additional training data obtained from the Amazon Mechanical Turk from Chen (2012b), which are the best obtained results in this paper, even though it uses external training data.

The experimental results clearly demonstrate that our two PCFG methods are better than Chen and Mooney's (2011), by a large margin. The Hierarchy Generation approach performs better by 6 points in F1 and the simplified version performs 8 points better. Furthermore, our two methods are better than Chen's (2012b), even compared to the results with additional training data. Our two PCFG-based approaches with semantic lexicon are able to probabilistically disambiguate the training data as well as simultaneously learn a statistical semantic parser within a single framework.

This results in better overall performance compared to previous studies, since they lose possibly useful information due to separate stages of the system, particularly during the refinement stage. In addition, their refinement process is limited to incorporating only the high-scoring lexemes. By contrast, our approaches probabilistically consider relatively low score but useful lexemes in the generative process, and, therefore, has more flexibility in the final MR interpretation. This is reflected in the increase of recall for our approaches since our methods have a wider coverage of lexemes during training.

One additional point to note is that the Unigram Generation PCFG approach performs better than the Hierarchy Generation PCFG approach.

Data	Methods	Precision	Recall	F1
Segmented	Hierarchy PCFG model	80.56	71.14	75.53
	Unigram PCFG model	79.45	∗73.66	∗**76.41**
	Chen (2012b)	∗88.87	58.76	70.74
Character	Hierarchy PCFG model	79.77	67.38	73.05
	Unigram PCFG model	79.73	∗75.52	∗**77.55**
	Chen (2012b)	∗92.48	56.47	70.01

Table 4.4: Test accuracy for semantic parsing for Chinese Mandarin data. "Segmented" refers to the word-segmented version of the Chinese corpus by Stanford Chinese Word Segmenter, and "Character" refers to the character-segmented version; '∗' denotes statistically significant difference compared to the second best results ($p < .05$).

Notably, the recall is boosted up by 10%, which results in better F1, even though the precision is low. Compared to the Hierarchy Generation approach, the Unigram Generation version tends to catch more MR elements. The hierarchical structure of lexemes in the Hierarchy Generation approach and the additional pseudo-lexemes tend to select fewer lexemes to connect with the NL sentence. On the other hand, the flat structure of lexemes of the Unigram Generation approach chooses more lexemes for NL groundings.

Table 4.4 shows the semantic parsing results for the Chinese data. In the experiments, we compare our two PCFG approaches with the best results for the previous approach by Chen (2012b). For the "Segmented" version of the corpus, we cite the results with SGOLL lexicon learning, and for the "Character" version, we cite the results with GILL lexicon learning. The resulting trends are similar to those from the English data. Overall, our two PCFG approaches perform much better in F1, which is mainly due to the much

Method	Single-sentence	Paragraph
Hierarchy Generation PCFG model	57.22%	20.17%
Unigram Generation PCFG model	∗**67.14%**	∗**28.12%**
Chen and Mooney (2011)	54.40%	16.18%
Chen (2012b)	57.28%	19.18%
Chen (2012b) with additional data	57.62%	20.64%

Table 4.5: Successful plan execution rates using the MARCO execution module on English test data; '∗' denotes statistically significant difference compared to the second best results ($p < .05$).

higher value of recall. In addition, the results of the Chinese data support our contention that the Unigram Generation PCFG approach performs better than the Hierarchy Generation approach. The flat structure of semantic lexemes generated by the Unigram PCFG approach helps the model capture more correct elements in its parse trees.

4.5.2.2 Navigation Plan Execution Results

The next evaluation is to test the end-to-end execution of the parsed navigation plans for test instructions in novel environments to determine whether they reach the exact desired destinations in the environment. Table 4.5 shows the successful end-to-end navigation task completion rates for both single-sentences and complete paragraph instructions for the English corpus. Following Chen and Mooney (2011), the execution is performed by the MARCO system (MacMahon et al., 2006), with the parsed navigation plans output from our model. For the single-sentence corpus, we also considered whether the virtual agent is facing the correct direction compared to the gold-standard.

Data	Method	Single-sentence	Paragraph
Segmented	Hierarchy PCFG model	61.03%	19.08%
	Unigram PCFG model	∗**63.40%**	∗**23.12%**
	Chen (2012b)	58.70%	20.13%
Character	Hierarchy PCFG model	55.61%	12.74%
	Unigram PCFG model	∗**62.85%**	∗**23.33%**
	Chen (2012b)	57.27%	16.73%

Table 4.6: Successful plan execution rates using the MARCO execution module on the Mandarin Chinese test data. "Segmented" refers to the word-segmented version of the Chinese corpus by Stanford Chinese Word Segmenter, and "Character" refers to the character-segmented version; '∗' denotes statistically significant difference compared to the second best results ($p < .05$).

Our two PCFG approaches outperform the best results of Chen and Mooney (2011) and Chen (2012b), since more accurate semantic parsing produces more successful plans. In particular, the Unigram Generation PCFG approach performs better than the results of the Hierarchy Generation approach by a large margin. This is mainly due to the accuracy difference in semantic parsing. Most notably, higher recall enables the simplified approach to catch many more correct MR elements that are essential for execution. Our two approaches show comparable or even better results than those of Chen (2012b) with additional training data.

Plan execution results for the Chinese data are shown in Table 4.6. The results are similar to those of the English corpus. Our proposed PCFG approaches generally perform better than Chen's (2012b) approach, except that the Hierarchy Generation PCFG approach with character-wise segmented data performs worse. We conjecture that the high complexity of the Hierarchy Gen-

Data	Hierarchy PCFG		Unigram PCFG	
	|Grammar|	Time (hrs)	|Grammar|	Time (hrs)
English	20451	17.26	16357	8.78
Chinese (Segmented)	21636	15.99	15459	8.05
Chinese (Character)	19792	18.64	13514	12.58

Table 4.7: Comparison of training time in seconds between our two PCFG approaches along with the average numbers of productions in the PCFG.

eration approach on the character-segmented corpus makes the model overfit to the training data. This is primarily caused by the large number of the produced PCFG rules due to considering all the permutations and the longer NL sentences by the character-level segmentation.

4.5.2.3 Training Time Comparison

We compared the average training time of our two PCFG approaches in Table 4.7. The Unigram Generation approach produces fewer PCFG rules, which is expected due to the simpler PCFG generation steps. As a result, the training time for estimating the most probable weight parameters using the Inside-Outside algorithm is considerably less.

4.6 Discussion

Our two PCFG approaches are novel compared to that of Börschinger et al. (2011) in the following ways:

- The basic building blocks for associating NL and MR are semantic lexemes instead of atomic MR constituents. This prevents the number of

produced PCFG rules from exploding, which happens easily in Börschinger et al.'s (2011) approach for even a moderately complex MR language. Considering there are a large number of examples in the navigation data that have 20 or more atomic actions in their context MRs per NL sentence, direct application of Börschinger et al. (2011) would cause 20! ($> 10^{18}$) PCFG production rules to deal with. As mentioned earlier, intuitively the lexemes are analogous to syntactic categories in syntax parsing, in that complex lexeme MRs represent complicated semantic concepts, whereas higher-level syntactic categories such as S, VP, or NP represent complex syntactic structure.

- Our approach has the ability to produce a previously unseen MR, whereas Börschinger et al.'s (2011) approach can generate only a parsed MR and only if it is included in the PCFG rules constructed from the training data. Even though our MR parse is restricted to a subset of the training contexts c_is, our model allows for an exponentially large number of combinations.

In addition, our approaches also cover wider selections of MR outputs than the approaches of Chen and Mooney (2011) and Chen (2012b), even though we use their semantic lexicon as our input. Their system deterministically builds a supervised training set by greedily selecting high-scoring lexemes, thus including only high-scoring lexemes during the training phase. In contrast, our probabilistic PCFG approaches also consider relatively low-scoring

but useful lexemes, thus covering more semantic concepts in the lexicon. The lexicon learning algorithms rely on correlational statistics, which may likely skip some semantic lexemes that occur in a few distinctive examples in the corpus. Because of this, it is not a good idea to systematically ignore low-scoring lexemes. This explains why our approaches perform particularly better in recall in the semantic parsing evaluations. Intuitively, we do a better job of utilizing all the semantic lexemes.

Also notable is that the Unigram Generation PCFG approach performs generally better than the Hierarchy Generation PCFG approach. We think this is due to the problem of hierarchical structure of LHG as well as to the complexity introduced by pseudo-lexemes. These decisions stem from the intuitive idea of complex concepts generating simpler concepts and are adopted in order to reduce the possible complexity of the resulting PCFG rules and to maintain feasibility. However, it turns out that such decisions work as noises at some point, and simple generation of semantic lexemes by the unigram Markov process provides an overall better model. In addition, the performance improvement brought about by the Unigram approach seems reduced in the word-segmented Chinese data. We speculate that the shorter average sentence lengths make the Hierarchy model less complicated due to fewer permutation rules. Therefore, the performance of the Hierarchy model is reasonably high, and the performance gain of the Unigram model appears less.

Although we have demonstrated our approaches for a fairly specific task, the navigation task, we can apply the general methodology to other

language grounding tasks, where an NL sentence is potentially connected to world states/events/actions expressed as a sequence/set of logical forms. Our approaches using PCFG induction with semantic lexicon are general PCFG frameworks for grounded language learning as long as an appropriate lexicon is provided, since the lexicon learning algorithm can be replaced with other domains.

4.7 Online Semantic Lexicon Learning

Our PCFG induction models are greatly affected by the quality of the semantic lexicon, since semantic lexemes are the basic building blocks for our model. It is interesting to see whether different lexicon learning algorithms would increase the overall performance of our models. In this section, we discuss a fast online lexicon learning algorithm called Subgraph Generation Online Lexicon Learning (SGOLL) proposed by Chen (2012a) and how it affects our models' performance in the evaluations. Chen (2012a, 2012b) demonstrated that SGOLL is relatively effective compared with the system by Chen and Mooney (2011) on the navigation task, but the learning process is much faster.

The algorithm by Chen and Mooney (2011) obtains candidate lexeme MRs for an NL phrase w by repeatedly taking intersections between MRs in the candidate lexeme MR set. Although it is quite effective for getting a *maximal* meaning for a phrase w, the learning process is slow. SGOLL is inspired by the fact that most words or short phrases correspond to small MR graphs;

	Precision	Recall	F1
Hierarchy PCFG + GILL	87.58	∗65.41	∗**74.81**
Hierarchy PCFG + SGOLL	∗89.04	61.06	72.30
Unigram PCFG + GILL	80.07	∗75.09	∗**77.49**
Unigram PCFG + SGOLL	∗81.58	67.89	74.10

Table 4.8: Semantic parsing results comparing models using different lexicon for English corpus, GILL (Chen & Mooney, 2011) and SGOLL (Chen, 2012b); '∗' denotes statistically significant difference compared to the second best results ($p < .05$).

thus, the algorithm focuses only on candidate meanings smaller than a certain size. The process collects co-occurrence information between n-grams w_j and connected subgraphs up to a certain size (in Chen and Mooney's paper, 3). Since each training example is processed only once, SGOLL results in a much faster learning process. The score of candidate lexemes is calculated by the same scoring function as in Chen and Mooney's (2011) study, and the final lexicon output is obtained by ranking the candidate lexemes with the scores.

The experimental results of using SGOLL with our two PCFG induction models on the English corpus are shown in Table 4.8 and Table 4.9. The Wilcoxon signed-rank test is performed for statistical significance and the significance is marked with $*$ ($p < .05$).

The results show that SGOLL does not improve the performance of our Hierarchy Generation model either in semantic parsing or in the subsequent plan execution. The primary reason is that since SGOLL is able to consider only small-sized lexemes, the entire LHG structure mainly stems from com-

	Single-sentence	Paragraph
Hierarchy PCFG + GILL	∗**57.22%**	∗**20.17%**
Hierarchy PCFG + SGOLL	55.01%	18.56%
Unigram PCFG + GILL	∗**67.14%**	∗**28.12%**
Unigram PCFG + SGOLL	56.69%	19.35%

Table 4.9: Plan execution rates using the MARCO execution module comparing models with different lexicon for English corpus, GILL (Chen & Mooney, 2011) and SGOLL (Chen, 2012b); '∗' denotes statistical significance compared to the second best results ($p < .05$).

posing pseudo-lexemes between SGOLL lexemes. This means that LHG with SGOLL may deviate from the real underlying semantics for composite NL phrases, thus showing worse performance overall. Moreover, SGOLL fails to enhance the Unigram Generation PCFG model also. In this case, the semantic lexemes obtained from the SGOLL algorithm represent pieces of the semantics of the context MR that are too small. Thus, the derived MRs from the parsing phase are constructed from small pieces of lexeme MRs, resulting in incorrect, unwanted constructions of final MRs. This results in quite a performance drop in all metrics.

4.8 Chapter Summary

We have proposed two novel methods for learning a semantic parser given only highly ambiguous supervision where each training example consists of one natural language sentence and a large full meaning representation whose subset refers only to true meaning. Our models are enhanced versions of

Börschinger et al.'s (2011) approach which reduces the problem of grounded learning of semantic parsers to PCFG induction. The major novelty of the proposed approaches stem from using a learned semantic lexicon to aid the construction of a smaller and more focused set of PCFG productions. This also allows our approaches to scale to complex MR languages that define a large (potentially infinite) space of representations for capturing the meaning of sentences. By contrast, the original PCFG approach (Börschinger et al., 2011) requires a finite MR language and its grammar grows intractably large for even moderately complex MR languages. In addition, our algorithm for composing MRs from the final parse tree provides the flexibility to produce a wide range of novel MRs that were not seen during training.

We have proposed two versions of such PCFG approaches whose differences depend mainly on whether PCFGs are constructed based on subgraph hierarchy structure among semantic lexemes. Surprisingly, the simpler version, which does not incorporate lexeme hierarchy and generates relevant semantic lexemes by the unigram Markov process, shows better performance in the evaluations.

Evaluations on a previous corpus of navigational instructions for virtual environments demonstrated the effectiveness of our methods compared to recent competing systems. In addition, our methods are shown to be effective in both the English and the Chinese versions of the corpus, which proves that our approaches are language-independent.

Chapter 5

Adapting Discriminatve Reranking to Grounded Language Learning

In this chapter, we describe how to adapt discriminative reranking to improve the performance of the generative models for grounded language learning. Specifically, we delve into the problem of navigational instruction following discussed in Chapter 4 and aid two PCFG models described earlier with the framework of discriminative reranking. Conventional methods of discriminative reranking require gold-standard references in order to evaluate candidates and update the model parameters in the training phase of reranking. However, grounded language learning problems do not have gold-standard references naturally available; therefore, direct application of conventional reranking approaches do not work. Instead, we show how the weak supervision of response feedback (e.g., successful task completion in the navigational task) can be used as an alternative, experimentally demonstrating that its performance is comparable and even more effective compared to training on gold-standard parse trees.

5.1 Chapter Overview

In Chapter 4, we reviewed the navigation task in the simulated virtual environment and proposed the two novel PCFG induction models for grounded language learning. The major challenge of the navigation task is the exponential ambiguity between an NL instruction and the matching MR contexts. The two approaches are the novel enhancements of the previous grounded language learning model using PCFG induction (Börschinger et al., 2011) in order to make it tractable for the complex problem of following navigation instructions. The observed sequence of actions provides very weak, ambiguous supervision for learning instructional language, since there are many possible ways to describe the same execution path. Although these two new approaches achieve much better performance than did the original studies of Chen and Mooney (2011) and Chen (2012b), they are still far below human performance.

Since the two approaches are essentially generative models where parameters are estimated by EM, *discriminative reranking* (Collins, 2000) is, potentially, one obvious choice to improve their performance. By training a discriminative classifier that uses global features of complete parses to identify correct interpretations, a reranker can significantly improve the accuracy of a generative model. Reranking has been successfully employed to improve a variety of tasks in natural language processing (Collins, 2002b; Lu et al., 2008; Ge & Mooney, 2006; Toutanova et al., 2005; Collins, 2002c). A standard reranking approach requires gold-standard reference interpretations (e.g., parse trees) to train the discriminative classifier. However, grounded language

learning problems do not provide gold-standard interpretations for the training examples in general. Instead, only the ambiguous perceptual context of the utterance is provided as weak supervision. The navigation task takes this weak supervision composed of the observed sequence of actions taken by a human when following an instruction, and thus it is impossible to directly apply conventional discriminative reranking approaches.

In the remainder of this chapter, we show how to adapt reranking to work with such weak supervision. We use the two PCFG induction approaches described in Section 4.3 and Section 4.4 as baseline generative models. Our proposed reranking model is used to discriminatively reorder the top parses produced by the two models. Instead of using gold-standard annotations to determine the correct interpretations and evaluate candidate representations during the training phase, we simply prefer to evaluate candidate interpretations by deciding whether the given candidate actually reaches the intended destination when executed in the world. Additionally, we extensively revise the features typically used in parse reranking tasks to work with the two PCFG approaches introduced in Chapter 4 in grounded language learning.

5.2 Modified Reranking Algorithm for Grounded Language Learning

In reranking, a baseline generative model is first trained and it generates a set of candidate outputs for each training example. Next, a second conditional model is trained using global features to rescore the candidates.

Reranking using an averaged perceptron (Collins, 2002a) has been successfully applied to a variety of NLP tasks. Therefore, we modify the averaged perceptron algorithm to rerank the parse trees generated by the two PCFG induction models introduced in Chapter 4. The approach requires three subcomponents: `1)` a GEN function that returns the top n candidate parse trees for each NL sentence produced by the generative model, `2)` a feature function Φ that maps an NL sentence, e, and a parse tree, y, into a real-valued feature vector $\Phi(e, y) \in R^d$, and `3)` a reference parse tree that is compared to the highest-scoring parse tree during training.

However, grounded language learning tasks, such as our navigation task, do not naturally provide reference parse trees for training examples. Therefore, our modified model replaces the gold-standard reference parse with the "pseudo-gold" parse tree whose derived MR plan is most successful at getting to the desired goal location in the virtual environment of the navigation task. This strategy can be easily extended and applied to other general grounded language learning tasks where there is a method of evaluating each candidate interpretation against the given perceptual environment. Thus, the third component in our reranking model becomes an evaluation function EXEC that maps a parse tree y into a real number representing the success rate (with regard to successfully reaching the intended destination in the virtual world) of the derived MR plan m composed from y.

Additionally, we improve the perceptron training algorithm by using multiple reference parses to update the weight vector $\bar{W}$. Although we de-

termine the pseudo-gold reference tree to be the candidate parse y^* such that $y^* = \arg\max_{y \in \text{GEN}(e)} \text{EXEC}(y)$, it may not actually be the correct parse for the sentence. Other parses may contain useful information for learning, and therefore we devise a way to update weights using *all* candidate parses whose successful execution rate is greater than the parse preferred by the currently learned model.

5.2.1 Response-Based Weight Updates

Since many grounded language learning tasks do not naturally come with a single gold-standard annotation for each example for the training purposes, we cannot obtain a gold-standard reference parse, as it is for a typical reranking case. In the navigation task, we cannot utilize the gold-standard MR plan in the training phase, either. Even though we have gold-standard MR plans only for the purposes of evaluation, it is impossible even to construct the actual full parse tree from a gold-standard MR due to the lack of corresponding relevant semantic lexemes.

To circumvent the need for gold-standard reference parses, we instead select a pseudo-gold parse from the candidates produced by the GEN function. In a similar vein, when reranking semantic parses, Ge and Mooney (2006) chose as a reference parse the one which was the most similar to the gold-standard semantic annotation. However, in the navigation task, the ultimate goal is to generate a plan that, when actually executed in the virtual environment, leads to the desired destination. Therefore, the pseudo-gold reference is chosen

as the candidate parse that produces the MR plan with the greatest execution success. This requires a module that evaluates the execution accuracy of the candidate parses. For the navigation task, we use the MARCO (MacMahon et al., 2006) execution module, which is also used to evaluate how well the overall system learns to follow directions (Chen & Mooney, 2011). Since MARCO is non-deterministic when executing underspecified plans, we execute each candidate plan 10 times, and its execution rate is the percentage of trials in which it reaches the correct destination. When there are multiple candidate parses tied for the highest execution rate, the one assigned the largest probability by the baseline model is selected. Therefore, the loss function L is a typical 0–1 loss function between the pseudo-gold parse $y*$ and the best candidate y predicted by the currently trained perceptron. Our modified averaged perceptron procedure with such a response-based update is shown in Algorithm 5.

One additional issue must be addressed when computing the output of the GEN function. The final plan MRs are produced from parse trees using compositional semantics (cf. two PCFG models explained in Section 4.3 and Section 4.4). Consequently, the n-best parse trees for the baseline model do not necessarily produce the n-best *distinct* plans, since many parses can produce the same plan. Therefore, we adapt the GEN function to produce the n-best distinct plans rather than the n-best parses. This may require examining many more than the n-best parses, because many parses have insignificant differences that do not affect the final plan. The score assigned to a plan

Algorithm 5 AVERAGED PERCEPTRON TRAINING WITH RESPONSE-BASED UPDATE

Input: A set of training examples (e_i, y_i^*), where e_i is an NL sentence and $y_i^* = \arg\max_{y \in \text{GEN}(e_i)} \text{EXEC}(y)$. $\text{EXEC}(y)$ is the execution rate of the MR plan m derived from parse tree y.

Output: The parameter vector $\bar{W}$, averaged over all iterations $1...T$

1: **procedure** PERCEPTRON
2: Initialize $\bar{W} = 0$
3: **for** $t = 1...T, i = 1...n$ **do**
4: $y_i = \arg\max_{y \in \text{GEN}(e_i)} \Phi(e_i, y) \cdot \bar{W}$
5: **if** $y_i \neq y_i^*$ **then**
6: $\bar{W} = \bar{W} + \Phi(e_i, y_i^*) - \Phi(e_i, y_i)$
7: **end if**
8: **end for**
9: **end procedure**

is the probability of the most probable parse that generates that plan. In order to efficiently compute the n-best plans, we modify the exact n-best parsing algorithm developed by Huang and Chiang (2005), which efficiently calculates n-best parses by only adding log-order computational complexity. The modified algorithm ensures that each plan in the computed n-best list produces a new distinct plan.

5.2.2 Weight Updates Using Multiple Parses

Typically, when used for reranking, the averaged perceptron updates its weights using the feature-vector difference between the current best predicted candidate and the gold-standard reference (line 6 in Algorithm 5). In our initial modified version, we replaced the gold-standard reference parse with the pseudo-gold reference, which has the highest execution rate among all the can-

didate parses. However, this ignores all the other candidate parses during the perceptron training. However, it is not ideal to regard other candidate parses as "useless." There may be multiple candidate parses with the same maximum execution rate, and even candidates with lower execution rates might represent the correct plan for the instruction given the weak, indirect supervision provided by the observed sequence of human actions.

Therefore, we also consider a further modification of the averaged perceptron algorithm which updates its weights using multiple candidate parses. Instead of only updating the weights with the single difference between the predicted and the pseudo-gold parses, the weight vector $\bar{W}$ is updated with the sum of feature-vector differences between the current predicted candidate and *all* the other candidates that have higher execution rates. Formally, in this version, we replace lines 5 through 7 to check and evaluate against all the other candidates. The modified algorithm is shown in Algorithm 6.

In the experiments shown in Section 5.4, we demonstrate that, by exploiting multiple reference parses, this new update rule helps achieve additional performance gain in the execution accuracy of the final system. Intuitively, this approach gathers additional information from all candidate parses with higher execution accuracy when learning the discriminative reranker. In addition, as shown in line 6 of Algorithm 6, it uses the difference in execution rates between a candidate and the currently preferred parse to weight the update to the parameters for that candidate. This allows more effective plans to have a larger impact on the learned model in each iteration.

Algorithm 6 AVERAGED PERCEPTRON TRAINING WITH WEIGHT UPDATE USING MULTIPLE PARSES

Input: A set of training examples (e_i, y_i^*), where e_i is an NL sentence and $y_i^* = \arg\max_{y \in \text{GEN}(e_i)} \text{EXEC}(y)$. $\text{EXEC}(y)$ is the execution rate of the MR plan m derived from parse tree y.

Output: The parameter vector $\bar{W}$, averaged over all iterations $1...T$

procedure PERCEPTRON
  Initialize $\bar{W} = 0$
  for $t = 1...T, i = 1...n$ **do**
    $y_i = \arg\max_{y \in \text{GEN}(e_i)} \Phi(e_i, y) \cdot \bar{W}$
    for all $y \in \text{GEN}(e_i)$ where $y \neq y_i$ and $\text{EXEC}(y) > \text{EXEC}(y_i)$ **do**
      $\bar{W} = \bar{W} + (\text{EXEC}(y) - \text{EXEC}(y_i))$
        $\times(\Phi(e_i, y) - \Phi(e_i, y_i))$
    end for
  end for
end procedure

5.3 Reranking Features

This section describes the features Φ extracted from the parses produced by the generative models and used to rerank the candidates. The two generative models in Section 4.3 and Section 4.4 share similar parse tree structures with semantic lexemes constituting nonterminals. Therefore, we use the same feature sets across these two baseline models in our experimental evaluations.

5.3.1 Base Features

The base features adapt those used in previous reranking methods, specifically those of Collins (2002a), Lu et al. (2008), and Ge and Mooney (2006), which are directly extracted from the parse trees produced by baseline

generative models. In addition, we also include the log probability of the parse tree as an additional feature, as did Lu et al. (2008). Figure 5.1 shows a sample full parse tree from our baseline model, which is used when explaining the reranking features below, each illustrated by an example.

a) PCFG Nonterminal. Indicates whether a PCFG nonterminal is used in the parse tree. The feature $f(\mathtt{L_1}) = 1$ in the example of Figure 5.1.

b) PCFG Rule. Indicates whether a particular PCFG rule is used in the parse tree: $f(\mathtt{L_1} \Rightarrow \mathtt{L_2L_3}) = 1$.

c) Grandparent PCFG Rule. Indicates whether a particular PCFG rule *as well as* the nonterminal above it is used in the parse tree: $f(\mathtt{L_3} \Rightarrow \mathtt{L_5L_6}|\mathtt{L1}) = 1$.

d) Long-range Unigram. Indicates whether a nonterminal has a given NL word below it in the parse tree: $f(\mathtt{L_2} \rightsquigarrow \mathbf{left}) = 1$ and $f(\mathtt{L_4} \rightsquigarrow \mathbf{turn}) = 1$.

e) Two-level Long-range Unigram. Indicates whether a nonterminal has a child nonterminal which eventually generates an NL word in the parse tree: $f(\mathtt{L_4} \rightsquigarrow \mathbf{left}|\mathtt{L_2}) = 1$

f) Unigram. Indicates whether a nonterminal produces a given child nonterminal or terminal NL word in the parse tree: $f(\mathtt{L_1} \rightarrow \mathtt{L_2}) = 1$ and $f(\mathtt{L_1} \rightarrow \mathtt{L_3}) = 1$.

g) Grandparent Unigram. Indicates whether a nonterminal has a given child

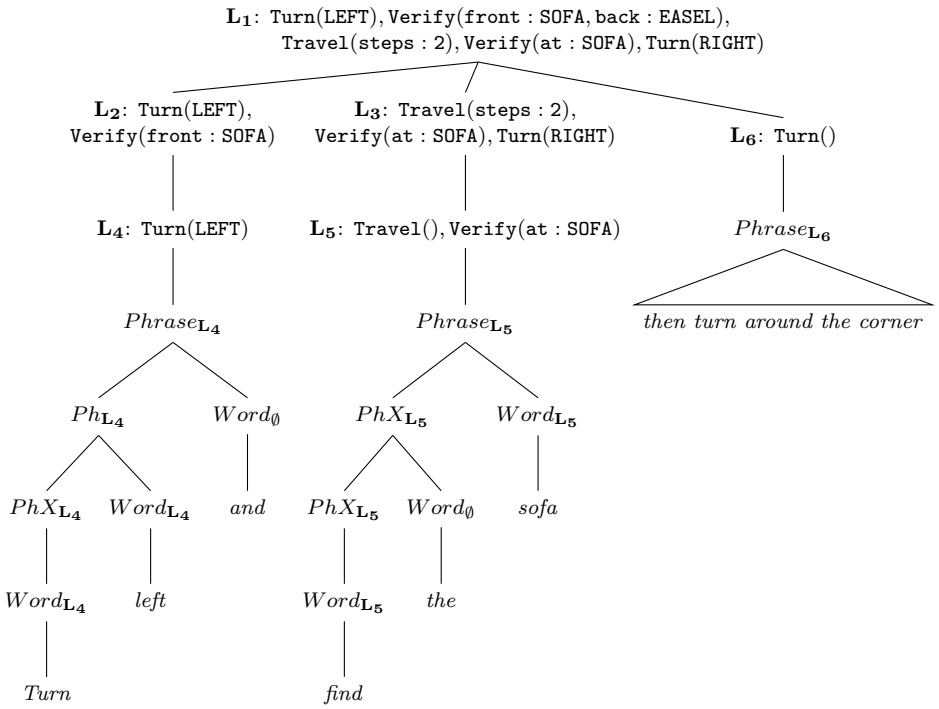


Figure 5.1: Sample full parse tree from our Hierarchy Generation PCFG model for the sentence, *"Turn left and find the soft then turn around the corner,"* used to explain reranking features. Nonterminals representing MR plan components are shown, labeled L_1 to L_6 for ease of reference. Additional nonterminals such as $Phrase$, Ph, PhX, and $Word$ are subsidiary ones for generating NL words from MR nonterminals. They are also shown in order to represent the entire process of how parse trees are constructed (for details, refer to Section 4.3).

nonterminal/terminal below it, as well as a given parent nonterminal: $f(\mathtt{L}_2 \rightarrow \mathtt{L}_4|\mathtt{L}_1) = 1$

h) Bigram. Indicates whether a given bigram of nonterminal/terminals occurs for given a parent nonterminal: $f(\mathtt{L}_1 \rightarrow \mathtt{L}_2 : \mathtt{L}_3) = 1$.

i) Grandparent Bigram. Same as Bigram, but also includes the nonterminal above the parent nonterminal: $f(\mathtt{L}_3 \rightarrow \mathtt{L}_5 : \mathtt{L}_6|\mathtt{L}_1) = 1$.

j) Log-probability of Parse Tree. Certainty assigned by the base generative model.

5.3.2 Predicate-Only Features

The base features above generally include the nonterminal symbols used in a parse tree. In the two PCFG models (Section 4.3 and Section 4.4) we use for the baseline models, the nonterminals are named after the components of the semantic representations (MRs), which are complex and numerous. There are roughly 2,500 to 3,000 nonterminals in the grammar constructed for the navigation data by the two baseline models, and most of the nonterminals are very specific and rare. This results in a very large, sparse feature space which can easily lead the reranking model to overfit the training data and prevent it from generalizing properly.

Therefore, we also tried to construct more general features that are less sparse. First, we constructed generalized versions of the base features in which nonterminal symbols use only predicate names and omit their arguments. In

the navigation task, action arguments frequently contain redundant, rarely used information. In particular, the interleaving verification steps frequently include many details that are never actually mentioned in the NL instructions. For instance, a nonterminal for the MR—

```
Turn(LEFT),
Verify(at:SOFA,front:EASEL),
Travel(steps:3)
```

—is transformed into the predicate-only form—

```
Turn(), Verify(), Travel()
```

—and then used to construct more general versions of the base features described in the previous section. Second, another version of the base features is constructed in which nonterminal symbols include action arguments but omit all interleaving verification steps. This is a somewhat more conservative simplification of the nonterminal symbols. Although verification steps sometimes help interpret the actions and their surrounding context, they frequently cause the nonterminal symbols to become unnecessarily complex and specific.

5.3.3 Descended Action Features

The final feature group we use in our reranking model captures whether a particular atomic action in a nonterminal "descends" into one of its child nonterminals or not. An atomic action consists of a predicate and its arguments, e.g., `Turn(LEFT)`, `Travel(steps:2)`, or `Verify(at:SOFA)`. When an

atomic action descends into lower nonterminals in a parse tree, it indicates that it is mentioned in the NL instruction and is therefore important. Below are several feature types related to descended actions that are used in our reranking model:

a) Descended Action. Indicates whether a given atomic action in a nonterminal descends to the next level. In Figure 5.1, $f(\texttt{Turn}(\texttt{LEFT})) = 1$ since it descends into L_2 and L_4.

b) Descended Action Unigram. Same as Descended Action, but also includes the current nonterminal: $f(\texttt{Turn}(\texttt{LEFT})|\texttt{L}_1) = 1$.

c) Grandparent Descended Action Unigram. Same as Descended Action Unigram, but additionally includes the parent nonterminal as well as the current one: $f(\texttt{Turn}(\texttt{LEFT})|\texttt{L}_2, \texttt{L}_1) = 1$.

d) Long-range Descended Action Unigram. Indicates whether a given atomic action in a nonterminal descends to a child nonterminal and this child generates a given NL word below it: $f(\texttt{Turn}(\texttt{LEFT}) \rightsquigarrow \textbf{left}) = 1$

5.4 Experimental Evaluation

5.4.1 Data and Methodology

The data we used for experimental evaluation is the navigation corpus discussed in Chapter 4. Again, we used the same experimental methodology as in Chapter 4, which follow that of the original study by Chen and Mooney

(2011), performing "leave one environment out" cross-validation, that is, three training trials on two environments and testing on the third.

In order to test the reranking performance, we use the two PCFG models proposed in Section 4.3 and Section 4.4 as two baseline models and test whether the reranking model can further improve the performances. First, a baseline model is trained on the training data of two environments; then, it is used to generate the 50-best plans for both training and testing instructions. As mentioned in Section 5.2.1, we need to generate many more top parse trees to get the final 50 distinct, formal MR plans. We limit the number of best parse trees to 1,000,000. Even with this high limit, some training examples were left with fewer than 50 distinct plans. Table 5.1 shows the statistics of examples per cross-validation split that produced fewer than 50 distinct candidate MR plans from GEN function evaluated on the two generative models. This is mostly due to exceeding the parse-tree limit and partly because the baseline model failed to parse some NL sentences. This result proves that our generative models produce a large number of duplicate parses that are only locally different among themselves. Although the statistics vary according to the datasets, the Unigram Generation PCFG model (Section 4.4) tends to produce more examples that have fewer than 50 distinct candidate MRs. Since the parse tree structure generated by the Unigram PCFG model is less complicated, more parse trees are locally different, which results in more duplicated derived MRs. Each candidate plan is then executed using MARCO, and its rate of successfully reaching the goal is recorded. Our reranking model

Data	Model	Grid-Jelly	Grid-L	L-Jelly	Average
English	Hierarchy PCFG	319	341	402	10.94%
	Unigram PCFG	991	1115	1087	32.89%
Chinese (Segmented)	Hierarchy PCFG	222	201	219	6.61%
	Unigram PCFG	803	965	851	26.98%
Chinese (Character)	Hierarchy PCFG	597	662	595	19.10%
	Unigram PCFG	1351	1440	1376	42.92%

Table 5.1: Statistics of examples that produced fewer than 50 distinct candidate MRs. Since the two baseline models produce different sets of candidate MRs from their own GEN functions, we present the results of each model separately. The statistics are gathered from both the training and testing data for each cross-validation split, and the total number of examples (single-sentence version) is 3236, which is the same for both English and Chinese data.

is trained on the training data containing the n-best candidate parses. We only retain reranking features that appear (i.e., have a value of 1) at least twice in the training data.

Finally, we measure both parse and execution accuracy on the test data. Parse accuracy evaluates how well a system maps novel NL sentences for new environments into the corresponding correct MR plans (Chen & Mooney, 2011). It is calculated by comparing the system's MR output to the gold-standard MR. Accuracy is measured using F1, the harmonic mean of precision and recall for individual MR constituents, thereby giving partial credit to approximately correct MRs. We then execute the resulting MR plans in the test environment to see whether they successfully reach the desired destinations. Execution is evaluated both for the single sentence and for the complete paragraph instructions. Successful execution rates are calculated by averaging 10 non-deterministic MARCO executions.

5.4.2 Reranking Results

5.4.2.1 Oracle results

As typical in reranking experiments, we first present results for an "oracle" that always returns the best result among the top-n candidates produced by the baseline system, thereby providing an upper bound on the improvements possible with reranking. Tables 5.2, 5.3, and 5.4 show the oracle accuracy for both semantic parsing and plan execution for the single sentence and complete paragraph instructions for various values of n. For the oracle parse accuracy, for each sentence, we pick the parse that gives the highest F1 score. For the oracle single-sentence execution accuracy, we pick the parse that gives the the highest execution success rate. These single-sentence plans are then concatenated to produce a complete plan for each paragraph instruction in order to measure the overall execution accuracy. Since making an error in *any* of the sentences in an instruction can easily lead to the wrong final destination, the paragraph-level accuracies are always much lower than the sentence-level ones. The high oracle accuracy shown in the tables indicates that significant improvement is possible if we train appropriate reranking methods. In order to balance the oracle accuracy and the computational effort required to produce n distinct plans, we chose $n = 50$ for the final experiments, since the oracle performance begins to asymptote at this point.

Model	n	1	2	5	10	25	50
Hierarchy PCFG	Parse Acc	74.81	79.08	82.78	85.32	87.52	88.62
	Single-sent	57.22	63.86	70.93	76.41	83.59	87.02
	Paragraph	20.17	28.08	35.34	40.64	48.69	53.66
Unigram PCFG	Parse Acc	76.45	80.66	84.42	86.14	88.21	89.18
	Single-sent	67.14	75.76	82.81	87.29	91.22	92.83
	Paragraph	28.36	38.65	47.50	54.27	62.55	65.72

Table 5.2: English results of oracle parse and execution accuracy for the single sentence and complete paragraph instructions for the n-best parses.

Model	n	1	2	5	10	25	50
Hierarchy PCFG	Parse Acc	75.53	79.08	83.32	85.78	88.06	89.37
	Single-sent	61.61	70.41	79.93	85.67	90.66	93.52
	Paragraph	18.97	30.09	41.31	48.09	56.79	62.68
Unigram PCFG	Parse Acc	76.41	80.89	84.17	85.99	87.88	88.75
	Single-sent	63.84	70.81	79.02	84.99	90.97	93.04
	Paragraph	22.87	29.37	39.86	47.66	58.50	61.46

Table 5.3: Word-segmented version of Chinese results of oracle parse and execution accuracy for the single sentence and complete paragraph instructions for the n-best parses.

Model	n	1	2	5	10	25	50
Hierarchy PCFG	Parse Acc	73.05	76.65	80.33	84.60	87.01	88.12
	Single-sent	56.01	63.70	71.61	81.65	87.42	90.80
	Paragraph	12.92	17.83	25.56	39.06	50.80	57.63
Unigram PCFG	Parse Acc	77.57	81.18	84.82	86.60	88.14	88.87
	Single-sent	63.24	70.11	77.92	84.28	89.13	90.96
	Paragraph	23.17	29.12	38.13	47.05	54.14	57.55

Table 5.4: Character-segmented version of Chinese results of oracle parse and execution accuracy for the single sentence and complete paragraph instructions for the n-best parses.

Baseline	Reranking	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	74.81	57.22	20.17
	`Gold`	**78.26**	52.57	19.33
	`Single`	73.32	59.65	22.62
	`Multi`	73.43	**62.81**	**26.57**
Unigram PCFG	`Baseline`	76.44	67.14	28.12
	`Gold`	**78.61**	66.52	26.85
	`Single`	77.24	68.27	**29.20**
	`Multi`	77.81	**68.93**	29.10

Table 5.5: English reranking results comparing our response-based methods using single (`Single`) or multiple (`Multi`) pseudo-gold parses to the standard approach using a single gold-standard parse (`Gold`). `Baseline` refers to the two PCFG models described in Sections 4.3 and 4.4 . Reranking results use all the features described in Section 5.3. "Single" means the single-sentence version, and "Para" means the full paragraph version of the corpus.

5.4.2.2 Response-based vs. gold-standard reference weight updates

Tables 5.5, 5.6, and 5.7 present the reranking results for our proposed response-based weight update (`Single`) for the averaged perceptron (cf. Section 5.2.1) compared to the typical weight update method using gold-standard parses (`Gold`). Since the gold-standard annotation gives the correct MR rather than a parse tree for each sentence, `Gold` selects as a single reference parse the candidate in the top 50 whose resulting MR is most similar to the gold-standard MR, as determined by its parse accuracy. Ge and Mooney (2006) employ a similar approach when reranking semantic parses.

The results show that our response-based approach (`Single`) clearly improves the performance of the baseline models of Hierarchy and Unigram

Baseline	Reranking	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	75.53	61.03	19.08
	`Gold`	**79.20**	56.62	17.25
	`Single`	77.26	64.12	21.29
	`Multi`	78.80	**64.15**	**21.55**
Unigram PCFG	`Baseline`	76.41	63.40	23.12
	`Gold`	**79.43**	64.48	23.28
	`Single`	77.74	65.64	23.74
	`Multi`	78.11	**66.27**	**25.95**

Table 5.6: Reranking results of the word-segmented version of the Chinese corpus comparing our response-based methods and the standard approach.

Baseline	Reranking	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	73.05	55.61	12.74
	`Gold`	**79.96**	61.43	20.86
	`Single`	76.26	**64.08**	22.25
	`Multi`	79.44	**64.08**	**22.58**
Unigram PCFG	`Baseline`	77.55	62.85	23.33
	`Gold`	**81.03**	64.93	23.97
	`Single`	79.76	65.50	25.35
	`Multi`	79.94	**66.84**	**27.16**

Table 5.7: Reranking results of the character-segmented version of the Chinese corpus comparing our response-based methods and the standard approach.

Generation PCFG models (Sections 4.3 and 4.4), particularly in the plan execution tasks, which is expected and desired because the reranking model is optimized for selecting the candidates with the best plan execution accuracies. In addition, `Single` performs generally better in the execution tasks than the standard approach using gold-standard parses (`Gold`). However, `Gold` does perform better on parse accuracy, since it explicitly focuses on maximizing the similarity accuracy of the resulting MR. In contrast, by focusing discriminative training on optimizing the performance of the ultimate end task, our response-based approach actually outperforms the traditional approach on the final task. In addition, it utilizes only feedback that is naturally available for the task, rather than requiring an expert to laboriously annotate each sentence with a gold-standard MR. Even though `Gold` captures more elements of the gold-standard MRs, it may miss some critical MR components that are crucial to the final navigation task. The overall result is very promising because it demonstrates how reranking can be applied to grounded language learning tasks where gold-standard parses are not readily available.

5.4.2.3 Weight update with single vs. multiple reference parses

Tables 5.5, 5.6, and 5.7 also show the performance when using multiple reference parse trees to update weights (cf. Section 5.2.2). Using multiple parses (`Multi`) shows generally better performance for all evaluation metrics, or at least performs comparably to using only a single parse (`Single`).

As explained in Section 5.2.2, the single-best pseudo-gold parse provides

weak, ambiguous feedback, since it only gives a rough estimate of the response feedback from the execution module. Using a variety of preferable parses to update weights provides a greater amount and variety of weak feedback and therefore leads to a more accurate model.[1]

For the tasks where `Multi` performs only comparably to `Single`, we conjecture that `Single` is already able to gain sufficient accurate information to classify the correct candidates. This seems to be because, for the examples that are correctly classified by the reranking model, the differences between the best and the other candidates are large enough that the additional information gain from other partially true candidates is negligible.

5.4.2.4 Comparison of different feature groups

Tables 5.8, 5.9, and 5.10 compare reranking results using the different feature groups described in Section 5.3. All the results shown in the tables used the weight update with multiple reference parses (`Multi`). Compared to the baselines of Hierarchy and Unigram Generation PCFG models, each of the feature groups—`Base` (base features), `Pred` (predicate-only and verification-removed features), and `Desc` (descended action features)—generally helps improve the performance of plan execution for both single sentence and complete paragraph navigation instructions in all the datasets.

[1]We also tried extending `Gold` to use multiple reference parses in the same manner, but this actually degraded its performance for all metrics. This indicates that, unlike `Multi`, parses other than the best one do not have useful information. Instead, in this case, additional parses seem to add noise to the training process. Therefore, updating with multiple parses does not appear to be useful in standard reranking.

Baseline Model	Features	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	**74.81**	57.22	20.17
	`Base`	71.50	60.09	23.20
	`Pred`	71.61	60.87	24.13
	`Desc`	73.90	61.33	25.00
	`Base+Pred`	69.52	61.49	26.24
	`Base+Desc`	73.66	61.72	25.58
	`Pred+Desc`	72.56	62.36	26.04
	`All`	73.43	**62.81**	**26.57**
Unigram PCFG	`Baseline`	**76.44**	67.14	28.12
	`Base`	78.22	67.93	26.13
	`Pred`	76.49	67.40	28.70
	`Desc`	76.71	68.08	29.43
	`Base+Pred`	77.84	69.05	28.93
	`Base+Desc`	78.14	**69.84**	30.18
	`Pred+Desc`	76.68	68.48	**30.60**
	`All`	77.81	68.93	29.10

Table 5.8: Reranking results comparing different sets of features using the two baseline models, Hierarchy and Unigram Generation PCFG models, on the English corpus. `Base` refers to base features (cf. Section 5.3.1), `Pred` refers to predicate-only features and also includes features based on removing interleaving verification steps (cf. Section 5.3.2), `Desc` refers to descended action features (cf. Section 5.3.3). `All` refers to all the features, including `Base`, `Pred`, and `Desc`. All results use weight update with multiple reference parses (cf. Section 5.2.2).

Baseline Model	Features	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	**75.53**	61.03	19.08
	`Base`	78.77	63.05	**23.69**
	`Pred`	75.35	57.75	15.16
	`Desc`	75.73	62.49	19.56
	`Base+Pred`	78.49	63.12	21.73
	`Base+Desc`	78.81	63.49	21.18
	`Pred+Desc`	77.19	62.44	17.70
	`All`	78.80	**64.15**	21.55
Unigram PCFG	`Baseline`	**76.41**	63.40	23.12
	`Base`	78.17	66.20	25.75
	`Pred`	76.56	64.07	22.98
	`Desc`	76.62	64.22	23.49
	`Base+Pred`	78.11	**66.50**	25.31
	`Base+Desc`	78.05	66.01	25.53
	`Pred+Desc`	76.82	64.55	23.23
	`All`	78.11	66.27	**25.95**

Table 5.9: Reranking results comparing different sets of features using the two baseline models, Hierarchy and Unigram Generation PCFG models, on the word-segmented Chinese corpus. All results use weight update with multiple reference parses.

Baseline Model	Features	Parse Acc	Plan Execution	
		F1	Single-sentence	Paragraph
Hierarchy PCFG	`Baseline`	**73.05**	55.61	12.74
	`Base`	78.81	63.44	22.54
	`Pred`	77.37	62.04	20.83
	`Desc`	72.85	55.84	15.51
	`Base+Pred`	78.24	60.22	20.87
	`Base+Desc`	78.60	63.88	**22.90**
	`Pred+Desc`	77.61	61.61	20.51
	`All`	79.44	**64.08**	22.58
Unigram PCFG	`Baseline`	**77.55**	62.85	23.33
	`Base`	80.23	66.76	26.82
	`Pred`	77.82	63.76	23.77
	`Desc`	78.01	63.96	23.81
	`Base+Pred`	80.18	**67.41**	26.89
	`Base+Desc`	79.97	67.12	**27.73**
	`Pred+Desc`	78.29	64.37	24.09
	`All`	79.94	66.84	27.16

Table 5.10: Reranking results comparing different sets of features using the two baseline models, Hierarchy and Unigram Generation PCFG models, on the character-segmented Chinese corpus. All results use weight update with multiple reference parses.

The most effective feature group differs from the dataset and the baseline model used. In general, `Base` tends to be the most effective feature group, considering the performances of each feature group. However, in the English corpus, `Pred` and `Desc` seem to outperform `Base`. The error analysis figures that the semantic lexemes learned for the English corpus tend to distribute well enough to cover the MR components of the context MRs with several lexemes. On the other hand, the semantic lexemes for both versions of the Chinese corpus tend to be quite skewed toward some popular chunks of the MR components within the context MRs. The reason is unclear, but we conjecture that the expressions of the Chinese translation are not as diverse those of the original English sentences, and thus that some common Chinese phrases are used a large number of times across the corpus. This difference results in more balanced, diverse parse structures in the English corpus from the two baseline models, and more skewed structures in the two Chinese corpora. Therefore, `Pred` and `Desc` in the two Chinese corpora provide features that are too general and that are not helpful in training an effective reranker, whereas `Pred` and `Desc` produce significant performance gain in the English corpus.

Combinations of the feature groups seem to help further improve the plan execution performance. However, using all of the feature groups (`All`) does not always result in optimal performance, because too many features make the reranking model tend to overfit to the training data. Although the best performance differs with each case, in general, combinations of two or more feature groups work the best.

5.5 Chapter Summary

In this chapter, we have shown how to adapt discriminative reranking to grounded language learning. Since typical grounded language learning problems, such as navigation instruction following, do not provide the gold-standard reference parses required by standard reranking models, we have devised a novel method for using the weaker supervision provided by response feedback when training a perceptron-based reranker. In the case of the navigation task, the response is expressed by how successfully the inferred navigation plans are executed and whether the desired goals in the virtual environment are reached. This approach was shown to be very effective compared to the traditional method of using gold-standard parses. In addition, since this response-based supervision is weak and ambiguous, we have also proposed a method for using multiple reference parses to perform the perceptron weight updates. With this approach, we have demonstrated significant additional improvement in end-task performance with this approach. Our experimental results show that our reranking approach achieves consistent improvement over the two baseline generative models described in the previous chapter. This indicates that our reranking approach is general and can be further applied in various other areas and applications.

Chapter 6

Related Work

In this chapter, we review some of the previous studies that are related to the models we have proposed. Section 6.1 first reviews studies of supervised semantic parsing and natural language generation. These two tasks are the core abilities that enable computers to communicate with humans using natural language interfaces. They are also directly related to our approach, which essentially learns semantic parsers in an ambiguous perceptual environment. Section 6.2 reviews other grounded language learning research that deals with ambiguous supervision and focuses mainly on how to resolve the ambiguity caused by the problem domains. Next, Section 6.3 presents earlier approaches that attempted to learn word meanings using statistical methods without considering deep knowledge about the syntactic or semantic structures of language. Section 6.4 describes the applications of learning from caption texts for images and videos. Finally, Section 6.5 reviews the applications in simulating robot movements in virtual environments by mapping the meanings of natural language instructions into the appropriate executable action sequences.

6.1 Learning for Semantic Parsing and Language Generation

Semantic parsing is the task of converting natural language sentences into appropriate logical forms that are easily interpreted by machines. Whereas syntax parsing discovers inherent structure in natural language, semantic parsing aims to find non-trivial structural relationship between natural language words and elements of logical forms. The challenge stems largely from the fact that natural language and logical forms do not always share structural similarities. Some words may correspond to a certain component of a logical form, whereas others may not have a true matching element. Thus, semantic parsing is a challenging task and has long been studied by many researchers in the natural language processing community.

Conventional semantic parsing approaches learn to map NL sentences to formal MRs via fully supervised training data consisting of NL/MR pairs (Zelle & Mooney, 1996; Zettlemoyer & Collins, 2005; Ge & Mooney, 2005; Kate & Mooney, 2006; Wong & Mooney, 2006, 2007b; Zettlemoyer & Collins, 2007; Lu et al., 2008; Zettlemoyer & Collins, 2009; Ge & Mooney, 2009; Kwiatkowski, Zettlemoyer, Goldwater, & Steedman, 2010, 2011). Such human annotated corpora are very expensive and difficult to build even for human experts, thus limiting the effectiveness of such conventional methods. Some semantic parser learners require additional syntactic annotations (Ge & Mooney, 2005), a syntactic parser trained with external domain knowledge (Ge & Mooney, 2009), or prior syntactic knowledge of the natural language (Zettlemoyer & Collins,

2005, 2007). Although such techniques are able to gain additional performance on learning semantics with the help of syntax, the utilization is limited in grounded language learning. Syntax cannot be obtained naturally from surrounding perceptual context, but can only be obtained from other external human intervention. Other research has attempted to model structural correspondences between NL and MRL grammar rules. In addition to the WASP (Wong & Mooney, 2006) and the Hybrid tree framework (Lu et al., 2008), on which our proposed models are built, Kate and Mooney (2006) used SVM classifiers trained for each MRL grammar rule to identify whether certain NL substrings indicate the MRL rule or not. Zettlemoyer and Collins (2009) showed that consideration of the context can further improve the semantic parsing results. Kwiatkowski et al. (2010) developed an online learning approach for learning semantic parsers but it still required one-to-one fully annotated corpora.

Language generation is the reverse process of semantic parsing, translating from MR to NL. Many recent systems solve this problem in the context of chart generation (Kay, 1996). Carroll, Copestake, Flickinger, and Poznanski (1999) and Carroll and Oepen (2005) proposed a chart generator for Head-Driven Phrase Structure Grammar (HPSG), whereas White and Baldridge (2003), White (2004), and White (2006) used Combinatory Categorial Grammar (CCG) for natural language generation. However, these systems based on chart generation focused only on how to properly order the NL words to make sensible sentences, not on the relationship between NL words and MR

elements so that the meanings can be realized. Wong and Mooney (2007b) proposed a natural language generation system called WASP^{-1} that inverted the semantic parsing system WASP (Wong & Mooney, 2006). WASP is a semantic parser learner that translates NL sentences into proper MRs using syntax-based statistical machine translation techniques with SCFG. Since an SCFG is symmetric with respect to the two languages it generates, the same trained model can be used for both semantic parsing (mapping NL to MR, as WASP does) and natural language generation (mapping MR to NL, as WASP^{-1} does). Similarly, Lu et al. (2008) showed that the hybrid tree framework can also be applied to the language generation task. By proposing the direct inversion model and the tree conditional random field model, Lu, Ng, and Lee (2009) showed that the inverted hybrid tree model performed better than WASP^{-1}.

Before generating an NL sentence from an MR with a language generation model, we first need to decide which MR to describe. This process is called *content selection* or *strategic generation*. It is the process of choosing *what to say*, as opposed to *surface realization* or *tactical generation*, which determines *how to say it*.

Chen and Mooney (2008) introduced IGSL for determining the most probable event types a human would comment on. Prior work on content selection by Duboue and McKeown (2003) proposed an automatic content selection from a corpus of text and associated semantics. They also showed that higher recall is preferred in the content selection task with the experiments of three proposed methods to infer rules from indirect observations. Zaragoza

and Li (2005) tried to solve the problem using reinforcement learning in a video game environment where the speaker's goal is to help the listener reach the destination. Their model found an optimal strategy so that it conveys the most appropriate information. In addition, Barzilay and Lapata (2005) approached content selection as a collective task. They found that consistently better output was achieved by considering all the content selection decisions jointly and finding dependencies between each uttered items.

6.2 Grounded Learning from Ambiguous Supervision

Conventional supervised settings for semantic parsing or language generation are not suitable for general purpose domains or large-scale tasks. Manually annotating each NL sentence with a complete MR is often prohibitively expensive for certain tasks. Instead, it is more desirable to train models on natural supervision where the meaning of a sentence can be explained by some subsets of the surrounding perceptual context that is automatically extracted. This kind of ambiguous supervision normally appears in the form of training data where each NL sentence is paired with a number of candidate MRs. Given such ambiguous supervision, the important challenge to overcome is finding the true semantic alignment of NL–MR out of many possibilities in order to learn effective semantic parsers or language generators.

Kate and Mooney (2007) extended a conventional semantic parser learner, KRISP (Kate & Mooney, 2006), to work with more relaxed supervision. The extension, KRISPER, learns from ambiguous training data where one NL sen-

tence has the true meaning among a small set of multiple MRs. This relaxation reflects a more natural and general learning environment. The hard EM algorithm of KRISPER alternates between finding the most probable one-to-one NL-MR matches based on parameters of the current iteration and updating the semantic parser with better estimates of the correct matches. While Kate and Mooney (2006) applied their approach only to an artificially generated ambiguous dataset, Chen and Mooney (2008) used this approach in the sportscasting corpus for the first time in order to disambiguate the relaxed supervision for learning an accurate semantic parser. They also introduced WASPER, which extended the supervised semantic parser learner, WASP, in order to work with ambiguous supervision. (Lu et al., 2008) employed the same extension as KRISPER using hard EM re-iterations of updating probabilistic counts. KRISPER and WASPER first train a corresponding initial semantic parser—KRISP and WASP, respectively—from the ambiguous training data by pairing each sentence with each of its candidate MRs. Then, the trained parser is used to select the most probable MR out of all the candidates. The algorithms iteratively improve the accuracies of both of the semantic parser and of the semantic alignment between NL–MR.

Börschinger et al. (2011) proposed a PCFG induction model for ambiguous supervision that our models in Chapter 4 extend. In their experiments in the sportscasting corpus, they showed even better performance than our approach in Chapter 3 by combining language-specific canonical word order encoded in their PCFG rules. Hajishirzi, Hockenmaier, Mueller, and Amir

(2011) presented an iterative approach for learning semantic parsers in ambiguous supervision with a little prior domain knowledge. Bordes, Usunier, and Weston (2010) viewed ambiguous supervision as a ranking problem. Since MR elements in the candidate sets are generally preferred, their approach learns a supervised ranking model that prefers those candidate MRs over other random MR elements.

Following the original work of Chen and Mooney (2011) on the navigation corpus, other researchers attempted to tackle the higher level of ambiguous supervision. Chen (2012a) applied the same methodology with a faster, more improved semantic lexicon to the same navigation task. However, the methodology also had the disadvantage of possible information losses while selecting lexicon entries with the greedy-covering algorithm during the refinement process. Chen also presented improved experimental results from the same model trained with additional data collected from crowdsourcing. Artzi and Zettlemoyer (2013) proposed a joint probabilistic model for simultaneously interpreting the meanings of natural language instructions and executing the corresponding actions in the navigation corpus. Their model used their own executor for evaluating the intermediate actions produced from the model and maximized the joint probability of parsing and execution by taking in immediate feedback from the environment. The idea of using environmental responses resembles our reranking model in Chapter 5. However, their model only works with the help of a small seed lexicon that contains core prior knowledge about the domain and the language.

Other research has focused more on the alignment of disambiguating the ambiguous data. Snyder and Barzilay (2007) proposed an alignment model between texts of American football game summaries with database entries containing statistics and events related to the game and the football players. However, their approach uses direct supervision of the correct correspondence between the text and the database records. A study by Liang et al. (2009) proposed a probabilistic generative approach that produces a Viterbi alignment between NL and MRs. They used a hierarchical semi-Markov generative model that first determines which facts to discuss and then generates words from the predicates and arguments of the chosen facts. However, they only addressed the alignment problem and were unable to parse new sentences into meaning representations or generate natural language from logical forms.

Other researchers have recently attempted to learn semantic parsers given only a weak supervision of *responses* (Clarke, Goldwasser, Chang, & Roth, 2010; Liang, Jordan, & Klein, 2011). Formal meaning representations (MRs) are easily understood and used for execution by machines. Thus, we could utilize the responses instead of full MRs as a weak indication of whether a certain intermediate MR output is correct or not during the learning process. This feedback drives the semantic parser learner toward more accurate internal parameter estimation. These response-driven semantic parsing models treat the formal MRs as latent variables to be estimated, and optimize the MR output for a novel NL sentence with respect to the known MR grammar structure, incorporating a small domain-specific knowledge. Our response-based

reranking approach in Chapter 5 shares the same general idea. Our approach drives the reranker with the weak indication of how candidate outputs lead to the desired destinations in the virtual environments without any form of gold-standard reference interpretations.

Other projects (Branavan et al., 2009; Branavan, Zettlemoyer, & Barzilay, 2010) learn to map natural language instructions to executable actions using reinforcement learning in Windows GUI and game environments. Their models learn the actions of the given instructions by observing how the environments react to the generated actions from the instructions. Assuming a reward function that assesses the quality of actions, their proposed models are able to learn the underlying meanings of low- and high-level instructions. Branavan et al. (2011) proposed a Monte-Carlo learning framework that learns to play in a complex computer game environment by incorporating a high-level text manual. They showed that compared to learning only by environment feedback, including high-level textual assistance can significantly improve the performance.

6.3 Learning Word Meanings from Ambiguous Supervision

One of the earliest studies on grounded language learning is by Siskind (1996). His approach solves referential uncertainties of words by capturing how the same word is uttered with the same perception. However, this approach only solves the ambiguity for the semantics of words and does not provide any

solution for compositional meanings.

Following Siskind, many researchers in robotics and computer vision tackled the problem of learning the grounded meanings of natural language words or short phrases from raw perceptual contexts (Bailey et al., 1997; Barnard et al., 2003; Roy, 2002; Yu & Ballard, 2004). However, most of these approaches have limited complexity on the side of natural language since their major challenge came from how to describe and abstract raw image descriptions. Consequently, they do not consider or exploit the syntactic or semantic structure of natural language while connecting it with perceptions. In contrast, our approach actively utilizes a single structure that covers both natural language and abstracted perceptions to understand the underlying semantics of language.

6.4 Learning from Images and Videos along with Relevant Texts

Recently, the computer vision community has paid a great deal of attention to learning from images with associated caption texts (Barnard et al., 2003; Bekkerman & Jeon, 2007; Duygulu, Barnard, de Freitas, & Forsyth, 2002; Gupta, Kim, Grauman, & Mooney, 2008; Li & Wang, 2008; Li, Socher, & Fei-Fei, 2009; Wang, Blei, & Li, 2009). These studies are mainly concerned with how words or short phrases of image/video descriptions can help the task of image classification or automatic scene annotation. First, visual contents are extensively extracted using state-of-the-art feature extraction techniques

from images or videos. The extracted features are then clustered to a few thousand similar categories in the vector space and normally they considered as a visual bag-of-words model. With or without considering the internal spatial relationship between the visual words, probabilistic models are learned in order to find alignment between visual words and natural language phrases or words. Although their descriptions and extracted features are extensive as regards their visual contents, their natural language captions consist of relatively simple, short phrases or sentences. In addition, their models for natural language captions do not utilize any linguistic cues, treating the text as a bag-of-words.

Beyond the static images, other researchers have focused on recognizing prominent activities while watching short video clips along with relevant descriptions in natural language (Fleischman & Roy, 2007; Laptev & Pérez, 2007; Fleischman & Roy, 2008; Laptev, Marszalek, Schmid, & Rozenfeld, 2008; Gupta & Mooney, 2010; Motwani & Mooney, 2012). Unlike scene or object understanding of static images, activity recognition of videos can be much harder because of background clutters or camera movements. In order to gain the useful information for classifying actions in the videos, the models utilize linguistic cues residing in the corresponding caption texts. Some models use a pure bag-of-words model for texts while others use more advanced linguistic information such as Part-Of-Speech (POS) tags, syntactic parsers, verb identifiers, and so on. These studies have proved in their experimental evaluations that incorporating textual information in activity recognition performs better

than relying only on visual classifiers.

Others recent work has focused on generating captions for image or videos. The goal is to generate a relevant caption text that describes the contents of a novel image or video. Li, Kulkarni, Berg, Berg, and Choi (2011) generated NL descriptive sentences given visual detections from static images including objects, attributes, and spatial relations. They first determined the subject, object, and verb from the learned model and then fused them together to form complete sentences using web-scale n-gram data. Farhadi, Hejrati, Sadeghi, Young, Rashtchian, Hockenmaier, and Forsyth (2010) presented a system that maps images and corresponding text descriptions into a semantic space consisting of a triplet of object, action, and scene. This system can be used in a two-way retrieval task, whether obtaining the relevant image after being given a text description or attaching an NL sentence to a given image. Feng and Lapata (2010) presented an approach that models a single topic distributions over clustered visual words and content NL words. Their language generation model selects the relevant text out of accompanying original texts in a news article corpus in order to generate captions. Barbu, Bridge, Burchill, Coroian, Dickinson, Fidler, Michaux, Mussman, Narayanaswamy, Salvi, Schmidt, Shangguan, Siskind, Waggoner, Wang, Wei, Yin, and Zhang (2012) suggested an approach to produce descriptions of short video clips using dynamic programming combined with Hidden Markov Models, while utilizing several hand-crafted templates to fit chosen components in language generation. Most recently, Krishnamoorthy, Malkarnenkar, Mooney, Saenko, and

Guadarrama (2013) presented a holistic data-driven approach for generating natural language descriptions for short Youtube videos. They first identified the best subject-verb-object triplet for a novel video clip and generated other relevant contents with a single template common for English. A set of generated candidate sentences were then ranked for the final output using a language model trained on an external web-scale n-gram corpus.

6.5 Learning for Robotics Applications

Understanding and interpreting the underlying meanings of natural language sentences are important in various robotics applications. Such capabilities enable robots to move about in the real environments given only natural language interfaces. Shimizu and Haas (2009) proposed a system that learned to interpret navigational instructions and guide a robot to the correct destinations in a simulated environment. However, they restricted the possible space of actions to be considered to 15 labels and transformed the entire parsing problem into a sequence labeling problem. Although this restriction worked well enough for the corpus they evaluated, it could be problematic when extended to other general navigation following domains, such as our navigation corpus discussed in Chapter 4. Matuszek, Fox, and Koscher (2010) presented a system that resembles the characteristics of our navigation task. Without assuming any prior linguistic knowledge, their system proposed to learn a semantic parser with the WASP system (Wong & Mooney, 2006) in order to understand the meanings of navigational instructions within their virtual

environment. However, the proposed environment is relatively simple and the associated natural language consists of simple direction following instructions without considering the surrounding environment. Vogel and Jurafsky (2010) used a more complex corpus of the HCRC Map Task Corpus (Anderson, Bader, Bard, Boyle, Doherty, Garrod, Isard, Kowtko, McAllister, Miller, Sotillo, Thompson, & Weinert, 1991). Their system used a reinforcement learning technique to penalize the agent when its route deviated from the desired path. The navigational instructions actively use the landmarks appearing in the environment and the system was evaluated as to whether the interpreted action sequences traversed the correct side and the order of visiting each landmarks in the desired actions. Kollar, Tellex, Roy, and Roy (2010) presented a navigation problem that was happening in a real office environment. They actually built a robot that was equipped with a laser range finder and cameras and collected visual information regarding the environment. The resulting semantic map was then used for simulating how their system interpreted navigational instructions. Tellex et al. (2011) proposed a probabilistic graphical model for connecting a particular natural language command to a compositional semantic structure. By collecting commands from Amazon Mechanical Turk, their system trained a computational model that inferred plans from natural language commands and then executed the plans in a robot simulator. Finally, Chao, Cakmak, and Thomaz (2011) presented an approach to train a socially interactive robot. While many other studies abstracted the raw perceptions that a robot would face in a form of semantic formal language,

they used the differences in the sensor readings as a possible meaning for natural language sentences. By demonstrating what the correct action was for a given instruction to the robot, they were able to build a system that trained the robot with natural language instructions interactively.

Although those approaches deal with full natural language sentences as linguistic inputs, their systems exploit excessive simplification of the semantic learning task by ignoring relevant objects in the environment, or they use external knowledge by assuming predefined spatial words, direct matchings between NL words and the names of objects and other landmarks in the MR, and/or an existing syntactic parser. In contrast, our work in the navigation task does not assume any prior linguistic knowledge, whether syntactic, lexical, or semantic, and must learn the mapping between NL words and phrases and the MR terms describing landmarks.

Chapter 7

Future Work

In this chapter, we describe future directions to pursue while extending and complementing the approaches presented in the previous chapters. Although we have gained a large improvement in performance by using the power of a probabilistic framework in the two previous grounded language learning tasks, sportscasting and navigation instruction following, there are many more possibilities in this field of study. We first discuss how to integrate syntactic information in our proposed models. After that, we consider extending our general approaches to work with large scale datasets, applying them to the area of machine translation, and modifying the ambiguous semantic parsing model to deal with real perception data.

7.1 Integrating Syntactic Components

The approaches discussed in earlier chapters do not incorporate the preprocessing of natural language texts before the models learn probabilistic mappings to MR components. This also means that we make no prior assumptions about the language itself and we have to learn the underlying meanings of the raw NL segments from scratch in a statistical manner.

The key ability of our methods for grounded language learning is semantic parsing. Under the ambiguous supervision of perceptual context, the main goal is to find the most probable correspondence between NL substrings and formal MR components which abstract surrounding perceptions. The major challenge of this process stems from the fact that raw natural language texts do not have visible formal structure as in logical meaning representations. This discrepancy often causes exponential ambiguity to resolve even in supervised semantic parsing settings.

Since the syntactic structures of natural language have been studied more thoroughly for a long time from various perspectives, the integration of syntactic structure learning could significantly benefit grounded language learning. In our presented models discussed in Chapters 3 and 4, the n-gram model is used for deciding the most probable meaningful boundaries of NL substrings to match with MR components. However, certain common n-grams are frequently used, but often they do not constitute any considerable units of meaning, such as phrases, but instead are composed of prepositions and/or determiners. On the other hand, syntactic categories in syntactic parsing such as NP or VP, referring noun phrase and verb phrase respectively, separate an NL sentence into more meaningful subparts. In these circumstances, such syntactic information can facilitate matching formal logical structure with raw natural language strings.

First, the hierarchical structure of syntactic categories obtained from syntactic parsing could be integrated into the formal MR structure. Ge and

Mooney (2005) proposed a supervised statistical semantic parsing technique that integrates syntax and semantics. The suggested semantically augmented parse tree (SAPT) extends a tree-structured relationship between formal MR and NL sentences by interleaving the additional syntactic categories involved. Such syntactic hierarchies intuitively resemble an LHG (see Chapter 4), and it is possible to combine both structures to utilize both cues for an improved model. However, this additional syntactic information is obtained by existing syntactic parsers utilizing external knowledge. Instead of incorporating such external supervision, a more preferred goal would be a joint probabilistic model of learning syntactic and semantic structure at the same time in an unsupervised manner.

Rather than integrating a complex full syntactic structure, it is also possible to integrate part-of-speech (POS) tags into the generative process of our proposed models. In particular, incorporating POS tag categories inside the generative process of our presented models would provide an additional layer to the correspondences of NL words and MR components, which could reduce the complexity of the resulting models. This would be of benefit when we apply our probabilistic models to more large-scale problems. For instance, POS-tagged natural language words will have limited choices for which lexeme MRs or atomic MR constituents they can be matched to. Guo and Mooney (unpublished) used POS tags only for filtering out inappropriate lexicon entries. However, POS tags can also be integrated in the generative process itself within the PCFG framework so that lexeme MRs or MR grammar rules

first generate appropriate POS tags and then subsequently produce the corresponding NL words. This process will jointly learn unsupervised POS tags as well as the PCFG structure that connects MR components and/or rules to NL words.

7.2 Learning in Large-Scale Data

Our proposed models in earlier chapters dealt mainly with two publicly available domains: the sportscasting task and the navigational instruction following task. Even though both of these domains are well-designed for grounded language learning tasks and simulate a reasonable level of ambiguity that would occur in the other real world problems, the size of whole dataset is relatively small, containing only a few thousand example pairs.

With the rise of crowdsourcing methodology, it has become easy these days to get annotated data with human labor involved. Chen (2012a, 2012b) has shown that additional data collection and annotation can be accomplished quite handily using Amazon Mechanical Turk. Besides the original data created for navigation task, he additionally gathered navigation instructions and follower traces given the other parts. Annotations from crowdsourcing might be very noisy, considering that the human annotators are randomly selected by the crowdsourcing system. However, this definitely reduces annotation cost, and thus makes it easy to extend the scale of our grounded learning methods.

In addition, crowdsourcing for grounded language learning tasks is much easier than that for supervised semantic parsing tasks. Since annotat-

ing fully supervised NL–MR pairs for a semantic parsing task often requires expert knowledge both of NL and of MRL, random annotators from crowdsourcing are not suitable in many cases. On the other hand, we can design easier annotation tasks for crowdsourcing annotators for grounded language learning which do not need any complicated prior knowledge. For example in the navigation task, we can ask annotators either to follow a given instruction in an environment or to write an appropriate instruction by observing a sample action sequence. In terms of scaling up the annotated training data, grounded language learning tasks are more preferable.

The data collection can be further improved with more interactive system that could engage the attention of the general public. Online systems such as the LabelMe (Russell, Torralba, Murphy, & Freeman, 2008) or the ESP Game (von Ahn & Dabbish, 2004) collect a large amount of very relevant data using game-style interactive user interfaces. For our navigation task, similar systems can be designed in which two people are playing so that one person teaches how to get to the destination and the other follows the natural language instruction toward the desired goal while recording the traces. By alternating roles of instructor and follower, this framework can be constructed as a social game in which each person gets points for successful task completion.

Based on the collected large-scale data, our model needs to be adapted to deal with such increased complexity in the training data. Our current models run relatively slow even though the amount of training data is in the order of thousands of examples. In order to maintain feasibility in the larger

datasets, our models need to be simplified.

7.3 Machine Translation

Semantic parsing is a particular form of machine translation from a source natural language to a target meaning representation language. WASP (Wong & Mooney, 2006) is inspired by this notion and builds a semantic parsing model based on state-of-the-art syntax-based statistical machine translation (SMT) (Chiang, 2005). In addition, it is also noteworthy that many existing semantic parsing approaches implicitly use the structural similarity between NL and MR. In this sense, it would be interesting to apply our semantic parsing methods back to the SMT tasks. Conventional SMT tasks resemble conventional supervised semantic parsing tasks, since the SMT approaches require a sentence aligned parallel corpus to train the models. Our navigation task with highly ambiguous supervision is analogous to the task of "summarized translation." In the navigation task, our PCFG model finds the probabilistic correspondences between the semantic concepts represented by lexeme MRs and the NL phrases, in which only some subparts of the given context (the full landmarks plan) are referred to by the given NL sentence. In the summarized translation task, a rich text in a source language is translated to a more concise text in a target language containing only the gist of the original text. For example, many Wikipedia articles are published in many languages, but the contents are usually not parallel between languages. Contents in some languages are rich in details, but contents in other languages

may contain only the gist of the topic. Currently, such summarized translation has not been investigated to our knowledge. However, beyond SMT trained on parallel corpora, there have been various attempts at using "comparable corpora," which are collections of documents that are comparable and similar in content and form in various degrees and dimensions (Diab & Finch, 2000; Munteanu & Marcu, 2005; Snover, Dorr, & Schwartz, 2008). However, the documents in comparable corpora usually have similar levels of complexity and thus the task has very different characteristics from summarized translation. Summarized translation can be thought of as the ambiguous correspondence problem between two languages. Only some subparts of the rich language text are translated to the concise target language. Based on the idea of our PCFG induction model that finds rich-to-concise correspondences, we could extend and apply our model to the summarized translation task in conjunction with SMT techniques.

7.4 Real Perceptual Data

Our proposed models in Chapters 3 and 4 assumed for simplicity's sake that we had abstracted the information of the surrounding world states which was ambiguously connected to natural language sentences. An obvious extension of our model would be to learn directly from real perceptional data instead of from abstracted logical representations. However, we first need a way to select what raw information is important and notable to be learned with natural language. A recent study by Chao et al. (2011) investigated how

to train a socially interactive robot through demonstration. They grounded the meaning of natural language by the state changes that appeared in the sensor readings. Even though their work limited the tasks to several categories and the complexity of natural language used is simple, it is notable that they were able to train a machine learning model which directly connected natural language instruction to the changes in real-valued raw sensory data. In a similar way, we can construct a semantic lexicon composed of state changes in sensor data, and then our PCFG induction model can be applied to further investigate NL–MR groundings with proper modifications. In addition, with a provided object recognition system, our proposed system can identify surrounding environments as well as notable landmarks to help interpret natural language instructions and follow them to navigate in the real environments as in other navigation work (Shimizu & Haas, 2009; Matuszek et al., 2010; Vogel & Jurafsky, 2010; Kollar et al., 2010; Tellex et al., 2011). Moreover, we can also extend and apply our methods to visual data such as images and videos. Extracted features such as SIFT (Lowe, 1999) and space-time interest points (Laptev, 2005) in images and videos will produce vectors that can be used to describe notable objects or landmarks. Then, our model will automatically discover how certain visual features are probabilistically related to natural language.

Chapter 8

Conclusion

The ultimate goal of grounded language learning for computational systems is to mimic the process of human language learning. In many cases, it is reduced to the problem of learning the underlying semantics of languages in ambiguous perceptual environments. This type of learning approach has gained growing attention recently, since there is no need for explicit human intervention to acquire training data. By contrast, conventional semantic learning methods, such as supervised semantic parsing, require costly, hard to acquire, one-to-one full supervision of natural language sentences and the corresponding logical forms.

In this thesis, we reviewed two grounded language learning problems with such natural, ambiguous supervision. The RoboCup sportscasting task and the navigation task are publicly available datasets collected from the automatic extraction of world state information associated with corresponding natural language texts. Prior grounded language learning studies on these datasets are vulnerable to information loss through failing to exploit the probabilistic nature of connections between NL and MR, or through only focusing on semantic alignments and not on learning the underlying semantics of lan-

guages. In contrast, our proposed probabilistic models use generative processes to select NL and MR components probabilistically, simultaneously learning the semantic alignment and meanings of natural languages.

Although the sportscasting domain incurs 1-to-N limited ambiguity that can be solved by an additional layer of selection model on top of generative semantic parsing approaches, the navigation domain brings up the much harder challenge of potentially exponential ambiguity where a given NL sentence can refer to a certain subset of all possibly relevant surrounding perceptions. Such complexity is reduced using a pre-learned semantic lexicon based on co-occurrence statistics, and further employed in our two presented PCFG induction models, constituting generative framework. The experimental results show the effectiveness of our methods compared to previous studies of the same domains and also prove our methods are language-independent.

In addition, in order to further improve the performance of the proposed generative models, we showed how discriminative reranking can be applied in grounded language learning. It is a non-trivial extension, because grounded language learning problems do not naturally have gold-standard references for each training example that can normally be used during comparison and evaluation in reranking algorithms. Instead, the reranking model is trained with the weak responses of evaluating how well each candidate MR plan reaches the intended goal, which can be naturally obtained from interactions with perceptual context. Our experimental results proved that this approach enables discriminative reranking without the need for explicit gold-standard reference

interpretations.

We have only scratched the surface of grounded language learning problems so far. There are a potentially infinite number of language understanding problems that need to be resolved. Moreover, natural language evolves and adapts to our constantly changing world as the life of humans changes rapidly in this modern era. What is required of a computerized language understanding system in this situation is to automatically adapt to the changing nature of language. In addition, since it is impossible to create a domain-specific system for every possible problems in the world, it is also imperative that we have more general, language-independent learning systems such as the models proposed in the present study. Above all, resolving the central challenge of disambiguating the weak supervision of language and perceptual contexts is the central key to solving the language understanding problems that have been studied for a long time.

Appendices

Appendix A

Details of the Sportscasting Data

Table A.1 shows the detailed statistics of the English and Korean sportscasting datasets per each game collected. Some NL comments do not have correct MRs associated with them and are essentially noise in the training data (18% of the English dataset and 8% of the Korean dataset). Each comment has, on average, more than two possible events to be matched, which means that over half of these possible connections between NL and MRs are incorrect and noisy. Since the 2001 final game was double overtime, there are almost double the number of events in the 2001 games.

	# events	# comments			# events / comment		
		Total	MRs	C.MRs	Max	AVG	STD
English dataset							
2001 final	4003	722	671	520	9	2.24	1.64
2002 final	2223	514	458	376	10	2.40	1.65
2003 final	2113	410	397	320	12	2.85	2.05
2004 final	2318	390	342	323	9	2.73	1.70
Korean dataset							
2001 final	4003	673	650	600	10	2.14	2.08
2002 final	2223	454	444	419	12	2.49	3.08
2003 final	2113	412	396	369	10	2.55	3.67
2004 final	2318	460	423	375	9	2.60	2.59

Table A.1: Detailed statistics for each game in the English and Korean sportscasting datasets. MRs refers to the number of NL comments that have matching MRs, and C.MRs refers to the number of NL comments that have correct MRs.

Appendix B

Details of the Navigation Data

The navigation task was originally proposed and designed by MacMahon et al. (2006). Their original dataset consists of the English instructions and the corresponding follower data. Later, Chen and Mooney (2011) created the single-sentence version of the corpus by annotating each sentence instruction along with suitable actions. Chen (2012a) then presented the Mandarin Chinese version of the data and evaluated their methods on both languages.

Figures B.1, B.2, B.3 show the three top view maps of the three virtual worlds from the navigation tasks, Grid, L, and Jelly, respectively. Each virtual world consists of seven hallways with different floor patterns: grass, brick, wood, gravel, blue, floral, and yellow octagons. Each world is again divided into three areas with different wall paintings: butterfly, fish, and Eiffel Tower. At the intersections of the hallways, furniture, such as a hat rack, a lamp, a chair, a sofa, a bar stool, and an easel, is placed, and these items are used extensively in describing route instructions. Each piece of furniture is marked with its corresponding capital letters in the maps.

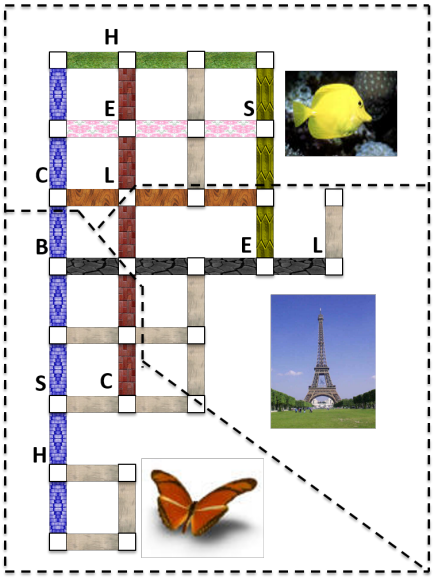


Figure B.1: Top view map of Grid

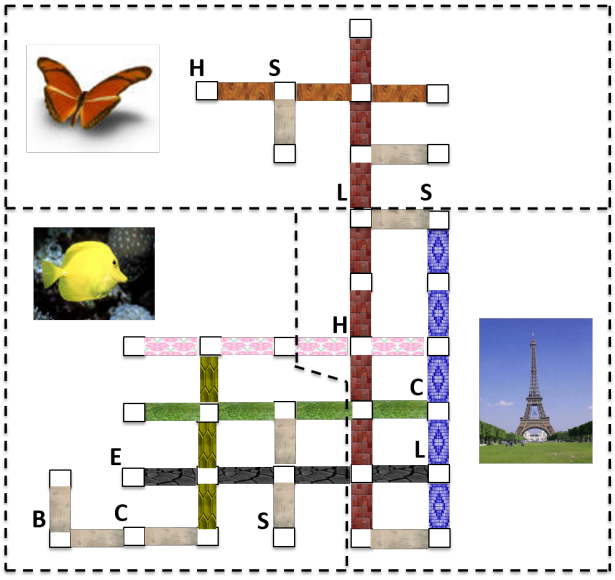


Figure B.2: Top view map of L

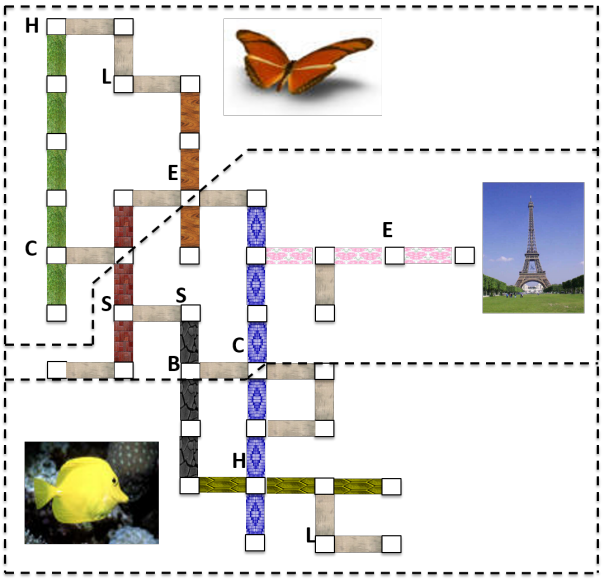


Figure B.3: Top view map of Jelly

Bibliography

Aho, A. V., & Ullman, J. D. (1972). *The Theory of Parsing, Translation, and Compiling*. Prentice Hall, Englewood Cliffs, NJ.

Anderson, A., Bader, M., Bard, E., Boyle, E., Doherty, G. M., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H. S., & Weinert, R. (1991). The HCRC map task corpus. *Language and Speech*, *34*, 351–366.

Artzi, Y., & Zettlemoyer, L. (2013). Weakly supervised learning of semantic parsers for mapping instructions to actions.. Vol. 1, pp. 49–62. Association for Computational Linguistic.

Bailey, D., Feldman, J., Narayanan, S., & Lakoff, G. (1997). Modeling embodied lexical development. In *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*.

Barbu, A., Bridge, A., Burchill, Z., Coroian, D., Dickinson, S. J., Fidler, S., Michaux, A., Mussman, S., Narayanaswamy, S., Salvi, D., Schmidt, L., Shangguan, J., Siskind, J. M., Waggoner, J. W., Wang, S., Wei, J., Yin, Y., & Zhang, Z. (2012). Video in sentences out. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI-2012)*, pp. 102–112.

Barnard, K., Duygulu, P., Forsyth, D., de Freitas, N., Blei, D. M., & Jordan, M. I. (2003). Matching words and pictures. *Journal of Machine Learning Research*, *3*, 1107–1135.

Barzilay, R., & Lapata, M. (2005). Collective content selection for concept-to-text generation. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP-05)*.

Bekkerman, R., & Jeon, J. (2007). Multi-modal clustering for multimedia collections.. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR-2007)*. IEEE Computer Society.

Bordes, A., Usunier, N., & Weston, J. (2010). Label ranking under ambiguous supervision for learning semantic correspondences. In *ICML*, pp. 103–110.

Börschinger, B., Jones, B. K., & Johnson, M. (2011). Reducing grounded learning tasks to grammatical inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pp. 1416–1425, Stroudsburg, PA, USA. Association for Computational Linguistics.

Branavan, S., Chen, H., Zettlemoyer, L. S., & Barzilay, R. (2009). Reinforcement learning for mapping instructions to actions. In *Joint Conference of the 47th Annual Meeting of the Association for Computational*

Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP), Singapore.

Branavan, S., Silver, D., & Barzilay, R. (2011). Learning to win by reading manuals in a monte-carlo framework. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-11)*.

Branavan, S., Zettlemoyer, L., & Barzilay, R. (2010). Reading between the lines: Learning to map high-level instructions to commands. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 1268–1277. Association for Computational Linguistics.

Brown, P. F., Cocke, J., Della Pietra, S. A., Della Pietra, V. J., Jelinek, F., Lafferty, J. D., Mercer, R. L., & Roossin, P. S. (1990). A statistical approach to machine translation. *Computational Linguistics*, *16*(2), 79–85.

Brown, P. F., Della Pietra, V. J., Della Pietra, S. A., & Mercer, R. L. (1993). The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, *19*(2), 263–312.

Carroll, J., Copestake, A., Flickinger, D., & Poznanski, V. (1999). An efficient chart generator for (semi-) lexicalist grammars. In *Proceedings of the 7th European workshop on natural language generation (EWNLG9)*, pp. 86–95.

Carroll, J., & Oepen, S. (2005). High efficiency realization for a wide-coverage unification grammar. In *Proceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP-2005)*, pp. 165–176, Jeju Island, Korea.

Chang, P.-C., Galley, M., & Manning, C. (2008). Optimizing Chinese word segmentation for machine translation performance. In *Proceedings of the ACL Third Workshop on Statistical Machine Translation*.

Chao, C., Cakmak, M., & Thomaz, A. (2011). Towards grounding concepts for transfer in goal learning from demonstration. In *Development and Learning (ICDL), 2011 IEEE International Conference on*, Vol. 2, pp. 1–6. IEEE.

Charniak, E., & Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pp. 173–180, Ann Arbor, MI.

Chen, D. L. (2012a). Fast online lexicon learning for grounded language acquisition. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL-2012)*, Jeju, Republic of Korea.

Chen, D. L. (2012b). *Learning Language from Ambiguous Perceptual Context*. Ph.D. thesis, Department of Computer Science, University of Texas at Austin.

Chen, D. L., Kim, J., & Mooney, R. J. (2010). Training a multilingual sportscaster: Using perceptual context to learn language. *Journal of Artificial Intelligence Research*, *37*, 397–435.

Chen, D. L., & Mooney, R. J. (2008). Learning to sportscast: a test of grounded language acquisition. In *ICML '08: Proceedings of the 25th International Conference on Machine Learning*, pp. 128–135, New York, NY, USA. ACM.

Chen, D. L., & Mooney, R. J. (2011). Learning to interpret natural language navigation instructions from observations. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-2011)*, San Francisco, CA, USA.

Chiang, D. (2005). A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pp. 263–270, Ann Arbor, MI.

Clarke, J., Goldwasser, D., Chang, M.-W., & Roth, D. (2010). Driving semantic parsing from the world's response. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning (CoNLL-2010)*, pp. 18–27, Uppsala, Sweden. Association for Computational Linguistics.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000)*, pp. 175–182, Stanford, CA.

Collins, M. (2002a). Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*, Philadelphia, PA.

Collins, M. (2002b). New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pp. 263–270, Philadelphia, PA.

Collins, M. (2002c). Ranking algorithms for named-entity extraction: Boosting and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pp. 489–496, Philadelphia, PA.

Collins, M., & Koo, T. (2005). Discriminative reranking for natural language parsing. *Computational Linguistics*, *31*(1), 25–69.

Diab, M., & Finch, S. (2000). A statistical word-level translation model for comparable corpora. In *Proceedings of the Conference on Content-based Multimedia Information Access (RIAO)*.

Duboue, P. A., & McKeown, K. R. (2003). Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the*

2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-03), pp. 121–128.

Duygulu, P., Barnard, K., de Freitas, N., & Forsyth, D. (2002). Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision*, pp. 97–112, London, UK. Springer-Verlag.

Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., & Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. In *Proceedings of the 11th European Conference on Computer Vision (ECCV-10)*.

Feng, Y., & Lapata, M. (2010). How many words is a picture worth? Automatic caption generation for news images. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*.

Fleischman, M., & Roy, D. (2007). Situated models of meaning for sports video retrieval. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT-07)*, Rochester, NY.

Fleischman, M., & Roy, D. (2008). Grounded language modeling for automatic speech recognition of sports video. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL-08)*.

Fraser, A., & Marcu, D. (2006). Semi-supervised training for statistical word alignment. In *Proceedings of the 21st International Conference on Com-*

putational Linguistics and the 44th annual meeting of the Association for Computational Linguistics (ACL-06), pp. 769–776, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ge, R., & Mooney, R. J. (2005). A statistical semantic parser that integrates syntax and semantics. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pp. 9–16, Ann Arbor, MI.

Ge, R., & Mooney, R. J. (2006). Discriminative reranking for semantic parsing. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, Sydney, Australia.

Ge, R., & Mooney, R. J. (2009). Learning a compositional semantic parser using an existing syntactic parser. In *Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP)*, Singapore.

Gold, K., & Scassellati, B. (2007). A robot that uses existing vocabulary to infer non-visual word meanings from observation. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)*.

Guo, L., & Mooney, R. J. (2012). Using part-of-speech to aid grounded learning of word meanings..

Gupta, S., Kim, J., Grauman, K., & Mooney, R. (2008). Watch, listen & learn: Co-training on captioned images and videos. In *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD-08)*, pp. 457–472, Antwerp, Belgium.

Gupta, S., & Mooney, R. J. (2010). Using closed captions as supervision for video activity recognition. In *Proceedings of the 25th AAAI Conference on Artificial Intelligence (AAAI-10)*, Atlanta.

Hajishirzi, H., Hockenmaier, J., Mueller, E. T., & Amir, E. (2011). Reasoning about robocup soccer narratives.. In Cozman, F. G., & Pfeffer, A. (Eds.), *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI-2011)*, pp. 291–300. AUAI Press.

Huang, L. (2008). Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pp. 586–594, Columbus, Ohio. Association for Computational Linguistics.

Huang, L., & Chiang, D. (2005). Better k-best parsing. In *Proceedings of the Ninth International Workshop on Parsing Technology*, Parsing '05, pp. 53–64, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jurafsky, D., & Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ.

Kate, R. J., & Mooney, R. J. (2006). Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, pp. 913–920, Sydney, Australia.

Kate, R. J., & Mooney, R. J. (2007). Learning language semantics from ambiguous supervision. In *Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI-07)*, pp. 895–900, Vancouver, Canada.

Kay, M. (1996). Chart generation. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics (ACL-96)*, pp. 200–204, San Francisco, CA.

Kim, J., & Mooney, R. J. (2010). Generative alignment and semantic parsing for learning from ambiguous supervision. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pp. 543–551. Association for Computational Linguistics.

Kim, J., & Mooney, R. J. (2012). Unsupervised PCFG induction for grounded language learning with highly ambiguous supervision. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Natural Language Learning*, EMNLP-CoNLL '12.

Kim, J., & Mooney, R. J. (2013). Adapting discriminative reranking to grounded language learning. In *Proceedings of the 51st Annual Meet-*

ing of the Association for Computational Linguistics (ACL-2013), Sofia, Bulgaria.

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

Konstas, I., & Lapata, M. (2012). Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pp. 369–378, Stroudsburg, PA, USA. Association for Computational Linguistics.

Krishnamoorthy, N., Malkarnenkar, G., Mooney, R. J., Saenko, K., & Guadarrama, S. (2013). Generating natural-language video descriptions using text-mined knowledge. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI-2013)*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., & Steedman, M. (2010). Inducing probabilistic ccg grammars from logical form with higher-order unification. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pp. 1223–1233. Association for Computational Linguistics.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., & Steedman, M. (2011). Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the Conference on Empirical Methods in Natural Lan-*

guage Processing, EMNLP '11, pp. 1512–1523, Stroudsburg, PA, USA. Association for Computational Linguistics.

Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision*, *64*(2-3), 107–123.

Laptev, I., Marszalek, M., Schmid, C., & Rozenfeld, B. (2008). Learning realistic human actions from movies. In *Proceedings of the 21st IEEE Conference on Computer Vision and Pattern Recognition (CVPR-08)*.

Laptev, I., & Pérez, P. (2007). Retrieving actions in movies. In *Proceedings of the IEEE 11th International Conference on Computer Vision (ICCV)*, pp. 1–8.

Li, J., & Wang, J. Z. (2008). Real-time computerized annotation of pictures. *IEEE Trans. Pattern Anal. Mach. Intell.*, *30*(6), 985–1002.

Li, L.-J., Socher, R., & Fei-Fei, L. (2009). Towards total scene understanding:classification, annotation and segmentation in an automatic framework. In *Proceedings of the IEEE Computer Vision and Pattern Recognition (CVPR)*.

Li, S., Kulkarni, G., Berg, T., Berg, A., & Choi, Y. (2011). Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning (CoNLL-2011)*.

Liang, P., Jordan, M. I., & Klein, D. (2009). Learning semantic correspondences with less supervision. In *Joint Conference of the 47th Annual*

Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP), Singapore.

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

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Vol. 2.

Lu, W., Ng, H. T., & Lee, W. S. (2009). Natural language generation with tree conditional random fields. In *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pp. 400–409, Morristown, NJ, USA. Association for Computational Linguistics.

Lu, W., Ng, H. T., Lee, W. S., & Zettlemoyer, L. S. (2008). A generative model for parsing natural language to meaning representations. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*, Honolulu, HI.

MacMahon, M., Stankiewicz, B., & Kuipers, B. (2006). Walk the talk: Connecting language, knowledge, and action in route instructions. In *Pro-*

ceedings of the Twenty-First National Conference on Artificial Intelligence (AAAI-06), Boston, MA.

Matuszek, C., Fox, D., & Koscher, K. (2010). Following directions using statistical machine translation. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, HRI '10, pp. 251–258, New York, NY, USA. ACM.

Motwani, T. S., & Mooney, R. J. (2012). Improving video activity recognition using object recognition and text mining. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI-2012)*, pp. 600–605.

Munteanu, D. S., & Marcu, D. (2005). Improving machine translation performance by exploiting non-parallel corpora. *Computational Linguistics*, *31*(4), 477–504.

Och, F. J., & Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, *29*(1), 19–51.

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL-2002)*, pp. 311–318, Philadelphia, PA.

Roy, D. (2002). Learning visually grounded words and syntax for a scene description task. *Computer Speech and Language*, *16*(3), 353–385.

Russell, B. C., Torralba, A., Murphy, K. P., & Freeman, W. T. (2008). Labelme: A database and web-based tool for image annotation.. Vol. 77, pp. 157–173, Hingham, MA, USA. Kluwer Academic Publishers.

Saffran, J. (2003). Statistical language learning mechanisms and constraints. *Current directions in psychological science*, *12*(4), 110–114.

Saffran, J., Johnson, E., Aslin, R., & Newport, E. (1999). Statistical learning of tone sequences by human infants and adults. *Cognition*, *70*(1), 27–52.

Shen, L., Sarkar, A., & Och, F. J. (2004). Discriminative reranking for machine translation. In Susan Dumais, D. M., & Roukos, S. (Eds.), *HLT-NAACL 2004: Main Proceedings*, pp. 177–184, Boston, Massachusetts, USA. Association for Computational Linguistics.

Shimizu, N., & Haas, A. (2009). Learning to follow navigational route instructions. In *Proceedings of the Twenty First International Joint Conference on Artificial Intelligence (IJCAI-2009)*.

Siskind, J. M. (1996). A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, *61*(1), 39–91.

Snover, M., Dorr, B., & Schwartz, R. (2008). Language and translation model adaptation using comparable corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pp. 857–866, Stroudsburg, PA, USA. Association for Computational Linguistics.

Snyder, B., & Barzilay, R. (2007). Database-text alignment via structured multilabel classification. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007)*.

Tellex, S., Kollar, T., Dickerson, S., Walter, M., Banerjee, A., Teller, S., & Roy, N. (2011). Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*.

Thompson, C. A., & Mooney, R. J. (2003). Acquiring word-meaning mappings for natural language interfaces. *Journal of Artificial Intelligence Research*, *18*, 1–44.

Toutanova, K., Haghighi, A., & Manning, C. D. (2005). Joint learning improves semantic role labeling. In *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL-05)*, pp. 589–596, Ann Arbor, MI.

Vogel, A., & Jurafsky, D. (2010). Learning to follow navigational directions. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL-10)*.

von Ahn, L., & Dabbish, L. (2004). Labeling images with a computer game. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI-04)*, Vienna, Austria.

Wang, C., Blei, D. M., & Li, F.-F. (2009). Simultaneous image classification and annotation. In *CVPR*, pp. 1903–1910.

White, M. (2006). CCG chart realization from disjunctive inputs. In *Proceedings of the Fourth International Natural Language Generation Conference*, pp. 12–19. Association for Computational Linguistics.

White, M., & Baldridge, J. (2003). Adapting chart realization to CCG. In *Proceedings of the 9th European Workshop on Natural Language Generation*, pp. 119–126.

White, M. (2004). Reining in CCG chart realization. In *Proceedings of the 3rd International Conference on Natural Language Generation (INLG-2004)*, New Forest, UK.

White, M., & Rajkumar, R. (2009). Perceptron reranking for CCG realization. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pp. 410–419, Stroudsburg, PA, USA. Association for Computational Linguistics.

Wong, Y., & Mooney, R. J. (2006). Learning for semantic parsing with statistical machine translation. In *Proceedings of Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL-06)*, pp. 439–446, New York City, NY.

Wong, Y., & Mooney, R. J. (2007a). Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of Human Language Technologies: The Conference of the North American Chapter*

of the Association for Computational Linguistics (NAACL-HLT-07), pp. 172–179, Rochester, NY.

Wong, Y., & Mooney, R. J. (2007b). Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, pp. 960–967, Prague, Czech Republic.

Yamada, K., & Knight, K. (2001). A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001)*, pp. 523–530, Toulouse, France.

Yu, C., & Ballard, D. H. (2004). On the integration of grounding language and learning objects. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pp. 488–493.

Zaragoza, H., & Li, C.-H. (2005). Learning what to talk about in descriptive games. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP-05)*, pp. 291–298, Vancouver, Canada.

Zelle, J. M., & Mooney, R. J. (1996). Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*, pp. 1050–1055, Portland, OR.

Zettlemoyer, L. S., & Collins, M. (2005). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In

Proceedings of 21st Conference on Uncertainty in Artificial Intelligence (UAI-2005), Edinburgh, Scotland.

Zettlemoyer, L. S., & Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*, pp. 678–687, Prague, Czech Republic.

Zettlemoyer, L. S., & Collins, M. (2009). Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pp. 976–984. Association for Computational Linguistics.

Vita

Joo Hyun Kim was born in 1982 in Seoul, the capital city of South Korea. After finishing high school at Hansung Science High School in 2000, he went to study Computer Science at the Korea Advanced Institute of Science and Technology (KAIST), where he received his Bachelor of Science degree in 2007. From 2003 to 2005, he worked at Medialand, Inc. in Seoul in order to fulfill his military duty. He obtained his Master of Science degree in Computer Science at the University of Texas at Austin in 2010 after he moved to U.S with the support of Samsung Scholarship. He then continued his doctoral study at the Department of Computer Science, the University of Texas at Austin, where he has been working in natural language processing, semantic parsing, and machine learning. After graduation, he will be joining eBay Applied Research in San Jose starting in September, 2013.

Permanent address: scimitar82@gmail.com

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.