Learning a Compositional Semantic Parser using an Existing Syntactic Parser

Ruifang Ge Raymond J. Mooney
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712
{grf,mooney}@cs.utexas.edu

Abstract

We present a new approach to learning a semantic parser (a system that maps natural language sentences into logical form). Unlike previous methods, it exploits an existing syntactic parser to produce disambiguated parse trees that drive the compositional semantic interpretation. The resulting system produces improved results on standard corpora on natural language interfaces for database querying and simulated robot control.

1 Introduction

Semantic parsing is the task of mapping a natural language (NL) sentence into a completely formal *meaning representation* (MR) or logical form. A *meaning representation language* (MRL) is a formal unambiguous language that supports automated inference, such as first-order predicate logic. This distinguishes it from related tasks such as *semantic role labeling* (SRL) (Carreras and Marquez, 2004) and other forms of "shallow" semantic analysis that do not produce completely formal representations. A number of systems for automatically learning semantic parsers have been proposed (Ge and Mooney, 2005; Zettlemoyer and Collins, 2005; Wong and Mooney, 2007; Lu et al., 2008). Given a training corpus of NL sentences annotated with their correct MRs, these systems induce an interpreter for mapping novel sentences into the given MRL.

Previous methods for learning semantic parsers do not utilize an existing syntactic parser that provides disambiguated parse trees.[1] However, accurate syntactic parsers are available for many languages and could potentially be used to learn more effective semantic analyzers. This paper presents an approach to learning semantic parsers that uses parse trees from an existing syntactic analyzer to drive the interpretation process. The learned parser uses standard compositional semantics to construct alternative MRs for a sentence based on its syntax tree, and then chooses the best MR based on a trained statistical disambiguation model. The learning system first employs a word alignment method from statistical machine translation (GIZA++ (Och and Ney, 2003)) to acquire a semantic lexicon that maps words to logical predicates. Then it induces rules for composing MRs and estimates the parameters of a maximum-entropy model for disambiguating semantic interpretations. After describing the details of our approach, we present experimental results on standard corpora demonstrating improved results on learning NL interfaces for database querying and simulated robot control.

(a) *If our player 2 has the ball, then position our player 5 in the midfield.*
```
((bowner (player our {2}))
(do (player our {5}) (pos (midfield))))
```
(b) *Which river is the longest?*
```
answer(x1,longest(x1,river(x1)))
```

Figure 1: Sample NLs and their MRs in the ROBOCUP and GEOQUERY domains respectively.

2 Background

In this paper, we consider two domains. The first is ROBOCUP (www.robocup.org). In the ROBOCUP Coach Competition, soccer agents compete on a simulated soccer field and receive coaching instructions in a formal language called CLANG (Chen et al., 2003). Figure 1(a) shows a sample instruction. The second domain is GEOQUERY, where a logical query language based on Prolog is used to query a database on U.S. geography (Zelle and Mooney, 1996). The logical lan-

[1] Ge and Mooney (2005) use training examples with semantically annotated parse trees, and Zettlemoyer and Collins (2005) learn a probabilistic semantic parsing model which initially requires a hand-built, ambiguous CCG grammar template.

Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 611–619,
Suntec, Singapore, 2-7 August 2009.

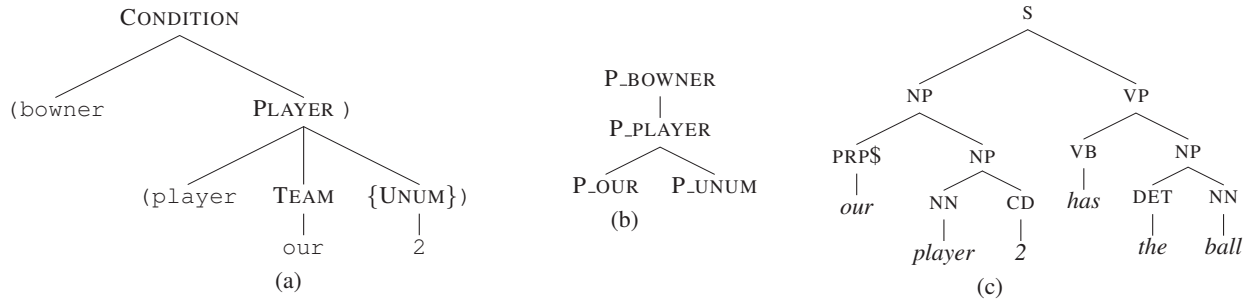


Figure 2: Parses for the condition part of the CLANG in Figure 1(a): (a) The parse of the MR. (b) The predicate argument structure of (a). (c) The parse of the NL.

PRODUCTION	PREDICATE
RULE→(CONDITION DIRECTIVE)	P_RULE
CONDITION→(bowner PLAYER)	P_BOWNER
PLAYER→(player TEAM {UNUM})	P_PLAYER
TEAM→our	P_OUR
UNUM→2	P_UNUM
DIRECTIVE→(do PLAYER ACTION)	P_DO
ACTION→(pos REGION)	P_POS
REGION→(midfield)	P_MIDFIELD

Table 1: Sample production rules for parsing the CLANG example in Figure 1(a) and their corresponding predicates.

guage consists of both first-order and higher-order predicates. Figure 1(b) shows a sample query in this domain.

We assume that an MRL is defined by an unambiguous context-free grammar (MRLG), so that MRs can be uniquely parsed, a standard requirement for computer languages. In an MRLG, each production rule introduces a single *predicate* in the MRL, where the type of the predicate is given in the left hand side (LHS), and the number and types of its arguments are defined by the nonterminals in the right hand side (RHS). Therefore, the parse of an MR also gives its predicate-argument structure.

Figure 2(a) shows the parse of the condition part of the MR in Figure 1(a) using the MRLG described in (Wong, 2007), and its predicate-argument structure is in Figure 2(b). Sample MRLG productions and their predicates for parsing this example are shown in Table 1, where the predicate P_PLAYER takes two arguments (a_1 and a_2) of type TEAM and UNUM (uniform number).

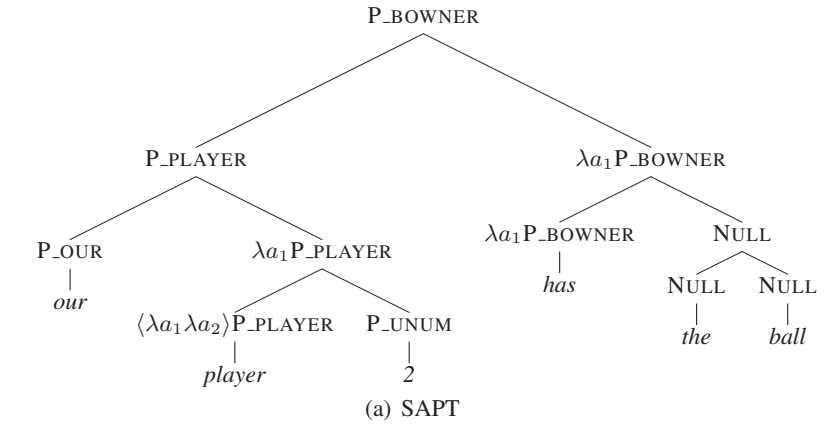
3 Semantic Parsing Framework

This section describes our basic framework, which is based on a fairly standard approach to computational semantics (Blackburn and Bos, 2005). The framework is composed of three components: 1) an existing syntactic parser to produce parse trees for NL sentences; 2) learned semantic knowledge (cf. Sec. 5), including a *semantic lexicon* to assign possible predicates (meanings) to words, and a set of *semantic composition rules* to construct possible MRs for each internal node in a syntactic parse given its children's MRs; and 3) a statistical disambiguation model (cf. Sec. 6) to choose among multiple possible semantic constructs as defined by the semantic knowledge.

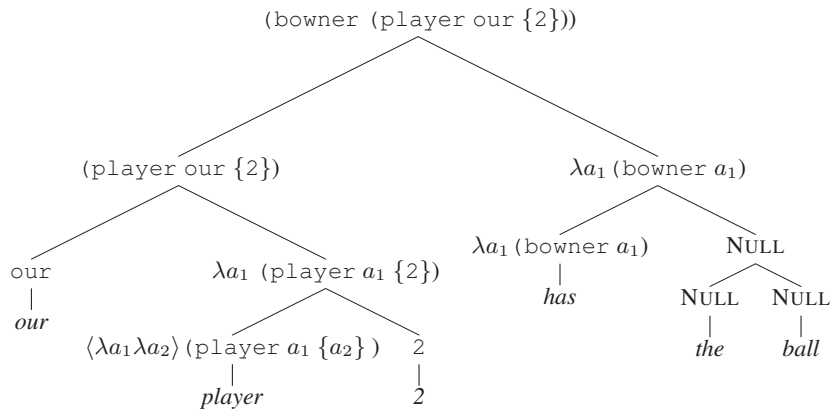
The process of generating the semantic parse for an NL sentence is as follows. First, the syntactic parser produces a parse tree for the NL sentence. Second, the semantic lexicon assigns possible predicates to each word in the sentence. Third, all possible MRs for the sentence are constructed compositionally in a recursive, bottom-up fashion following its syntactic parse using composition rules. Lastly, the statistical disambiguation model scores each possible MR and returns the one with the highest score. Fig. 3(a) shows one possible *semantically-augmented parse tree* (SAPT) (Ge and Mooney, 2005) for the condition part of the example in Fig. 1(a) given its syntactic parse in Fig. 2(c). A SAPT adds a semantic label to each non-leaf node in the syntactic parse tree. The label specifies the MRL predicate for the node and its remaining (unfilled) arguments. The compositional process assumes a binary parse tree suitable for predicate-argument composition; parses in Penn-treebank style are binarized using Collins' (1999) method.

Consider the construction of the SAPT in Fig. 3(a). First, each word is assigned a semantic label. Most words are assigned an MRL predicate. For example, the word *player* is assigned the predicate P_PLAYER with its two unbound arguments, a_1 and a_2, indicated using λ. Words that do not introduce a predicate are given the label NULL, like *the* and *ball*.[2] Next, a semantic label is as-

[2]The words *the* and *ball* are not truly "meaningless" since the predicate P_BOWNER (ball owner) is conveyed by the



(a) SAPT



(b) Semantic Derivation

Figure 3: Semantic parse for the condition part of the example in Fig. 1(a) using the syntactic parse in Fig. 2(c): (a) A SAPT with syntactic labels omitted for brevity. (b) The semantic derivation of the MR.

signed to each internal node using learned composition rules that specify how arguments are filled when composing two MRs (cf. Sec. 5). The label λa_1P_PLAYER indicates that the remaining argument a_2 of the P_PLAYER child is filled by the MR of the other child (labeled P_UNUM).

Finally, the SAPT is used to guide the composition of the sentence's MR. At each internal node, an MR for the node is built from the MRs of its children by filling an argument of a predicate, as illustrated in the *semantic derivation* shown in Fig. 3(b). Semantic composition rules (cf. Sec. 5) are used to specify the argument to be filled. For the node spanning *player 2*, the predicate P_PLAYER and its second argument P_UNUM are composed to form the MR: λa_1 (player a_1 {2}). Composing an MR with NULL leaves the MR unchanged. An MR is said to be *complete* when it contains no remaining λ variables. This process continues up the tree until a complete MR for the entire sentence is constructed at the root.

4 Ensuring Meaning Composition

The basic compositional method in Sec. 3 only works if the syntactic parse tree strictly follows the predicate-argument structure of the MR, since meaning composition at each node is assumed to combine a predicate with one of its arguments. However, this assumption is not always satisfied, for example, in the case of verb gapping and flexible word order. We use constructing the MR for the directive part of the example in Fig. 1(a) according to the syntactic parse in Fig. 4(b) as an example. Given the appropriate possible predicate attached to each word in Fig. 5(a), the node spanning *position our player 5* has children, P_POS and P_PLAYER, that are not in a predicate-argument relation in the MR (see Fig. 4(a)).

To ensure meaning composition in this case, we automatically create *macro-predicates* that combine multiple predicates into one, so that the children's MRs can be composed as argu-

phrase *has the ball*. For simplicity, predicates are introduced by a single word, but statistical disambiguation (cf. Sec. 6) uses surrounding words to choose a meaning for a word whose lexicon entry contains multiple possible predicates.

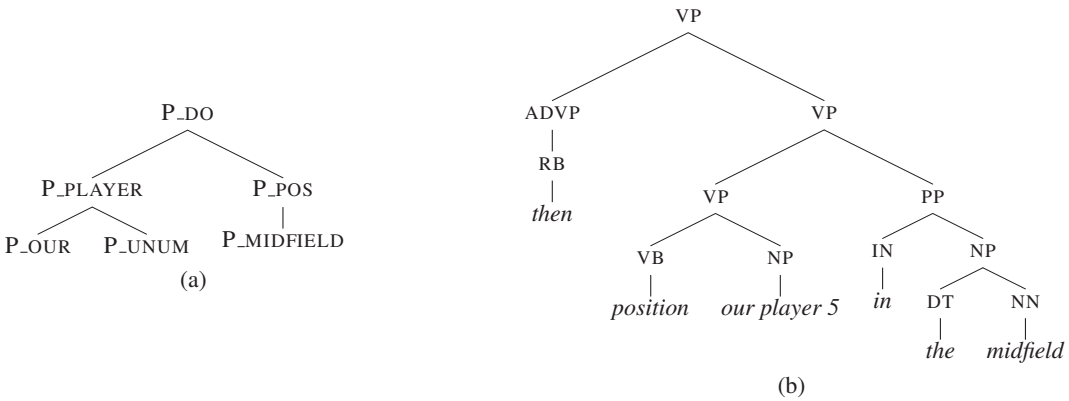


Figure 4: Parses for the directive part of the CLANG in Fig. 1(a): (a) The predicate-argument structure of the MR. (b) The parse of the NL (the parse of the phrase *our player 5* is omitted for brevity).

ments to a macro-predicate. Fig. 5(b) shows the macro-predicate P_DO_POS (DIRECTIVE→(do PLAYER (pos REGION))) formed by merging the P_DO and P_POS in Fig. 4(a). The macro-predicate has two arguments, one of type PLAYER (a_1) and one of type REGION (a_2). Now, P_POS and P_PLAYER can be composed as arguments to this macro-predicate as shown in Fig. 5(c). However, it requires assuming a P_DO predicate that has not been formally introduced. To indicate this, a lambda variable, p_1, is introduced that ranges over predicates and is provisionally bound to P_DO, as indicated in Fig. 5(c) using the notation p_1:do. Eventually, this predicate variable must be bound to a matching predicate introduced from the lexicon. In the example, p_1:do is eventually bound to the P_DO predicate introduced by the word *then* to form a complete MR.

Macro-predicates are introduced as needed during training in order to ensure that each MR in the training set can be composed using the syntactic parse of its corresponding NL given reasonable assignments of predicates to words. For each SAPT node that does not combine a predicate with a legal argument, a macro-predicate is formed by merging all predicates on the paths from the child predicates to their lowest common ancestor (LCA) in the MR parse. Specifically, a child MR becomes an argument of the macro-predicate if it is complete (i.e. contains no λ variables); otherwise, it also becomes part of the macro-predicate and its λ variables become additional arguments of the macro-predicate. For the node spanning *position our player 5* in the example, the LCA of the children P_PLAYER and P_POS is their immediate parent P_DO, therefore P_DO is included in the macro-predicate. The complete child P_PLAYER becomes the first argument of the macro-predicate. The incomplete child P_POS is added to the macro-predicate P_DO_POS and its λ variable becomes another argument.

For improved generalization, once a predicate in a macro-predicate becomes complete, it is removed from the corresponding macro-predicate label in the SAPT. For the node spanning *position our player 5 in the midfield* in Fig. 5(a), P_DO_POS becomes P_DO once the arguments of pos are filled.

In the following two sections, we describe the two subtasks of inducing semantic knowledge and a disambiguation model for this enhanced compositional framework. Both subtasks require a training set of NLs paired with their MRs. Each NL sentence also requires a syntactic parse generated using Bikel's (2004) implementation of Collins parsing model 2. Note that unlike SCISSOR (Ge and Mooney, 2005), training our method does not require gold-standard SAPTs.

5 Learning Semantic Knowledge

Learning semantic knowledge starts from learning the mapping from words to predicates. We use an approach based on Wong and Mooney (2006), which constructs word alignments between NL sentences and their MRs. Normally, word alignment is used in statistical machine translation to match words in one NL to words in another; here it is used to align words with predicates based on a "parallel corpus" of NL sentences and MRs. We assume that each word alignment defines a possible mapping from words to predicates for building a SAPT and semantic derivation which compose the correct MR. A semantic lexicon and composition rules are then extracted directly from the

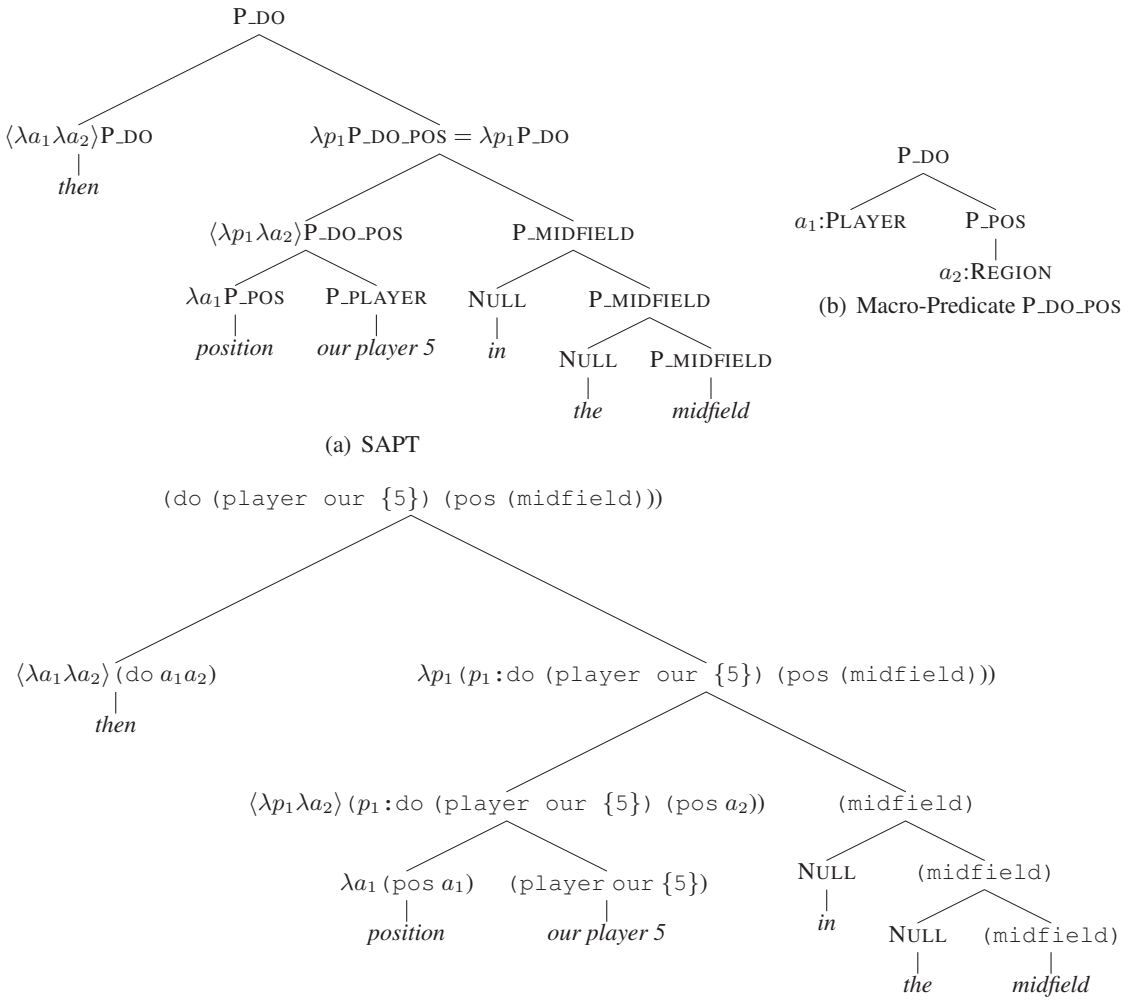


Figure 5: Semantic parse for the directive part of the example in Fig. 1(a) using the syntactic parse in Fig. 4(b): (a) A SAPT with syntactic labels omitted for brevity. (b) The predicate-argument structure of macro-predicate P_DO_POS (c) The semantic derivation of the MR.

nodes of the resulting semantic derivations.

Generation of word alignments for each training example proceeds as follows. First, each MR in the training corpus is parsed using the MRLG. Next, each resulting parse tree is linearized to produce a sequence of predicates by using a top-down, left-to-right traversal of the parse tree. Then the GIZA++ implementation (Och and Ney, 2003) of IBM Model 5 is used to generate the five best word/predicate alignments from the corpus of NL sentences each paired with the predicate sequence for its MR.

After predicates are assigned to words using word alignment, for each alignment of a training example and its syntactic parse, a SAPT is generated for composing the correct MR using the processes discussed in Sections 3 and 4. Specifically, a semantic label is assigned to each internal node of each SAPT, so that the MRs of its children are composed correctly according to the MR for this example.

There are two cases that require special handling. First, when a predicate is not aligned to any word, the predicate must be inferred from context. For example, in CLANG, *our player* is frequently just referred to as *player* and the `our` must be inferred. When building a SAPT for such an alignment, the assumed predicates and arguments are simply bound to their values in the MR. Second, when a predicate is aligned to several words, i.e. it is represented by a phrase, the alignment is transformed into several alignments where each predicate is aligned to each single word in order to fit the assumptions of compositional semantics.

Given the SAPTs constructed from the results of word-alignment, a semantic derivation for each training sentence is constructed using the methods described in Sections 3 and 4. Composition rules

are then extracted from these derivations.

Formally, composition rules are of the form:

$$\Lambda_1.P_1 + \Lambda_2.P_2 \Rightarrow \{\Lambda_p.P_p, R\} \quad (1)$$

where P_1, P_2 and P_p are predicates for the left child, right child, and parent node, respectively. Each predicate includes a lambda term Λ of the form $\langle \lambda p_{i_1}, \ldots, \lambda p_{i_m}, \lambda a_{j_1}, \ldots, \lambda a_{j_n} \rangle$, an unordered set of all unbound predicate and argument variables for the predicate. The component R specifies how some arguments of the parent predicate are filled when composing the MR for the parent node. It is of the form: $\{a_{k_1}{=}R_1, \ldots, a_{k_l}{=}R_l\}$, where R_i can be either a child (c_i), or a child's complete argument (c_i, a_j) if the child itself is not complete.

For instance, the rule extracted for the node for *player 2* in Fig. 3(b) is:

$\langle \lambda a_1 \lambda a_2 \rangle$.P_PLAYER + P_UNUM $\Rightarrow \{\lambda a_1$.P_PLAYER, $a_2{=}c_2\}$,

and for *position our player 5* in Fig. 5(c):

λa_1.P_POS + P_PLAYER $\Rightarrow \{\langle \lambda p_1 \lambda a_2 \rangle$.P_DO_POS, $a_1{=}c_2\}$,

and for *position our player 5 in the midfield*:

$\langle \lambda p_1 \lambda a_2 \rangle$.P_DO_POS + P_MIDFIELD
$\Rightarrow \{\lambda p_1$.P_DO_POS, $\{a_1{=}(c_1, a_1), a_2{=}c_2\}\}$.

The learned semantic knowledge is necessary for handling ambiguity, such as that involving word senses and semantic roles. It is also used to ensure that each MR is a legal string in the MRL.

6 Learning a Disambiguation Model

Usually, multiple possible semantic derivations for an NL sentence are warranted by the acquired semantic knowledge, thus disambiguation is needed. To learn a disambiguation model, the learned semantic knowledge (see Section 5) is applied to each training example to generate all possible semantic derivations for an NL sentence given its syntactic parse. Here, unique word alignments are not required, and alternative interpretations compete for the best semantic parse.

We use a maximum-entropy model similar to that of Zettlemoyer and Collins (2005) and Wong and Mooney (2006). The model defines a conditional probability distribution over semantic derivations (D) given an NL sentence S and its syntactic parse T:

$$\Pr(D|S, T; \bar{\theta}) = \frac{\exp \sum_i \theta_i f_i(D)}{Z_{\bar{\theta}}(S, T)} \quad (2)$$

where $\bar{f}$ $(f_1, \ldots, f_n)$ is a feature vector parameterized by $\bar{\theta}$, and $Z_{\bar{\theta}}(S, T)$ is a normalizing factor. Three simple types of features are used in the model. First, are lexical features which count the number of times a word is assigned a particular predicate. Second, are bilexical features which count the number of times a word is assigned a particular predicate *and* a particular word precedes or follows it. Last, are rule features which count the number of times a particular composition rule is applied in the derivation.

The training process finds a parameter $\bar{\theta}^*$ that (approximately) maximizes the sum of the conditional log-likelihood of the MRs in the training set. Since no specific semantic derivation for an MR is provided in the training data, the conditional log-likelihood of an MR is calculated as the sum of the conditional probability of all semantic derivations that lead to the MR. Formally, given a set of NL-MR pairs $\{(S_1, M_1), (S_2, M_2), ..., (S_n, M_n)\}$ and the syntactic parses of the NLs $\{T_1, T_2, ..., T_n\}$, the parameter $\bar{\theta}^*$ is calculated as:

$$\begin{aligned} \bar{\theta}^* &= \arg\max_{\bar{\theta}} \sum_{i=1}^{n} \log \Pr(M_i|S_i, T_i; \bar{\theta}) \quad (3) \\ &= \arg\max_{\bar{\theta}} \sum_{i=1}^{n} \log \sum_{D_i^*} \Pr(D_i^*|S_i, T_i; \bar{\theta}) \end{aligned}$$

where D_i^* is a semantic derivation that produces the correct MR M_i.

L-BFGS (Nocedal, 1980) is used to estimate the parameters $\bar{\theta}^*$. The estimation requires statistics that depend on all possible semantic derivations and all correct semantic derivations of an example, which are not feasibly enumerated. A variant of the Inside-Outside algorithm (Miyao and Tsujii, 2002) is used to efficiently collect the necessary statistics. Following Wong and Mooney (2006), only candidate predicates and composition rules that are used in the best semantic derivations for the training set are retained for testing. No smoothing is used to regularize the model; We tried using a Gaussian prior (Chen and Rosenfeld, 1999), but it did not improve the results.

7 Experimental Evaluation

We evaluated our approach on two standard corpora in CLANG and GEOQUERY. For CLANG, 300 instructions were randomly selected from the log files of the 2003 ROBOCUP Coach

Competition and manually translated into English (Kuhlmann et al., 2004). For GEOQUERY, 880 English questions were gathered from various sources and manually translated into Prolog queries (Tang and Mooney, 2001). The average sentence lengths for the CLANG and GEOQUERY corpora are 22.52 and 7.48, respectively.

Our experiments used 10-fold cross validation and proceeded as follows. First Bikel's implementation of Collins parsing model 2 was trained to generate syntactic parses. Second, a semantic parser was learned from the training set augmented with their syntactic parses. Finally, the learned semantic parser was used to generate the MRs for the test sentences using their syntactic parses. If a test example contains constructs that did not occur in training, the parser may fail to return an MR.

We measured the performance of semantic parsing using *precision* (percentage of returned MRs that were correct), *recall* (percentage of test examples with correct MRs returned), and *F-measure* (harmonic mean of precision and recall). For CLANG, an MR was correct if it exactly matched the correct MR, up to reordering of arguments of commutative predicates like *and*. For GEOQUERY, an MR was correct if it retrieved the same answer as the gold-standard query, thereby reflecting the quality of the final result returned to the user.

The performance of a syntactic parser trained only on the Wall Street Journal (WSJ) can degrade dramatically in new domains due to corpus variation (Gildea, 2001). Experiments on CLANG and GEOQUERY showed that the performance can be greatly improved by adding a small number of treebanked examples from the corresponding training set together with the WSJ corpus. Our semantic parser was evaluated using three kinds of syntactic parses. Listed together with their PARSEVAL F-measures these are: gold-standard parses from the treebank (GoldSyn, 100%), a parser trained on WSJ plus a small number of in-domain training sentences required to achieve good performance, 20 for CLANG (Syn20, 88.21%) and 40 for GEOQUERY (Syn40, 91.46%), and a parser trained on no in-domain data (Syn0, 82.15% for CLANG and 76.44% for GEOQUERY).

We compared our approach to the following alternatives (where results for the given corpus were available): SCISSOR (Ge and Mooney, 2005), an integrated syntactic-semantic parser; KRISP (Kate and Mooney, 2006), an SVM-based parser using string kernels; WASP (Wong and Mooney, 2006; Wong and Mooney, 2007), a system based on synchronous grammars; Z&C (Zettlemoyer and Collins, 2007)[3], a probabilistic parser based on relaxed CCG grammars; and LU (Lu et al., 2008), a generative model with discriminative reranking. Note that some of these approaches require additional human supervision, knowledge, or engineered features that are unavailable to the other systems; namely, SCISSOR requires gold-standard SAPTs, Z&C requires hand-built template grammar rules, LU requires a reranking model using specially designed global features, and our approach requires an existing syntactic parser. The F-measures for syntactic parses that generate correct MRs in CLANG are 85.50% for syn0 and 91.16% for syn20, showing that our method can produce correct MRs even when given imperfect syntactic parses. The results of semantic parsers are shown in Tables 2 and 3.

	Precision	Recall	F-measure
GOLDSYN	84.73	74.00	79.00
SYN20	85.37	70.00	76.92
SYN0	87.01	67.00	75.71
WASP	88.85	61.93	72.99
KRISP	85.20	61.85	71.67
SCISSOR	89.50	73.70	80.80
LU	82.50	67.70	74.40

Table 2: Performance on CLANG.

	Precision	Recall	F-measure
GOLDSYN	91.94	88.18	90.02
SYN40	90.21	86.93	88.54
SYN0	81.76	78.98	80.35
WASP	91.95	86.59	89.19
Z&C	91.63	86.07	88.76
SCISSOR	95.50	77.20	85.38
KRISP	93.34	71.70	81.10
LU	89.30	81.50	85.20

Table 3: Performance on GEOQUERY.

First, not surprisingly, more accurate syntactic parsers (i.e. ones trained on more in-domain data) improved our approach. Second, in CLANG, all of our methods outperform WASP and KRISP, which also require no additional information during training. In GEOQUERY, Syn0 has significantly worse results than WASP and our other systems using better syntactic parses. This is not surprising since Syn0's F-measure for syntactic parsing is only 76.44% in GEOQUERY due to a lack

[3]These results used a different experimental setup, training on 600 examples, and testing on 280 examples.

	Precision	Recall	F-measure
GOLDSYN	61.14	35.67	45.05
SYN20	57.76	31.00	40.35
SYN0	53.54	22.67	31.85
WASP	88.00	14.37	24.71
KRISP	68.35	20.00	30.95
SCISSOR	85.00	23.00	36.20

Table 4: Performance on CLANG40.

	Precision	Recall	F-measure
GOLDSYN	95.73	89.60	92.56
SYN20	93.19	87.60	90.31
SYN0	91.81	85.20	88.38
WASP	91.76	75.60	82.90
SCISSOR	98.50	74.40	84.77
KRISP	84.43	71.60	77.49
LU	91.46	72.80	81.07

Table 5: Performance on GEO250 (20 in-domain sentences are used in SYN20 to train the syntactic parser).

of interrogative sentences (questions) in the WSJ corpus. Note the results for SCISSOR, KRISP and LU on GEOQUERY are based on a different meaning representation language, FUNQL, which has been shown to produce lower results (Wong and Mooney, 2007). Third, SCISSOR performs better than our methods on CLANG, but it requires extra human supervision that is not available to the other systems. Lastly, a detailed analysis showed that our improved performance on CLANG compared to WASP and KRISP is mainly for long sentences (> 20 words), while performance on shorter sentences is similar. This is consistent with their relative performance on GEOQUERY, where sentences are normally short. Longer sentences typically have more complex syntax, and the traditional syntactic analysis used by our approach results in better compositional semantic analysis in this situation.

We also ran experiments with less training data. For CLANG, 40 random examples from the training sets (CLANG40) were used. For GEOQUERY, an existing 250-example subset (GEO250) (Zelle and Mooney, 1996) was used. The results are shown in Tables 4 and 5. Note the performance of our systems on GEO250 is higher than that on GEOQUERY since GEOQUERY includes more complex queries (Tang and Mooney, 2001). First, all of our systems gave the best F-measures (except SYN0 compared to SCISSOR in CLANG40), and the differences are generally quite substantial. This shows that our approach significantly improves results when limited training data is available. Second, in CLANG, reducing the training data increased the difference between SYN20 and SYN0. This suggests that the quality of syntactic parsing becomes more important when less training data is available. This demonstrates the advantage of utilizing existing syntactic parsers that are learned from large open domain treebanks instead of relying just on the training data.

We also evaluated the impact of the word alignment component by replacing Giza++ by gold-standard word alignments manually annotated for the CLANG corpus. The results consistently showed that compared to using gold-standard word alignment, Giza++ produced lower semantic parsing accuracy when given very little training data, but similar or better results when given sufficient training data (> 160 examples). This suggests that, given sufficient data, Giza++ can produce effective word alignments, and that imperfect word alignments do not seriously impair our semantic parsers since the disambiguation model evaluates multiple possible interpretations of ambiguous words. Using multiple potential alignments from Giza++ sometimes performs even better than using a single gold-standard word alignment because it allows multiple interpretations to be evaluated by the global disambiguation model.

8 Conclusion and Future work

We have presented a new approach to learning a semantic parser that utilizes an existing syntactic parser to drive compositional semantic interpretation. By exploiting an existing syntactic parser trained on a large treebank, our approach produces improved results on standard corpora, particularly when training data is limited or sentences are long. The approach also exploits methods from statistical MT (word alignment) and therefore integrates techniques from statistical syntactic parsing, MT, and compositional semantics to produce an effective semantic parser.

Currently, our results comparing performance on long versus short sentences indicates that our approach is particularly beneficial for syntactically complex sentences. Follow up experiments using a more refined measure of syntactic complexity could help confirm this hypothesis. Reranking could also potentially improve the results (Ge and Mooney, 2006; Lu et al., 2008).

Acknowledgments

This research was partially supported by NSF grant IIS–0712097.

References

Daniel M. Bikel. 2004. Intricacies of Collins' parsing model. *Computational Linguistics*, 30(4):479–511.

Patrick Blackburn and Johan Bos. 2005. *Representation and Inference for Natural Language: A First Course in Computational Semantics*. CSLI Publications, Stanford, CA.

Xavier Carreras and Luis Marquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proc. of 8th Conf. on Computational Natural Language Learning (CoNLL-2004)*, Boston, MA.

Stanley F. Chen and Ronald Rosenfeld. 1999. A Gaussian prior for smoothing maximum entropy model. Technical Report CMU-CS-99-108, School of Computer Science, Carnegie Mellon University.

Mao Chen, Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Patrick Riley, Timo Steffens, Yi Wang, and Xiang Yin. 2003. Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at `http://sourceforge.net/projects/sserver/`.

Michael Collins. 1999. *Head-driven Statistical Models for Natural Language Parsing*. Ph.D. thesis, University of Pennsylvania.

Ruifang Ge and Raymond J. Mooney. 2005. A statistical semantic parser that integrates syntax and semantics. In *Proc. of 9th Conf. on Computational Natural Language Learning (CoNLL-2005)*, pages 9–16.

Ruifang Ge and Raymond J. Mooney. 2006. Discriminative reranking for semantic parsing. In *Proc. of the 21st Intl. Conf. on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, Sydney, Australia, July.

Daniel Gildea. 2001. Corpus variation and parser performance. In *Proc. of the 2001 Conf. on Empirical Methods in Natural Language Processing (EMNLP-01)*, Pittsburgh, PA, June.

Rohit J. Kate and Raymond J. Mooney. 2006. Using string-kernels for learning semantic parsers. In *Proc. of the 21st Intl. Conf. on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (COLING/ACL-06)*, pages 913–920, Sydney, Australia, July.

Greg Kuhlmann, Peter Stone, Raymond J. Mooney, and Jude W. Shavlik. 2004. Guiding a reinforcement learner with natural language advice: Initial results in RoboCup soccer. In *Proc. of the AAAI-04 Workshop on Supervisory Control of Learning and Adaptive Systems*, San Jose, CA, July.

Wei Lu, Hwee Tou Ng, Wee Sun Lee, and Luke S. Zettlemoyer. 2008. A generative model for parsing natural language to meaning representations. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing (EMNLP-08)*, Honolulu, Hawaii, October.

Yusuke Miyao and Jun'ichi Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proc. of Human Language Technology Conf.(HLT-2002)*, San Diego, CA, March.

Jorge Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, July.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Lappoon R. Tang and Raymond J. Mooney. 2001. Using multiple clause constructors in inductive logic programming for semantic parsing. In *Proc. of the 12th European Conf. on Machine Learning*, pages 466–477, Freiburg, Germany.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proc. of Human Language Technology Conf. / N. American Chapter of the Association for Computational Linguistics Annual Meeting (HLT-NAACL-2006)*, pages 439–446.

Yuk Wah Wong and Raymond J. Mooney. 2007. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proc. of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, pages 960–967.

Yuk Wah Wong. 2007. *Learning for Semantic Parsing and Natural Language Generation Using Statistical Machine Translation Techniques*. Ph.D. thesis, Department of Computer Sciences, University of Texas, Austin, TX, August. Also appears as Artificial Intelligence Laboratory Technical Report AI07-343.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proc. of 13th Natl. Conf. on Artificial Intelligence (AAAI-96)*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proc. of the 21th Annual Conf. on Uncertainty in Artificial Intelligence (UAI-05)*.

Luke S. Zettlemoyer and Michael Collins. 2007. Online learning of relaxed CCG grammars for parsing to logical form. In *Proc. of the 2007 Joint Conf. on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*, pages 678–687, Prague, Czech Republic, June.