# Automatic Grammar Induction from Semantic Parsing

*Debajit Ghosh*[*] *and David Goddeau*

Compaq Cambridge Research Laboratory
One Kendall Square, Building 700, 2nd Floor
Cambridge, MA 02139, USA
e-mail: debajit@mit.edu, dg@crl.dec.com

## ABSTRACT

This research investigates using semantic information to learn syntax rules automatically. After describing a semantic parsing mechanism for parsing utterances based on meaning, we illustrate a grammar induction technique which uses semantic parsing's results to create syntactic rules. We also present and discuss several experiments which use the learned grammar in syntactic parsing experiments in two domains. Overall, the learned grammar covers 98% of semantically-valid utterances in its original domain and 85% in a different domain.

## 1. INTRODUCTION

One approach to spoken language understanding converts a transcribed utterance into a semantic meaning representation, which is then interpreted to produce a response. This can be accomplished with conventional parsing technology given a syntactic grammar and semantic composition rules. However, constructing such a grammar, even within a given domain, is a difficult and time-consuming task. An alternative approach is to learn the translation rules (grammar and construction functions) from a corpus of translated examples. This eliminates the need for knowledge engineering but requires the collection and annotation of the corpus, which can be as difficult and expensive.

This paper describes work toward automatically learning syntactic grammar rules from an un-annotated corpus, using a minimal amount of knowledge engineering. Inspired by the observation that people can understand agrammatical constructs (and perhaps acquire grammatical knowledge) based on semantic and real-world constraints on what "makes sense," our approach uses a "semantic parser" to seed the grammar induction process. After describing the exact mechanism we use for semantic parsing, we illustrate a grammar induction technique which makes use of the output of semantic parsing to generate the desired grammatical rules. We present several experiments making use of the learned grammar, measuring how well this grammar covers utterances from the domain in which it was learned as well as a different domain, assessing the portability of this grammar. We conclude with a discussion of the results of these experiments as well as ideas for future work.

## 2. SEMANTIC PARSING

Semantic parsing involves using semantics directly in parsing and understanding input utterances. There has been some work done in this field, mostly involving using hidden Markov models (HMMs) [6] and artificial neural networks (ANNs) [4, 3], as well as simply using semantics to *complement* rather than replace syntax [1]. However, none of the techniques we researched seem to provide information which would be useful for driving grammar induction techniques, so we developed our own.

### 2.1. Mechanism

Our semantic parser is based on a bottom-up chart parsing system. The original system utilizes a grammar, semantic functions, and semantic constraints. Using the grammar, the system computes a syntactic parse tree for an utterance. It then generates the utterance's semantic frame representation, consisting of a head, denoting the main concept, and possibly a set of key-value pair modifiers which further describe this concept. The system generates this representation for an utterance by looking up or combining the semantics of each syntactic node's constituents, working up to the root node.

Semantic functions dictate how semantics combine; each takes its arguments in a specified order (i.e., $f$ (0 1) specifies that the first constituent represents the first argument; $f$ (1 0) specifies that the second constituent is the first argument) and combines them into output semantics. A set of semantic constraints, describing which words can take which predicates and arguments, restrict the semantic functions to combining only those words and phrases which "make sense" to combine.

Semantic parsing builds on this concept further. This technique can be summarized as parsing utterances based on meaning, only combining words and phrases which are "meaningful" to combine. Instead of relying on a grammar to dictate which semantic functions should be applied, semantic parsing eliminates the grammar altogether and tries to use *all* functions, resulting in whatever combinations are allowed by the semantic constraints.

Thus, our semantic parsing system works by using semantic-based edges, instead of syntax-based ones. The parser uses the semantics of input words (defined in a lexicon) as the semantics of its edge. Instead of attempting to combine edges based on syntax, the system tries to apply the semantics of two edges as

[*] you may reach this author at: Nuance Communications, 1380 Willow Road, Menlo Park, CA 94025, USA

arguments to each semantic function. This results in the creation of edges for whatever combinations are permitted (what “makes sense”) by the constraints. Although semantic parsing still requires writing these constraints, this task involves identifying basic semantic concepts and the various predicates and arguments that concept may involve, and therefore tends to be more simple than writing syntax rules. Effectively, semantic parsing requires defining and understanding *what* concepts can combine but not the syntax-level details of *how* they actually combine.

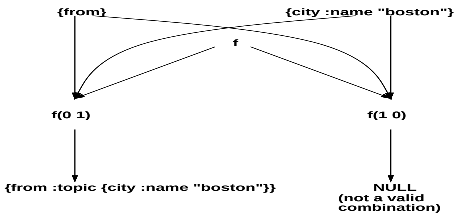

**Figure 1:** Semantic parsing ignores word order when combining two words or phrases. The system attempts to compute $f$ (0 1) as well as $f$ (1 0) for all semantic functions $f$, relying on the semantic constraints to restrict output to only meaningful combinations.

Semantic parsing attempts to avoid syntactic influence as much as possible. For example, semantic parsing should ignore the order in which words appear in an utterance. For efficiency, our system approximates word order independence by maintaining adjacency constraints. It ignores the order in which adjacent words and phrases appear, only trying to combine those edges two different ways: one with the semantics of the first edge as the first argument to a semantic function, and one with the semantics of the second edge as the first argument, as shown in Figure 1.

Also for efficiency, the system uses inheritance to simplify writing these constraints. By declaring certain semantic concepts to be “children” of other concepts, any valid arguments of the ancestors of a word can be considered valid arguments of the word itself. We also define sentence-level functions which examine the semantics of an utterance and ignore pre-defined filler words (i.e., *“please”*), possibly combine non-filler fragments, and determine if the utterance refers to a query, statement, or command.

## 2.2. Example

As this system processes input utterances, it records the computed meaning representation for these utterances. It also records the parts of speech (also defined in the lexicon) of the words that combine in semantic parsing, as well as the semantic function that allows for their combination. This information is useful in later processing by grammar induction mechanisms.

Therefore, if the system is presented with the utterance *“cheapest flight from boston to philadelphia”* (and provided the appropriate lexicon and constraints), it logs the information shown in Figure 2. The system first combines *“cheapest”* with *“flight”* (*“cheapest”* is a valid argument of *“flight”* describing its cost) by some semantic function $f_0$, combines *“from”* with *“boston”* using $f_1$, subsequently combines those two fragments together with $f_2$, and so forth.

| | |
|---|---|
| utterance: | cheapest flight from boston to philadelphia |
| meaning representation: | {flight<br>:from { city :name “boston”}<br>:to { city :name “philadelphia”}<br>:pred { cheap :type “superlative”}<br>} |
| parts of speech: | [$f_2$ (0 1):<br>[$f_2$ (0 1):<br>[$f_0$ (1 0): Adj N]<br>[$f_1$ (0 1): P Name]<br>]<br>[$f_1$ (0 1): P Name]<br>] |

**Figure 2:** Semantic parse of *“cheapest flight from boston to philadelphia.”* The parser records the utterance, the computed meaning representation, and the parts of speech of words that combine as well as the function that allows for their combination.

## 2.3. Domains and Coverage

For our semantic parsing and grammar induction development and our associated syntactic parsing experiments, we used data from two domains. Most of the testing and development utilized the ATIS domain [5]; our subsequent portability experiment utilized the Jupiter domain [8]. We split data from the ATIS II and III collections into a 3764 utterance training set (ATIS TRAIN) and a 1033 test set (ATIS TEST); we chose 1000 utterances from Jupiter (Jupiter TEST) randomly.

We also defined lexicons and constraints for each of these domains. Our goal was not to define these components completely but to define a sufficient lexicon and constraints to allow us to run several meaningful experiments. For ATIS, our lexicon and constraints contained 600 and 175 entries, respectively; for Jupiter, those components contained 829 and 126 entries, respectively.

We ran our semantic parser through each of these sets to assess the coverage (percentage of complete parses, or parses consisting of only a single, unfragmented piece) of our constraints; again, we were not seeking to achieve complete coverage but rather wanted to obtain these numbers for evaluating subsequent parsing experiments. We achieved 42% coverage of the ATIS sets and 64% coverage of the Jupiter set.

# 3. GRAMMAR INDUCTION

Grammar induction is a well researched field in which many different approaches have been investigated [2]. Several basic techniques approach the problem by extracting an overly simplistic grammar and clustering syntactic units for generalization [2, 7].

Accordingly, we originally investigated a clustering-based grammar induction approach. This approach extracted a new rule for every unique bracketing seen in the semantic parse logs and iteratively clustered these rules. After analyzing this approach, we decided that it did not make enough use of the semantic-level information available and decided to develop an approach that did;

we called the resulting technique semantic-head driven induction (SHDI), as described below.

## 3.1. Semantic-Head Driven Induction

In our implementation of semantic functions, the first argument to a function constitutes the head, or major concept, of the resulting semantic frame. This recognizes how semantic-level phrases are constructed; we can use these phrases to influence the learning of syntactic structure. By using the part of speech of the first argument to a semantic function to create syntactic phrases, SHDI can generate clean, readable rules directly from the semantic parse logs. These rules are often compact and recursive, eliminating the need for clustering; essentially, the grammar is pre-merged.

[$f_2$ (0 1):
  [$f_2$ (0 1):
    [$f_0$ (1 0): Adj N]
    [$f_1$ (0 1): P Name]
  ]
  [$f_1$ (0 1): P Name]
]

||
Semantic-Head Driven Induction
⇓

| Left-hand Side | | Right-hand Side | Sem. Function |
|---|---|---|---|
| N_0 | ⟶ | Adj N | $f_0$ (1 0) |
| P_0 | ⟶ | P Name | $f_1$ (0 1) |
| N_0 | ⟶ | N_0 P_0 | $f_2$ (0 1) |

**Figure 3:** Rules learned in semantic-head driven induction from *"cheapest flight from boston to philadelphia."* SHDI uses the part of speech of the first argument to each function to create syntactic phrases.

The example in Figure 3 illustrates this mechanism, using the parse log from Figure 2 as input. SHDI begins by recognizing the noun *"flight"* as the first argument to $f_0$ and extracts a syntactic rule **N_0** for the **Adj N** combination. It then recognizes the preposition *"from"* as the head in the combination of *"from"* and *"boston"* and creates the **P_0** rule and phrase type for this combination. Finally, this technique observes that *"cheapest flight"* is the first argument to $f_2$ and therefore creates another instance of the **N_0** (the derived part of speech for *"cheapest flight"*) rule corresponding to the **N_0 P_0** combination.

## 3.2. Results of Induction

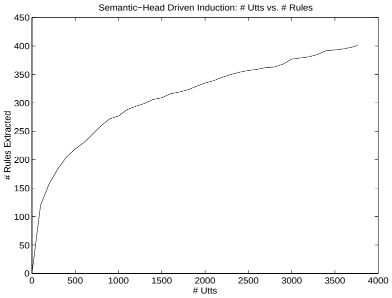

**Figure 4:** Number of utterances vs. number of rules learned in semantic-head driven induction.

For our research, we perform a semantic parse of the ATIS TRAIN set and use SHDI to induce the corresponding grammar. Figure 4 provides an interesting view of the learning process, plotting the number of utterances against the number of rules learned. As desired, one can clearly see that as the system parses more utterances, it encounters fewer previously unseen formations and does not need to create as many new rules.

| Left-hand Side | | Right-hand Side | Sem. Function | Count |
|---|---|---|---|---|
| N_0 | ⟶ | N_0 P_0 | $f_2$ (0 1) | 2400 |
| N_0 | ⟶ | N N | $f_3$ (1 0) | 150 |
| N_0 | ⟶ | Number_0 N | $f_0$ (1 0) | 12 |
| P_0 | ⟶ | Name P | $f_1$ (1 0) | 111 |
| P_0 | ⟶ | P Name | $f_1$ (0 1) | 4241 |
| Number_0 | ⟶ | Number Number | $f_5$ (0 1) | 137 |
| Number_0 | ⟶ | Number Number | $f_5$ (1 0) | 132 |
| SENT_0 | ⟶ | Aux Pro V_0 | $f_6$ (2) | 23 |
| SENT_0 | ⟶ | Please V_0 | $f_6$ (1) | 98 |
| SENT_0 | ⟶ | WhObj Cop N_0 | $f_7$ (0 2) | 223 |

**Figure 5:** Examples of rules learned in semantic-head driven induction.

Using SHDI, the system extracts 401 grammar rules; Figure 5 lists some examples of these rules. Qualitatively, many of the rules learned under SHDI are quite reasonable, such as **N_0** → **N_0 P_0**, or **N_0** → **N N**. The system even extracts sentence-level syntactic rules which handle sentences such as *"can you show me flights ..."*, *"please show me flights ..."*, and *"what are flights ..."*.

Some rules are more puzzling, such as **P_0** → **Name P** in *addition* to the expected **P_0** → **P Name**. However, SHDI records not only the rules that it learns but also counts of how many times each rule occurs in the training data. One can see that the expected **P Name** combination clearly dominates the **Name P** combination. A statistical parser could assign that rule a lower probability or ignore it and recover from the existence of this otherwise confusing pair of rules.

Some constructs do give the semantically-based induction system less avoidable problems. Specifically, it is difficult to distinguish numbers based solely on semantics, while ignoring word order. Because the components of the number *"fifty two"* are considered in both orders (fifty two, two fifty), the system cannot ascertain which number is meant; extracting useful syntactic rules for numbers is difficult without the use of syntactic cues or prelabeled examples.

# 4. EXPERIMENTS

After learning a grammar using the SHDI approach, we assess its usefulness through a series of syntactic parsing experiments using the learned grammar. These experiments involve using the grammar both in the original domain in which it was learned (ATIS) as well as a new domain (Jupiter).

To run these experiments, we addressed a concern involving the speed of the parsing system. Using an all-parses bottom-up chart parser, we found the system to be too slow to run our experiments in a reasonable amount of time. Consequently, we enhanced the system by adding semantic filtering. Instead of performing all semantic computation after completing a parse, we modified the chart parser so it computed semantics *during* parsing, immediately removing any semantically-invalid syntactic edges. This prevented the system from keeping track of the edges which could not contribute to a valid parse. Because of this filtering, we evaluated our syntactic parsing experiments relative to the semantic parsing coverage; alternatively, we restricted our experiments to semantically-valid utterances (ones which were covered by the defined semantics and constraints).

The first experiment assessed the learned grammar's performance in the original domain. After inducing a grammar from the semantic parse of ATIS TRAIN, we tried this grammar on two sets of utterances from ATIS. As an initial check, we simply ran a syntactic parsing experiment in ATIS TRAIN to ensure we got full coverage. Next, we used this grammar on a new set of utterances (ATIS TEST), achieving promising results. The induced grammar covered 98% of the utterances that were covered semantically (or 41% overall).

The final parsing experiment involved assessing how well the ATIS-trained grammar performed in a different domain altogether. We chose the Jupiter domain for this portability experiment. Jupiter, a weather information domain, primarily consisted of weather-related queries, such as *"what is the weather forecast for boston."* The grammar performed reasonably well, covering 85% of valid utterances (54% overall).

Figure 6 displays the overall coverage. As shown, there is some disparity between the results in the parsing and portability experiments. We believe this involves sentence-level rules. While Jupiter contains many *"is it"* constructs (*"is it raining"*), ATIS contains no such construct, and the ATIS-trained grammar never learns the corresponding syntactic rules and lacks coverage of these utterances. To improve the portability of our system, we need to improve the way sentence-level rules are learned.

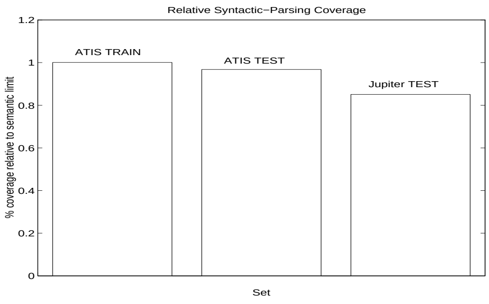

**Figure 6:** Overall parsing coverage. The system achieves 98% and 85% coverage of semantically-valid utterances from test sets in ATIS and Jupiter, respectively.

## 5. CONCLUSIONS AND FUTURE WORK

We feel that semantic parsing and semantic-head driven induction are both powerful and useful techniques with much potential. Semantic parsing proves to be an interesting mechanism for producing meaning representations for utterances directly, and semantic-head driven induction produces a readable and useful grammar with potential for portability. We would like to work on improving some of the results of this research by addressing four issues. First, we want to implement and use a statistical best-first parser to speed up the parsing experiments, allowing us to run syntactic parsing without semantic filtering. Second, we would like to seed the system with syntactic rules for semantically-unconstrained but syntactically-straightforward concepts, like number combinations. Third, we would like to improve the way sentence-level rules are learned, perhaps training the system over several domains to learn a more robust and portable set of rules. Finally, we want to investigate mirroring the results of this research toward learning semantic constraints automatically. This research begins with constraints and uses them to learn syntax, trying the learned rules in a new domain. We would like to see if one can similarly use these syntax rules in a new domain to deduce the correlation between different words and learn semantic constraints for that domain automatically. This could effectively allow a system to port itself to new domains.

## 6. ACKNOWLEDGMENTS

We would like to thank Dr. James Glass for his help and advice in conducting this research as well as the Spoken Language Systems group at MIT for providing us with some of the data used in our experiments.

## 7. REFERENCES

1. R. Bobrow, R. Ingria, and D. Stallard. Syntactic/semantic coupling in the BBN DELPHI system. In *Proceedings DARPA Speech and Natural Language Workshop*, pages 311–315, Harriman, NY, February 1992.
2. K-S Fu and T.L. Booth. Grammatical inference: Introduction and survey – Parts I and II. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-5 No. 1 and No. 4, January and July 1975.
3. A.L. Gorin, S.E. Levinson, and A.N. Gertner. Adaptive acquisition of spoken language. In *Proceedings ICASSP*, pages 805–808, Toronto, Canada, April 1991.
4. A.L. Gorin, S.E. Levinson, L.G. Miller, A.N. Gertner, A. Ljolje, and E.R. Goldman. On adaptive acquisition of language. In *Proceedings ICASSP*, pages 601–604, Albuquerque, NM, April 1990.
5. L. Hirschman et al. Multi-site data collection for a spoken language system. In *Proceedings DARPA Speech and Natural Language Workshop*, pages 7–14, Harriman, NY, February 1992.
6. E. Levin and R. Pieraccini. Concept-based spontaneous speech understanding system. In *Proceedings Eurospeech*, pages 555–558, Madrid, Spain, September 1995.
7. M. McCandless. Automatic acquisition of language models for speech recognition. Master's thesis, Massachusetts Institute of Technology, May 1994.
8. V. Zue, S. Seneff, J. Glass, L. Hetherington, E. Hurley, H. Meng, C. Pao, J. Polifroni, R. Schloming, and P. Schmid. From interface to content: Translingual access and delivery of on-line information. In *Proceedings Eurospeech*, pages 2227–2230, Rhodes, Greece, September 1997.