Appears in *Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99),* pp. 487-493, Orlando, FL, July 1999

Automatic Construction of Semantic Lexicons for Learning Natural Language Interfaces

Cynthia A. Thompson
Center for the Study of Language and Information
Stanford University
Stanford, CA 94305-4115
cthomp@csli.stanford.edu

Raymond J. Mooney
Department of Computer Sciences
University of Texas
Austin, TX 78712
mooney@cs.utexas.edu

Abstract

This paper describes a system, WOLFIE (WOrd Learning From Interpreted Examples), that acquires a semantic lexicon from a corpus of sentences paired with semantic representations. The lexicon learned consists of words paired with meaning representations. WOLFIE is part of an integrated system that learns to parse novel sentences into semantic representations, such as logical database queries. Experimental results are presented demonstrating WOLFIE's ability to learn useful lexicons for a database interface in four different natural languages. The lexicons learned by WOLFIE are compared to those acquired by a similar system developed by Siskind (1996).

Introduction & Overview

The application of learning methods to natural-language processing (NLP) has drawn increasing attention in recent years. Using machine learning to help automate the construction of NLP systems can eliminate much of the difficulty of manual construction. The semantic lexicon, or the mapping from words to meanings, is one component that is typically challenging and time consuming to construct and update by hand. This paper describes a system, WOLFIE (WOrd Learning From Interpreted Examples), that acquires a semantic lexicon of word/meaning pairs from a corpus of sentences paired with semantic representations. The goal of this research is to automate lexicon construction for an integrated NLP system that acquires both semantic lexicons and parsers for natural-language interfaces from a single training set of annotated sentences.

Although a few others (Siskind 1996; Hastings & Lytinen 1994; Brent 1991) have presented systems for learning information about lexical semantics, this work is unique in combining several features. First, interaction with a system, CHILL (Zelle & Mooney 1996), that learns to parse sentences into semantic representations is demonstrated. Second, it uses a fairly straightforward batch, greedy learning algorithm that is fast and accurate. Third, it is easily extendible to new representation formalisms. Fourth, no prior knowledge is required although it can exploit an initial lexicon if provided.

We tested WOLFIE's ability to acquire a semantic lexicon for an interface to a geographical database using a corpus of queries collected from human subjects and annotated with their logical form. In this test, WOLFIE was integrated with CHILL, which learns parsers but requires a semantic lexicon (previously built manually). The results demonstrate that the final acquired parser performs nearly as accurately at answering novel questions when using a learned lexicon as when using a hand-built lexicon. WOLFIE is also compared to an alternative lexicon acquisition system developed by Siskind (1996), demonstrating superior performance on this task. Finally, the corpus was translated into Spanish, Japanese, and Turkish and experiments conducted demonstrating an ability to learn successful lexicons and parsers for a variety of languages. Overall, the results demonstrate a robust ability to acquire accurate lexicons directly usable for semantic parsing. With such an integrated system, the task of building a semantic parser for a new domain is simplified. A single representative corpus of sentence/representation pairs allows the acquisition of both a semantic lexicon and parser that generalizes well to novel sentences.

Background

CHILL uses *inductive logic programming* (Muggleton 1992; Lavrač & Džeroski 1994) to learn a deterministic shift-reduce parser written in Prolog. The input to CHILL is a corpus of sentences paired with semantic representations, the same input required by WOLFIE. The parser learned is capable of mapping the sentences into their correct representations, as well as generalizing well to novel sentences. In this paper, we limit our discussion to acquiring parsers that map natural-language questions directly into Prolog queries that can be executed to produce an answer (Zelle & Mooney 1996). Following are two sample queries for a database on U.S. geography, paired with their corresponding Prolog query:

What is the capital of the state with the biggest population?

```
answer(C, (capital(S,C), largest(P,
```

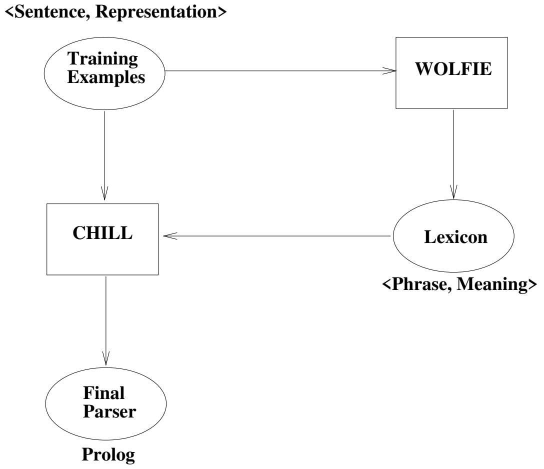

Figure 1: The Integrated System

```
    (state(S), population(S,P))))).
```

What state is Texarkana located in?

```
answer(S, (state(S),
   eq(C,cityid(texarkana,_)), loc(C,S))).
```

CHILL treats parser induction as a problem of learning rules to control the actions of a shift-reduce parser. During parsing, the current context is maintained in a stack and a buffer containing the remaining input. When parsing is complete, the stack contains the representation of the input sentence. There are three types of operators used to construct logical queries. One is the introduction onto the stack of a predicate needed in the sentence representation due to the appearance of a phrase at the front of the input buffer. A second type of operator unifies variables appearing in stack items. Finally, a stack item may be embedded as an argument of another stack item. The introduction operators require a semantic lexicon as background knowledge. By using WOLFIE, the lexicon can be provided automatically. Figure 1 illustrates the complete system.

Problem Definition

A semantic lexicon learner is presented with a set of sentences, each consisting of an ordered list of words and annotated with a semantic representation in the form of a labeled tree; the goal is to find a semantic lexicon consistent with this data. Such a lexicon consists of *(phrase, meaning)* pairs (e.g., ([**biggest**], `largest(_,_)`)), where the phrases and their meanings are extracted from the input sentences and their representations, respectively, such that each sentence's representation can be composed from a set of components each chosen from the possible meanings of a phrase appearing in the sentence. Such a lexicon is said to *cover* the corpus. We will also refer to the coverage of components of a representation (or sentence/representation pair) by a lexicon entry. Ideally, the goal is to minimize the ambiguity and size of the learned lexicon, since this should improve accuracy and ease parser acquisition. Note that this notion of semantic lexicon acquisition is distinct from learning selectional restrictions (Manning 1993; Brent 1991) or clusters of semantically similar words (Riloff & Sheperd 1997).

Note that we allow phrases to have multiple meanings (homonymy) and multiple phrases to have the same meaning (synonymy). Also, some phrases may have a null meaning. We make only a few fairly straightforward assumptions. First is *compositionality*: the meaning of a sentence is composed from the meanings of phrases in that sentence. Since we allow multiword phrases in the lexicon (e.g., ([**kick the bucket**], `die(_)`)), this assumption seems fairly unproblematic. Second, we assume each component of the representation is due to the meaning of exactly one word or phrase in the sentence, and not more than one or to an external source such as noise. Third, we assume the meaning for each word in a sentence appears at most once in the sentence's representation. Finally, we assume that a phrase's meaning is a connected subgraph of a sentence's representation, not a more distributed representation. The second and third assumptions are preliminary, and we are exploring methods for relaxing them. If any of these assumptions are violated, WOLFIE may not learn a covering lexicon; however, the system can still be run and produce a potentially useful result.

For each phrase, p (of at most two words):
  1.1) Collect the training examples in which p appears
  1.2) Calculate LICS from (sampled) pairs of these examples' representations
  1.3) For each l in the LICS, add (p, l) to the set of candidate lexicon entries
Until the input representations are covered, or there are no remaining candidate lexicon entries do:
  2.1) Add the best (phrase, meaning) pair to the lexicon
  2.2) Update candidate meanings of phrases occurring in the same sentences as the phrase just learned
Return the lexicon of learned (phrase, meaning) pairs.

Figure 2: WOLFIE Algorithm Overview

The WOLFIE Algorithm and an Example

The WOLFIE algorithm outlined in Figure 2 has been implemented to handle two kinds of semantic representations: a case-role form based on *conceptual dependency* (Schank 1975) and a logical query language illustrated above. The current paper will focus on the latter; the changes required for the former are minimal. In order to limit search, a form of greedy set covering is used to find a covering lexicon. The first step is to derive an initial set of candidate meanings for each possible phrase. The current implementation is limited to one and two word phrases, but easily extended to longer phrases with a linear increase in complexity. For example, consider the following corpus:

1. What is the capital of the state with the biggest population?

```
answer(C, (capital(S,C),
    largest(P, (state(S), population(S,P))))).
```

2. What is the highest point of the state with the biggest area?

```
answer(P, (high_point(S,P),
    largest(A, (state(S), area(S,A))))).
```

3. What state is Texarkana located in?

```
answer(S, (state(S), eq(C,cityid(texarkana,_)),
    loc(C,S))).
```

4. What capital is the biggest?

```
answer(A, largest(A, capital(A))).
```

5. What is the area of the United States?

```
answer(A, (area(C,A), eq(C,countryid(usa)))).
```

6. What is the population of a state bordering Minnesota?

```
answer(P, (population(S,P), state(S),
    next_to(S,M), eq(M,stateid(minnesota)))).
```

7. What is the highest point in the state with the capital Madison?

```
answer(C, (high_point(B,C), loc(C,B), state(B),
    capital(B,A), eq(A,cityid(madison,_)))).
```

Although not required, for simplification, assume sentences are stripped of phrases that we know have empty meanings ([**what**], [**is**], [**with**], [**the**]) and that it is known that some phrases refer directly to given database constants (e.g., location names).

Initial candidate meanings are produced by computing the common substructures between pairs of representations of sentences that contain a given phrase. This is performed by computing their Largest Isomorphic Connected Subgraphs (LICS), taking labels into account in the isomorphism. The Largest Common Subgraph problem is solvable in polynomial time if, as we assume, both inputs are trees (Garey & Johnson 1979). The exact algorithm is complicated a bit by variables and conjunction. Therefore, we use LICS with an addition similar to computing the Least General Generalization (LGG) of first-order clauses (Plotkin 1970), i.e., the most specific clause subsuming two given clauses. Specifically, we find the LICS between two trees and then compute the LGG of the resulting subexpressions. The sets of initial candidate meanings for some of the phrases in the sample corpus (after removing the mandatory `answer` predicate) are:

Phrase	LICS	From Sent's
[**capital**]:	`largest(_,_)`	1,4
	`(capital(A,_), state(A))`	1,7
[**biggest**]:	`largest(_,state(_))`	1,2
	`largest(_,_)`	1,4;2,4
[**state**]:	`largest(_,state(_))`	1,2
	`state(_)`	1,3;2,3
	`(capital(A,_), state(A))`	1,7
	`(high_point(B,_), state(B))`	2,7
	`(state(S),loc(_,S))`	3,7
[**highest point**]:	`(high_point(B,_), state(B))`	2,7
[**located**]:	`(state(S), loc(_,S))`	3
[**in**]:	`(state(S), loc(_,S))`	3,7

Note that [**state**] has five candidate meanings, each generated from a different pair of representations of sentences in which it appears. For phrases appearing in only one sentence (e.g., [**located**]), the entire sentence representation is used as an initial candidate meaning. Such candidates are typically generalized in step 2.2 to only the correct portion of the representation before they are added to the lexicon.

After deriving initial candidates, the greedy search begins. The heuristic used to evaluate candidates is the sum of two weighted components, where p is the phrase and m its candidate meaning:

1. $P(m|p) \times P(p|m) \times P(m) = P(p) \times P(m|p)^2$
2. The generality of m

The first component is analogous the the cluster evaluation heuristic used by COBWEB (Fisher 1987). The goal is to maximize the probability of predicting the correct meaning for a randomly sampled phrase. The equality holds by Bayes Theorem. Looking at the right side, $P(m|p)^2$ is the expected probability that meaning m is correctly guessed for a given phrase, p. This assumes a strategy of probability matching, in which a a meaning m is chosen for p with probability $P(m|p)$ and correct with the same probability. The other term, $P(p)$, biases the component by how common the phrase is. Interpreting the left side of the equation, the first term biases towards lexicons with low ambiguity, the second towards low synonymy, and the third towards frequent meanings. The probabilities are estimated from the training data and then updated as learning progresses to account for phrases and meanings already covered.

The second component, *generality*, is computed as the negation of the number of nodes in the meaning's tree structure, and helps prefer smaller, more general meanings. In this example and all experiments, we use a weight of 10 for the first component of the heuristic, and a weight of 1 for the second. The first component has smaller absolute values and is therefore given a higher weight. Results are not overly-sensitive to the weights and automatically setting them using cross-validation on the training set (Kohavi & John 1995) had little effect on overall performance. To break ties, less ambiguous (those with currently fewer meanings) and shorter phrases are preferred. Below we illustrate the calculation of the heuristic measure for some of the above twelve pairs, and the resulting value for all.

([**capital**], `largest(_,_)`): $10(2^2/3) + 1(-1) = 12.33$,
([**capital**], `(capital(A,_), state(A))`): 11.33
([**biggest**], `largest(_,_)`): 29,
([**biggest**], `largest(_,state(_))`): 11.3,
([**state**], `largest(_,state(_))`): 8,
([**state**], `state(_)`): $10(4^2/4) + 1(-1) = 39$,
([**state**], `(capital(A,_), state(A))`): 8,
([**state**], `(high_point(B,_), state(B))`): 8,
([**state**], `(state(S), loc(_,S))`): 8
([**highest point**], `(high_point(B,_), state(B))`):
$10(2^2/2) + 1(-2) = 18$,
([**located**], `(state(S), loc(_,S))`):
$10(1^2/1) + 1(-2) = 8$,

([**in**], `(state(S), loc(_,S))`): 18.
The highest scoring pair is ([**state**], `state(_)`), so it is added to the lexicon.

Next is the candidate generalization phase (step 2.2). One of the key ideas of the algorithm is that each (phrase, meaning) choice can constrain the candidate meanings of phrases yet to be learned. Given the assumption that each portion of the representation is due to at most one phrase in the sentence, once part of a representation is covered, no other phrase in the sentence can be paired with that meaning (at least for that sentence). Therefore, in this step the meaning of a new lexicon entry is potentially removed from the candidate meanings of other words occurring in the same sentences. In our example, the learned pair covers all occurrences of [**state**], so remaining meanings for it are removed from the candidate set. However, now that `state(_)` is covered, the candidates for several other words are generalized. For example, the meaning `(capital(A,_), state(A))` for [**capital**], is generalized to `capital(_,_)`, with a new heuristic value of $10(2^2/3) + 1(-1) = 12.3$. Subsequently, the greedy search continues until the resulting lexicon covers the training corpus, or until no candidate phrase meanings remain.

Experimental Results

This section describes results on a database query application. The first corpus discussed contains 250 questions about U.S. geography. This domain was originally chosen due to the availability of a hand-built natural language interface, *Geobase*, to a database containing about 800 facts. It was supplied with Turbo Prolog 2.0 (Borland International 1988), and designed specifically for this domain. The corpus was assembled by asking undergraduate students to generate English questions for this database. To broaden the test, we had the same 250 sentences translated into Spanish, Turkish, and Japanese. The Japanese translations are in word-segmented Roman orthography.

To evaluate the learned lexicons, we measured their utility as background knowledge for Chill. This is performed by choosing a random set of 25 test examples and then creating lexicons and parsers using increasingly larger subsets of the remaining 225 examples. The test examples are parsed using the learned parser, the resulting queries submitted to the database, the answers compared to those generated by submitting the correct representation, and the percentage of correct answers recorded. By using the difficult "gold standard" of retrieving a correct answer, we avoid measures of partial accuracy which we believe do not adequately measure final utility. We repeated this process for ten different random training and test sets and evaluate performance differences using a two-tailed, paired t-test with a significance level of $p \leq 0.05$.

We compared our system to an incremental (on-line) lexicon learner developed by Siskind (1996), originally

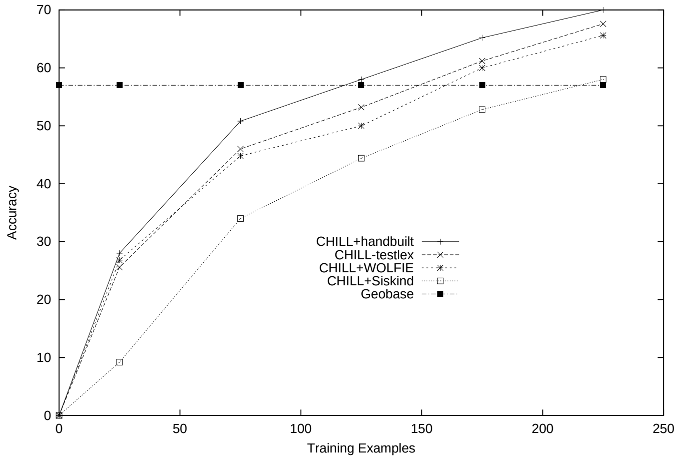

Figure 3: Accuracy on English Geography Corpus

evaluated only on artificial data. To make a more equitable comparison to our batch algorithm, we ran his in a "simulated" batch mode, by repeatedly presenting the corpus 500 times, analogous to running 500 epochs to train a neural network. We also removed Wolfie's ability to learn phrases of more than one word, since the current version of Siskind's system does not have this ability. We also made comparisons to the parsers learned by Chill when using a hand-coded lexicon as background knowledge.

In this and similar applications, there are many terms, such as state and city names, whose meanings can be automatically extracted from the database. Therefore, all tests below were run with such names given to the learner as an initial lexicon; this is helpful but not required.

The first experiment was a comparison of the two systems on the original English corpus. Figure 3 shows learning curves for Chill when using the lexicons learned by Wolfie (CHILL+WOLFIE) and by Siskind's system (CHILL+Siskind). The uppermost curve (CHILL+handbuilt) shows Chill's performance when given the hand-built lexicon. CHILL-testlex shows the performance when words that never appear in the training data are deleted from the hand-built lexicon (since a learning algorithm has no chance of getting these). Finally, the horizontal line shows the performance of the *Geobase* benchmark.

The results show that a lexicon learned by Wolfie led to parsers that were almost as accurate as those generated using a hand-built lexicon. The best accuracy is achieved by the hand-built lexicon, followed by the hand-built lexicon with words only in the test set removed, followed by Wolfie, followed by Siskind's system. All the systems do as well or better than *Geobase* by 225 training examples. The differences between Wolfie and Siskind's system are statistically significant at all training example sizes except 125. These results show that Wolfie can learn lexicons that lead

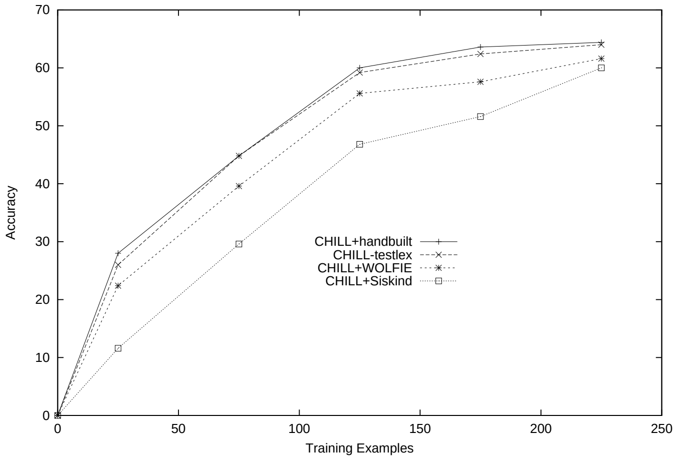

Figure 4: Accuracy on Spanish

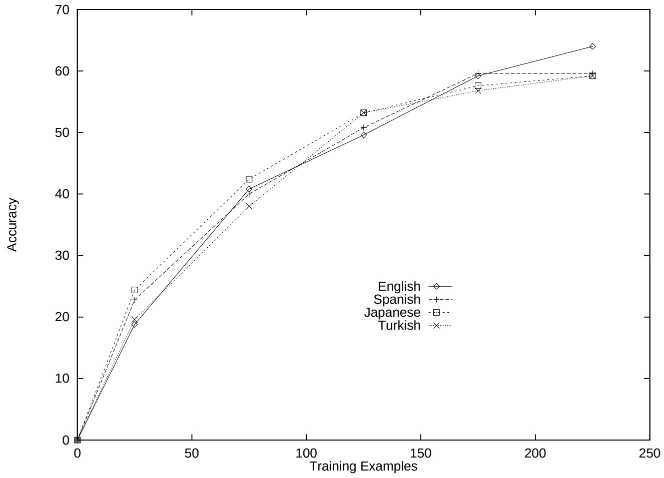

Figure 5: Accuracy on All Four Languages

to successful parsers, and that are better from this perspective than those learned by a competing system. Also, comparing to the CHILL-testlex curve, we see that most of the drop in accuracy from a hand-built lexicon is due to words in the test set that the system has not seen during training.

One of the implicit hypotheses of our problem definition is that coverage of the training data implies a good lexicon. The results show a coverage of 100% of the 225 training examples for WOLFIE versus 94.4% for Siskind. In addition, the lexicons learned by Siskind's system were more ambiguous and larger than those learned by WOLFIE. WOLFIE's lexicons had an average of 1.1 meanings per word, and an average size of 56.5 words (after 225 training examples) versus 1.7 meanings per word and 154.8 entries in Siskind's lexicons. For comparison, the hand-built lexicon had 88 entries and 1.2 meanings per word on average. These differences undoubtedly contribute to the final performance differences.

Figure 4 shows the results on the Spanish version of the corpus. In these tests, we gave closed class words to the lexicon learners as background knowledge since a similar addition for English improved performance slightly. Though the performance compared to a hand-built lexicon is not quite as close as in English, the accuracy of the parser using the learned lexicon is very similar.

Figure 5 shows the results for all four languages without any information about closed-class words. The performance differences among the four languages are quite small, demonstrating that our methods are not language dependent.

Finally, we present results on a larger, more diverse corpus from the geography domain, where the additional sentences were collected from computer science undergraduates in an introductory AI course. The set of questions in the previous experiments was collected from students in a German class, with no special instructions on the complexity of queries desired. The AI students tended to ask more complex queries: their task was to give 5 sentences and the associated logical query representation for a homework assignment. They were requested to give at least one sentence whose representation included a predicate containing embedded predicates, for example `largest(S, state(S))`, and we asked for variety in their sentences. There were 221 new sentences, for a total of 471 (including the original 250 sentences).

For these experiments, we split the data into 425 training sentences and 46 test sentences, for 10 random splits, then trained WOLFIE and then CHILL as before. Our goal was to see whether WOLFIE was still effective for this more difficult corpus, since there were approximately 40 novel words in the new sentences. Therefore, we tested against the performance of CHILL with an extended hand-built lexicon. For this test, we gave the system access to background knowledge about closed class words. We did not use phrases of more than one word, since these do not seem to make a significant difference in this domain.

Figure 6 shows the resulting learning curves. None of the differences between CHILL and WOLFIE are statistically significant, probably because the difficulty of parsing overshadows errors in word learning. Also, the improvement of machine learning methods over the Geobase hand-built interface is much more dramatic for this corpus.

Related Work

Work on automated lexicon and language acquisition dates back to Siklossy (1972), who demonstrated a system that learned transformation patterns from logic back to natural language. More recently, Pedersen & Chen (1995) describe a method for acquiring syntactic and semantic features of an unknown word, assuming access to an initial concept hierarchy, but they give no experimental results. Manning (1993)

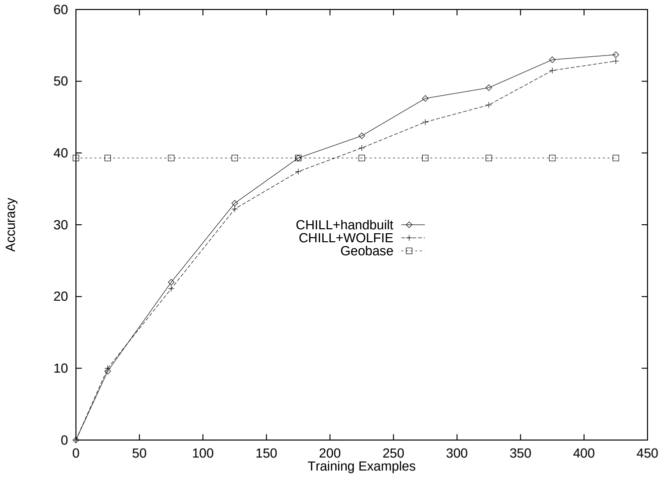

Figure 6: Accuracy on the Larger Geography Corpus

and Brent (1991) acquire subcategorization information for verbs, which is different from the information required for mapping to semantic representation. Several systems (Knight 1996; Hastings & Lytinen 1994; Russell 1993) learn new words from context, assuming that a large initial lexicon and parsing system are already available. Tishby and Gorin (1994) learn associations between words and actions (as meanings of those words). Their system was tested on a corpus of sentences paired with representations but they do not demonstrate the integration of learning a semantic parser using the learned lexicon.

The aforementioned work by Siskind is the closest. His approach is somewhat more general in that it handles noise and referential uncertainty (multiple possible meanings for a sentence), while ours is specialized for applications where a single meaning is available. The experimental results in the previous section demonstrate the advantage of our method for such an application. His system does not currently handle multiple-word phrases. Also, his system operates in an incremental or on-line fashion, discarding each sentence as it processes it, while ours is batch. While he argues for psychological plausibility, we do not. In addition, his search for word meanings is most analogous to a version space search, while ours is a greedy search. Finally, his system does not compute statistical correlations between words and their possible meanings, while ours does.

His system proceeds in two stages, first learning what *symbols* are part of a word's meaning, and then learning the structure of those symbols. For example, it might first learn that `capital` is part of the meaning of **capital**, then in the second stage learn that `capital` can have either one or two arguments. By using common substructures, we can combine these two stages in WOLFIE.

This work also has ties to the work on automatic construction of translation lexicons (Wu & Xia 1995; Melamed 1995; Kumano & Hirakawa 1994; Catizone, Russell, & Warwick 1993; Gale & Church 1991). While most of these methods also compute association scores between pairs (in their case, word/word pairs) and use a greedy algorithm to choose the best translation(s) for each word, they do not take advantage of the constraints between pairs. One exception is Melamed (1996); however, his approach does not allow for phrases in the lexicon or for synonymy within one text segment, while ours does.

Future Work

Although the current greedy search method has performed quite well, a better search heuristic or alternative search strategy could result in improvements. A more important issue is lessening the burden of building a large annotated training corpus. We are exploring two options in this regard. One is to use *active learning* (Cohn, Atlas, & Ladner 1994) in which the system chooses which examples are most usefully annotated from a larger corpus of unannotated data. This approach can dramatically reduce the amount of annotated data required to achieve a desired accuracy (Engelson & Dagan 1996). Initial promising results for semantic parser acquisition are given in Thompson (1998).

A second avenue of exploration is to apply our approach to learning to parse into more popular SQL database queries. Such corpora should be easily constructible by recording queries submitted to existing SQL applications along with their original English forms, or translating existing lists of SQL queries into English (presumably an easier direction to translate). The fact that the same training data can be used to learn both a semantic lexicon and a parser also helps limit the overall burden of constructing a complete natural language interface.

Conclusions

Acquiring a semantic lexicon from a corpus of sentences labeled with representations of their meaning is an important problem that has not been widely studied. WOLFIE demonstrates that a fairly simple greedy symbolic learning algorithm performs fairly well on this task and obtains performance superior to a previous lexicon acquisition system on a corpus of geography queries. Our results also demonstrate that our methods extend to a variety of natural languages besides English.

Most experiments in corpus-based natural language have presented results on some subtask of natural language, and there are few results on whether the learned subsystems can be successfully integrated to build a complete NLP system. The experiments presented in this paper demonstrated how two learning systems, WOLFIE and CHILL were successfully integrated to learn a complete NLP system for parsing database queries into executable logical form given only a single corpus of annotated queries.

Acknowledgements

We would like to thank Jeff Siskind for providing us with his software, and for all his help in adapting it for use with our corpus. Thanks also to Agapito Sustaita, Esra Erdem, and Marshall Mayberry for their translation efforts. This research was supported by the National Science Foundation under grants IRI-9310819 and IRI-9704943.

References

Borland International. 1988. *Turbo Prolog 2.0 Reference Guide*. Scotts Valley, CA: Borland International.

Brent, M. 1991. Automatic acquisition of subcategorization frames from untagged text. In *Proceedings of the 29th Annual Meeting of the Association for Computational Linguistics*, 209–214.

Catizone, R.; Russell, G.; and Warwick, S. 1993. Deriving translation data from bilingual texts. In *Proceedings of the First International Lexical Acquisition Workshop*.

Cohn, D.; Atlas, L.; and Ladner, R. 1994. Improving generalization with active learning. *Machine Learning* 15(2):201–221.

Engelson, S., and Dagan, I. 1996. Minimizing manual annotation cost in supervised training from corpora. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*.

Fisher, D. H. 1987. Knowledge acquisition via incremental conceptual clustering. *Machine Learning* 2:139–172.

Gale, W., and Church, K. 1991. Identifying word correspondences in parallel texts. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*.

Garey, M., and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. New York, NY: Freeman.

Hastings, P., and Lytinen, S. 1994. The ups and downs of lexical acquisition. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 754–759.

Knight, K. 1996. Learning word meanings by instruction. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, 447–454.

Kohavi, R., and John, G. 1995. Automatic parameter selection by minimizing estimated error. In *Proceedings of the Twelfth International Conference on Machine Learning*, 304–312.

Kumano, A., and Hirakawa, H. 1994. Building an MT dictionary from parallel texts based on linguistic and statistical information. In *Proceedings of the Fifteenth International Conference on Computational Linguistics*.

Lavrač, N., and Džeroski, S. 1994. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.

Manning, C. D. 1993. Automatic acquisition of a large subcategorization dictionary from corpora. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, 235–242.

Melamed, I. 1995. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. In *Proceedings of the Third Workshop on Very Large Corpora*.

Melamed, I. 1996. Automatic construction of clean broad-coverage translation lexicons. In *Second Conference of the Association for Machine Translation in the Americas*.

Muggleton, S. H., ed. 1992. *Inductive Logic Programming*. New York, NY: Academic Press.

Pedersen, T., and Chen, W. 1995. Lexical acquisition via constraint solving. In *Papers from the 1995 AAAI Symposium on the Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, 118–122.

Plotkin, G. D. 1970. A note on inductive generalization. In Meltzer, B., and Michie, D., eds., *Machine Intelligence (Vol. 5)*. New York: Elsevier North-Holland.

Riloff, E., and Sheperd, K. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, 117–124.

Russell, D. 1993. *Language Acquisition in a Unification-Based Grammar Processing System Using a Real World Knowledge Base*. Ph.D. Dissertation, University of Illinois, Urbana, IL.

Schank, R. C. 1975. *Conceptual Information Processing*. Oxford: North-Holland.

Siklossy, L. 1972. Natural language learning by computer. In Simon, H. A., and Siklossy, L., eds., *Representation and meaning: Experiments with Information Processsing Systems*. Englewood Cliffs, NJ: Prentice Hall.

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

Thompson, C. A. 1998. *Semantic Lexicon Acquisition for Learning Natural Language Interfaces*. Ph.D. Dissertation, Department of Computer Sciences, University of Texas, Austin, TX. Also appears as Artificial Intelligence Laboratory Technical Report AI 99-278 (see http://www.cs.utexas.edu/users/ai-lab).

Tishby, N., and Gorin, A. 1994. Algebraic learning of statistical associations for language acquisition. *Computer Speech and Language* 8:51–78.

Wu, D., and Xia, X. 1995. Large-scale automatic extraction of an English-Chinese translation lexicon. *Machine Translation* 9(3-4):285–313.

Zelle, J. M., and Mooney, R. J. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*.