Semantic Lexicon Acquisition for Learning Natural Language Interfaces

Cynthia A. Thompson and Raymond J. Mooney
Department of Computer Sciences
University of Texas
Austin, TX 78712
cthomp@cs.utexas.edu, 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 representations of their meaning. 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 comparable system developed by Siskind (1996).

1 Introduction & Overview

The application of learning methods to natural-language processing (NLP) is a growing area. Using machine learning to help automate the construction of NLP systems can eliminate much of the difficulty of building such systems by hand. 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, as noted by Copestake et al. (1995) and Walker and Amsler (1986). In addition, new lexicons are needed when transferring a system to new applications or domains. Johnston et al. (1995) also discuss the need for systems that can learn the meanings of novel words.

This paper describes a system, WOLFIE (WOrd Learning From Interpreted Examples), that learns a semantic lexicon of word/meaning pairs from input consisting of sentences paired with semantic representations. The goal of this research is to automate lexicon construction for an integrated NLP system that acquires semantic parsers and lexicons. A subgoal is to learn a lexicon that is as good or better than a manually-built one based on performance on a chosen task.

Although a few others (Siskind, 1996; Hastings and Lytinen, 1994; Brent, 1991) have presented systems for semantic lexicon acquisition, this work is unique in combining several features. First, interaction with a system, CHILL (Zelle, 1995), that learns to parse sentences into their semantic representations, is demonstrated. Second, it uses a fairly simple batch, greedy algorithm that is quite fast and accurate. Third, it is easily extendible to new representation formalisms. Finally, it is able to bootstrap from an existing lexicon.

We tested WOLFIE on its ability to acquire a semantic lexicon for the task of answering geographical database queries, using a corpus of queries collected from human subjects and annotated by an expert with their executable logical form. To perform 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 system performs nearly as accurately at answering novel questions when using a learned lexicon as compared to a hand-built lexicon. The system is also compared to an alternative lexicon acquisition system developed by Siskind (1996), demonstrating superior performance on this task. Finally, we translated the corpus from English into Spanish, Japanese, and Turkish and ran experiments on learning database interfaces with these languages as well.

Overall, the results demonstrate a robust ability to acquire accurate lexicons to be directly used for semantic parsing. Because we have developed this integrated system, the task of building a semantic parsing system for a new domain is simplified. One now only needs to build one representative corpus of sentence/representation pairs for the new domain. This one corpus allows the acquisition of both a semantic lexicon and a semantic parser that can together process that corpus.

2 Background

The output produced by WOLFIE can be used to assist a larger language acquisition system; in particular, it is currently used as part of the input to a parser acquisition system called CHILL (Constructive Heuristics Induction for Language Learning). CHILL uses *inductive logic programming* (Muggleton, 1992; Lavrač and Džeroski, 1994) to learn a deterministic shift-reduce parser written in

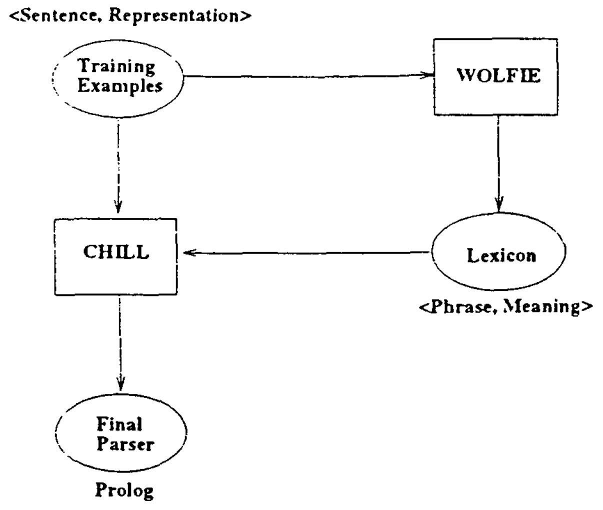

Figure 1: The Integrated System

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.

CHILL requires a lexicon as background knowledge in order to learn to parse into deeper semantic representations. By using WOLFIE, the lexicon can be provided automatically, easing the task of parser acquisition. Figure 1 illustrates the inputs and outputs of the complete system. The output of WOLFIE is a lexicon of *(phrase, meaning)* pairs; these aspects will be discussed more thoroughly in the following sections.

One of the components of CHILL is an initial *overly-general parser*, used to analyze the training data. This initial parser is specialized by the learner to generate only correct parses for the training examples. Given a correct lexicon, the overly-general parser should be able to parse all of the training examples.

In this paper, we limit our discussion of CHILL to its ability to learn parsers that map natural-language questions directly into Prolog queries that can be executed to produce an answer (Zelle and 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,
    (state(S), population(S,P))))).
```
What state is Texarkana located in?
```
answer(S, (state(S),
   eq(C,cityid(texarkana,_)),
   loc(C,S))).
```

Given a sufficient corpus of such sentence/representation pairs, CHILL is able to learn a parser that correctly parses many novel sentences into logical queries.

CHILL treats parser induction as a problem of learning rules to control the actions of the shift-reduce parser mentioned above. During parsing, the current context is contained in the contents of 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 by the parser to construct logical queries. One is the introduction onto the stack of a predicate needed in the sentence representation, due to the appearance a phrase at the front of the input buffer. The semantic lexicon associates phrases and their representations for use by this type of operator. A second type of operator unifies variables appearing in stack items. For example, in the first representation of a sample query given above, the first argument of `answer` is unified with the second argument of `capital`. Finally, a stack item may be embedded into the argument of another stack item, as is required for the first sentence/representation pair given above, to embed `state(_)` and `population(_,_)` into the second argument of `largest`.

In sum, we concentrate on using machine learning methods to build a system for processing sentences in a narrow domain, but with the goal of obtaining deep semantic representations. This is in contrast to work in this general area that attempts to process broader corpora, but only obtains shallow representations as a result of processing.

3 The Semantic Lexicon Acquisition Problem

We now define the learning problem at hand. Given a set of sentences, each consisting of an ordered list of words and annotated with a single semantic representation, we assume that each representation can be `fractured` into all of its components (Siskind, 1992). The fracturing method depends upon the given representation and must be explicitly provided or implicit in the algorithm that forms hypotheses for word meanings. Given a valid set of components, they can be constructed into a valid sentence meaning using a relation we will call `compose`.

The goal is to find a semantic lexicon that will assist parsing. Such a lexicon consists of *(phrase, meaning)* pairs, 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 (unique) phrase appearing in the sentence. If such a lexicon is found, we say that the lexicon *covers* the corpus. We will also talk about the coverage of components of a representation (or sentence/representation pair) by a lexicon entry. Ideally, we would like to minimize the ambiguity and size of the learned lexicon, since this should ease the parser acquisition task. Note that this notion of semantic lexicon acquisition is distinct from work on learning selectional restrictions (Manning, 1993; Brent, 1991) and learning clusters of semantically similar words (Riloff and Sheperd, 1997).

Note that we allow phrases to have multiple meanings (homonymy) and for multiple phrases to have the same meaning (synonymy). Also, some phrases in the sentences may have a null meaning. We make only a few fairly straightforward assumptions about the input. First is *compositionality*, i.e. the meaning of a sentence is composed from the meanings of phrases in that sentence. Since we allow multi-word 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 a word or phrase in the sentence, not to an external source such as noise. Third, we assume the meaning for each word in a sentence appears only once in the sentence's representation. The second and third assumptions are preliminary, and we are exploring methods for relaxing them. If any of these assumptions are violated, we do not guarantee coverage of the training corpus; however, the system can still be run and learn a potentially useful lexicon.

4 The WOLFIE Algorithm and an Example

In order to limit search, a greedy algorithm is used to learn phrase meanings. At each step, the best phrase/meaning pair is chosen, according to a heuristic described below, and added to the lexicon. The initial list of candidate meanings for a phrase is formed by finding the common substructure between sampled pairs of representations of sentences in which the phrase appears.[1] In the current implementation, phrases are limited to at most two words. This is for efficiency reasons only, and in the future we hope to incorporate an efficient method for including potentially meaningful phrases of more than two words.

The WOLFIE algorithm, outlined in Figure 2, has been implemented to handle two kinds of semantic representations. One is a case-role meaning representation based on *conceptual dependency* (Schank, 1975). For example, the sentence "The man ate the cheese" is represented by: `[ingest, agent:[person, sex:male, age:adult], patient:[food, type:cheese]]`. Experiments in this domain were presented in Thompson (1995).

[1]We restrict ourselves to a sampled pairs instead of all pairs because this provides enough information to get good initial candidate meanings. Using all pairs is possible but not generally necessary.

```
For each phrase (of at most two words):
  1) Sample the examples in which the phrase appears
  2) Find largest common subexpressions of pairs of
     representations from these examples
Until the input representations are covered, or there are
  no remaining candidate pairs do:
    1) Add the best phrase/meaning pair to the lexicon.
    2) Constrain 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 second representation handled is the logical query representation illustrated earlier, and is the focus of the current paper. To find the common substructure between pairs of query representations, we use a method similar to finding the Least General Generalization of first-order clauses (Plotkin, 1970). However, instead of using subsumption to guide generalization, we find the set of largest common substructures that two representations share. For example, given the two queries from Section 2, the (unique) common substructure is `state(_)`.[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. This is the second step of the loop in Figure 2. Such constraints exist because of the assumption that each portion of the representation is due to at most one phrase in the sentence. Therefore, once part of a sentence's representation is covered by the meaning of one of its phrases, no other phrase in the sentence has to be paired with that meaning (for that sentence).

For example, assume we have the sentence/representation pairs in Section 2, plus the additional pair:

> 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))))).`

As a simplification, assume sentences are stripped of phrases that we know *a priori* have a null meaning (although in general this is not required). In the example sentences, these phrases are [**what**], [**is**], [**with**], and [**the**]. From these three examples, the meaning of [**state**], the only phrase common to all sentences, is determined to be `state(_)`, which is the only predicate the three representations have in common. Before determining this, the candidate meaning for

[2]Since CHILL initializes the parse stack with the `answer` predicate, it is first stripped from the input given to WOLFIE.

[biggest] is [largest(_, state(_))] (the largest substructure shared by the representations of the two sentences containing "biggest"). However, since state(_) is now covered by ([state], state(_)), it can be eliminated from consideration as part of the meaning of [biggest], and the candidate meaning for [biggest] becomes [largest(_,_)].

We now describe the algorithm in more detail. The first step is to select a random sample of the sentences that each one and two word phrase appears in, and derive an initial set of candidate meanings for each phrase. This is done by deriving common substructure between pairs of representations of sentences that contain these phrases. For example, let us suppose we have the following pairs as input:

```
What is the capital of the state with the biggest
   population?
answer(C, (capital(S,C),
   largest(P,(state(S),population(S,P))))).

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))))).

What state is Texarkana located in?
answer(S, (state(S),
   eq(C,cityid(texarkana,_)), loc(C,S))).

What is the area of the United States?
answer(A,(area(C,A),eq(C,countryid(usa)))).

What is the population of the states bordering
Minnesota?
answer(P, (population(S,P), state(S),
   next_to(S,M),eq(M,stateid(minnesota)))).
```

The sets of initial candidate meanings for some of the phrases in this corpus are:

[biggest]: [largest(_,state(_))],

[state]: [state(_), largest(_,state(_))],

[area]: [area(_)],

[population]: [(population(_,_), state(_))],

[capital]: [(capital(S,_),
 largest(P, (state(S), population(S,P))))].

Note that [state] has two candidate meanings, each generated from a different pair of representations of sentences in which it appears. A detail is that for phrases that only appear in one sentence, we use the entire representation of the sentence in which they appear as an initial candidate meaning. An example in this corpus is [capital]. As we will see, this type of pair typically has a low score, so the meaning will usually get pared down to just the correct portion of the representation, if any. Finally, if a phrase is ambiguous, the pairwise matchings to generate candidate items, together with the constraining of representations, would enable multiple meanings to be learned for it.

After deriving these initial meanings, the greedy search begins. The heuristic used to evaluate candidates has five weighted components:

1. Ratio of the number of times the phrase appears with the meaning to the number of times the phrase appears, or $P(meaning|phrase)$.
2. Ratio of the number of times the phrase appears with the meaning to the number of times the meaning appears, or $P(phrase|meaning)$.
3. Frequency of the phrase, or $P(phrase)$.
4. Percent of orthographic overlap between the phrase and its meaning.
5. The generality of the meaning.

The first measure helps reduce ambiguity (homonymy) by preferring phrases that indicate a particular meaning with high probability. The second measure helps reduce synonymy by favoring pairs in which the meaning appears with few other phrases. The third measure is used because frequent phrases are more likely to be paired with a correct meaning since we have more information about the representations of sentences in which they appear.

The fourth measure is useful in some domains since sometimes phrases have many characters in common with their meanings, as in **area** and **area(_)**. It measures the maximum number of consecutive characters in common between the phrase and the terms and predicates in the meanings, as an average of the percent of both the number of characters in the phrase and in the term and predicate names. However, as we will demonstrate in our experiments, the use of this portion of the heuristic is not required to learn useful lexicons.

The final measure, generality, measures the number of terms and predicates in the meaning. Preferring a meaning with fewer terms helps evenly distribute the predicates in a sentence's representation among the meanings of the phrases in that sentence and thus leads to a lexicon that is more likely to be correct. To see this, we note that some words are likely to co-occur with one another, and so their joint representation (meaning) is likely to be in the list of candidate meanings for both words. By preferring a more general meaning, we more easily ignore these incorrect joint meanings. In the candidate set above for example, if all else were equal, the generality portion of the heuristic would prefer state(_) over largest(_,state(_)) as the meaning of state.

For purposes of this example, we will use a weight of 50 for each of the first four parameters, and a weight of 8 for the last. The first four components have smaller values than the last, so they have higher weights. Results are not overly-sensitive to the heuristic weights. Automatically setting the weights using cross-validation on the training set (Kohavi and John, 1995) had little effect on overall performance. In all of the experiments, these same weights were used. To break ties, we choose less "ambiguous" phrases first and learn short phrases before longer ones. A phrase is considered more ambiguous if it currently has more meanings in the partially learned lexicon.

The heuristic measure for the above six pairs is:
[[**biggest**], **largest(_,state(_))**]: $50(2/2) + 50(2/2) + 50(2/21) + 50((4/12 + 4/7)/2) - 8 * 2 = 111$

[[**area**], **area(_)**]: $50(2/2) + 50(2/2) + 50(2/21) + 50((4/4 + 4/4)/2) - 8 * 1 = 147$

[[**state**], **state(_)**]: $50(3/3) + 50(3/4) + 50(3/21) + 50((5/5 + 5/5)/2) - 8 * 1 = 137$

[[**state**], **largest(_,state(_))**]: 110

[[**population**], **(population(_,_), state(_))**]: 130

[[**capital**], **(capital(S,_), largest(P, (state(S), population(S,P))))**]: 101

The best pair by our measure is ([**area**], **area(_)**), so it is added to the lexicon.

The next step in the algorithm is to constrain the remaining candidate meanings for the learned phrase, if any, so as to only consider sentences for which no meaning has yet been learned for the phrase. In our example, the learned pair covers all occurrences of [**area**], so there are no remaining meanings that need to be constrained. Next, for the remaining unlearned phrases, their candidate meanings are constrained to take into account the meaning just learned, as was discussed at the beginning of this section. In our example, learning [**area**] would not affect any of the meanings listed above, but the next best pair, [[**state**], **state(_)**], would constrain the (only) candidate meaning for [**population**] to become **population(_,_)**, the candidate meaning for [**capital**] to become **(capital(S,_), largest(P, population(S,P)))**, and the candidate meaning for [**biggest**] to become **largest(_,_)**. The greedy search continues until the lexicon covers the training corpus.

A detail of the search not yet mentioned is to check if covered sentence/representation pairs can be parsed by CHILL's overly-general parser. If this is not the case, we know that some phrase in the sentence has a meaning that is not useful to CHILL. Therefore, whenever a sentence is covered, we check whether it can be parsed. If not, we retract the most recently learned pair, and adjust that phrase's candidate meanings to omit that meaning. We call this the *parsability heuristic*.

5 Experimental Results

This section describes our experimental results on a database query application. The corpus contains 250 questions about U.S. geography paired with logical representations. This domain was chosen due to the availability of an existing hand-built natural language interface, *Geobase*, to a simple geography database containing about 800 facts. This interface was supplied with Turbo Prolog 2.0 (Borland International, 1988), and was designed specifically for this domain. The questions were collected from uninformed undergraduates and mapped into their logical form by an expert. Examples from the corpus were given in the previous sections. 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. Translated questions were paired with the appropriate logical queries from the English corpus.

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 the correct representation, and the percentage of correct answers recorded. By making a comparison to the "gold standard" of retrieving a correct answer to the original query, we avoid measures of partial accuracy which do not give a picture of the real usefulness of the parser. To improve the statistical significance of the results, we repeated the above steps for ten different random splits of the data into training and test sets. For all significance tests we used a two-tailed, paired t-test and a significance level of $p \leq 0.05$.

We compared our system to that developed by Siskind (1996). Siskind's system is an on-line (incremental) learner, while ours is batch. To make a closer comparison between the two, 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 made comparisons to the parsers learned by CHILL when using a hand-coded lexicon as background knowledge. This lexicon was available for this domain because when CHILL was originally developed, WOLFIE had not yet been developed.

In this application, there are many terms, such as state and city names, whose meanings are easily extracted from the database. Therefore, all tests below were run with such names given to the learner as an initial lexicon, although this is not required for learning in general.

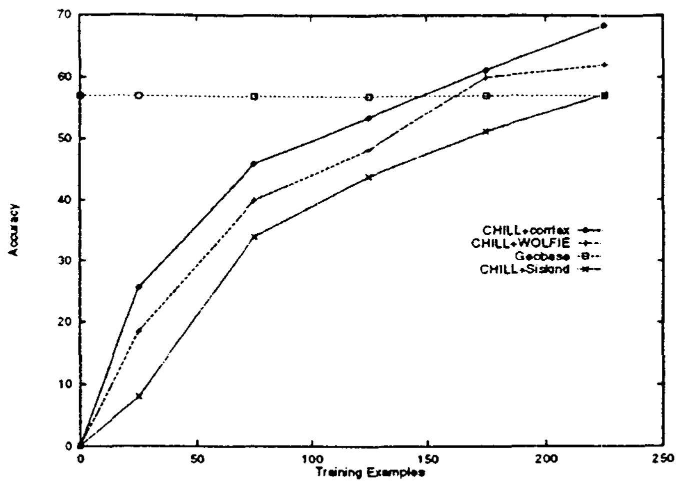

Figure 3: Accuracy on English Geography Corpus

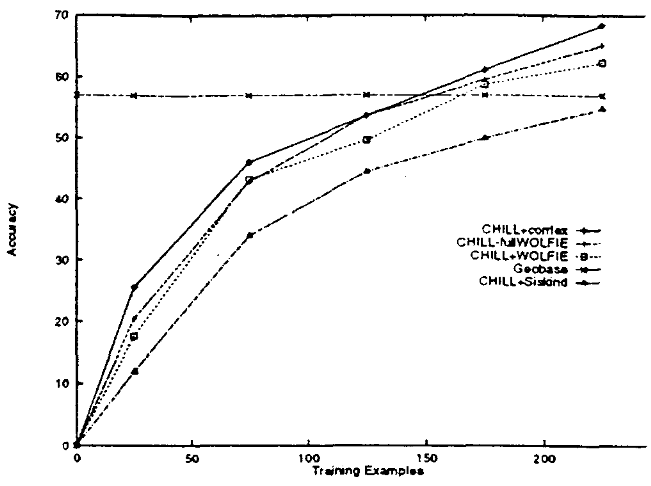

Figure 4: Accuracy Given Closed Class Words

5.1 Comparisons using English

The first experiment was a comparison of the two systems on the original English corpus. However, since Siskind has no measure of orthographic overlap, and it could arguably give our system an unfair advantage on this data, we ran WOLFIE with a weight of zero for this component. We also did not use the parsability heuristic for this test. By making these adjustments, we attempted to generate the fairest head-to-head comparison between the two systems.

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+corrlex) is CHILL's performance when given the hand-built lexicon. 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 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 25 and 175 examples. These results show that WOLFIE can learn lexicons that lead to successful learning of parsers, and that are somewhat better from this perspective than those learned by a competing system.

As noted above, these tests were run with only the meaning of database constants provided as background knowledge. Next, we examined the effect of also providing closed-class words as background knowledge. Figure 4 shows the resulting learning curves. For these tests, we also show the advantage of adding both the orthographic overlap and parsability heuristics to WOLFIE (CHILL-fullWOLFIE). Both the additional backgrou knowledge and the improved heuristic increase the ove all performance a couple of percentage points. The d ferences between Siskind's system and WOLFIE witho parsing or overlap are statistically significant at 75, 17 and 225 examples. Finally, we noted that Siskind's sy tem run in batch mode on this test averaged 54.8% at 2 examples, versus non-batch mode which attained 49.6 accuracy, giving evidence that batch mode does impro his system.

One of the implicit hypotheses of our approach is th coverage of the training pairs implies a good lexicon. V can compare the coverage of WOLFIE's lexicons to tho of Siskind's and verify that WOLFIE's have better co erage. For the first experiment above, WOLFIE cover 100% of the 225 training examples, while Siskind co ered 94.4%. For the second experiment, the coverag were 100% and 94.5%, respectively. This may accou for some of the performance difference between the tv systems.

Further differences may be explained by the percer age of training examples usable by CHILL, which is t percentage parsable by its overly-general parser. For t first experiment, CHILL could parse 93.7% of the 225 e amples when given the lexicons learned by WOLFIE b only 78% of the examples when given lexicons learn by Siskind's system. When the lexicon learners are giv closed class words, these percentages rise to 98.1% a 84.6%, respectively. In addition, the lexicons learn by Siskind's system were more ambiguous than tho learned by WOLFIE. WOLFIE's lexicons had 1.1 mea ings per word for the second experiment (after 225 trai ing examples) versus 1.7 meanings per word in Siskinc lexicons. These differences most likely contribute to t differences seen in the generalization accuracy of CHIL

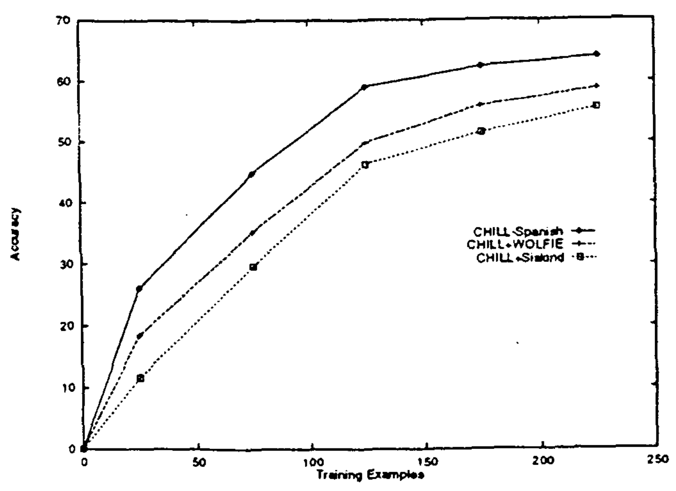

Figure 5: Accuracy on Spanish

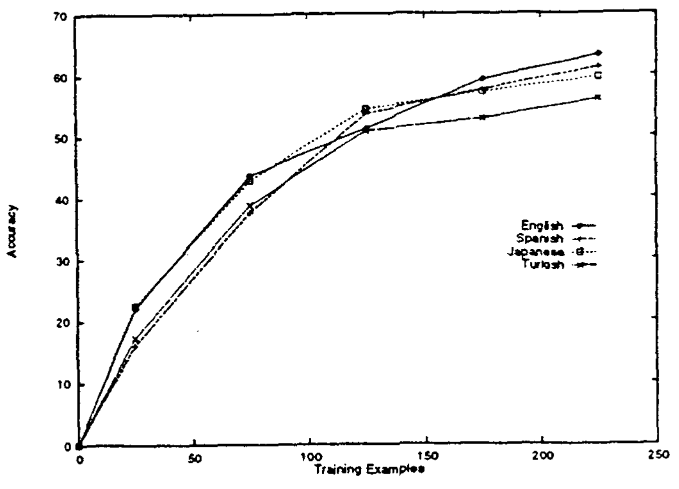

Figure 6: Accuracy on All Four Languages

The ability to learn multiple-word phrases is not a significant source of the advantage of WOLFIE over Siskind's system. since only 2% of the lexicon entries learned by WOLFIE on average contained two-word phrases.

5.2 Comparisons using Spanish

Next. we examined the performance of the two systems on the Spanish version of the corpus. We again omitted orthographic overlap and the parsability heuristic. Figure 5 shows the results. In these tests, we also gave closed class words to the lexicon learners as background knowledge, since these results were slightly better for English. 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.

5.3 Accuracy on Other Languages

We also had the geography query sentences translated into Japanese and Turkish, and ran similar tests to determine how well WOLFIE could learn lexicons for these languages, and how well CHILL could learn to parse them. Figure 6 shows the results. For all four of these tests, we used the parsability heuristic, but did not give the learner access to the closed class words of any of the languages. We also set the weight of the orthographic overlap heuristic to zero for all four languages, since this gives little advantage in the foreign languages. The performance differences among the four languages are quite small, demonstrating that our methods are not language dependent.

6 Related Work

Pedersen and Chen (1995) describe a method for acquiring syntactic and semantic features of an unknown word. They assume access to an initial concept hierarchy, and do not present any experimental results. Many systems (Fukumoto and Tsujii, 1995; Haruno, 1995; Johnston et al., 1995; Webster and Marcus, 1995) focus only on acquisition of verbs or nouns, rather than all types of words. Also, these either do not experimentally evaluate their systems, or do not show the usefulness of the learned lexicons. Manning (1993) and Brent (1991) acquire subcategorization information for verbs. Finally, several systems (Knight, 1996; Hastings and Lytinen, 1994; Russell, 1993) learn new words from context, assuming that a large initial lexicon and parsing system are available.

Tishby and Gorin (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, and perhaps most significantly, 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 and Xia, 1995; Melamed, 1995; Kumano and Hirakawa, 1994; Catizone et al., 1993; Gale and 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.

7 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 et al., 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 and Dagan, 1996).

Second, we are currently developing a corpus of sentences paired with SQL database queries. Extending our system to handle this representation should be a fairly simple matter. Such corpora should be easily constructed 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 NL interface.

On a separate note, the learning algorithm may be applicable to other domains, such as learning for translation or diagnosis. We hope to investigate these possibilities in the future as well.

8 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.

9 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. This research was supported by the National Science Foundation under grants IRI-9310819 and IRI-9704943. Thanks also to Agapito Sustaita, Esra Erdem, and Marshall Mayberry for their translation efforts.

References

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

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

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

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

A. Copestake, T. Briscoe, P. Vossen, A. Ageno, I. Castellon, F. Ribas, G. Rigan, H. Rodríguez, and A. Samiotou. 1995. Acquisition of lexical translation relations from MRDS. *Machine Translation*, 9.

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

Fumiyo Fukumoto and Jun'ichi Tsujii. 1995. Representation and acquisition of verbal polysemy. In *Papers from the 1995 AAAI Symposium on the Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, pages 39–44, Stanford, CA, March.

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

Masahiko Haruno. 1995. A case frame learning method for Japanese polysemous verbs. In *Papers from the*

1995 AAAI Symposium on the Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity, pages 45–50, Stanford, CA, March.

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

M. Johnston, B. Boguraev, and J. Pustejovsky. 1995. The acquisition and interpretation of complex nominals. In *Papers from the 1995 AAAI Symposium on the Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity, and Generativity*, pages 69–74, Stanford, CA.

Kevin Knight. 1996. Learning word meanings by instruction. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 447–454, Portland, Or, August.

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

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

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

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

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

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

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

Ted Pedersen and Weidong Chen. 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*, pages 118–122, Stanford, CA.

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

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

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

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

Jeffrey M. Siskind. 1992. *Naive Physics, Event Perception, Lexical Semantics and Language Acquisition*. Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, December.

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

Cynthia A. Thompson. 1995. Acquisition of a lexicon from semantic representations of sentences. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pages 335–337, Cambridge, MA.

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

D. Walker and R. Amsler. 1986. The use of machine-readable dictionaries in sublanguage analysis. In R. Grishman and R. Kittredge, editors, *Analyzing Language in Restricted Domains*, pages 69–83. Lawrence Erlbaum Associates, Hillsdale, NJ.

Mort Webster and Mitch Marcus. 1995. Automatic acquisition of the lexical semantics of verbs from sentence frames. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*.

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

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

J. M. Zelle. 1995. *Using Inductive Logic Programming to Automate the Construction of Natural Language Parsers*. Ph.D. thesis, Department of Computer Sciences, University of Texas, Austin, TX, August. Also appears as Artificial Intelligence Laboratory Technical Report AI 96-249.