Appears in *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL-95)*, pp.325-337, Boston, MA, July 1995

# Acquisition of a Lexicon from Semantic Representations of Sentences*

**Cynthia A. Thompson**
Department of Computer Sciences
University of Texas
2.124 Taylor Hall
Austin, TX 78712
cthomp@cs.utexas.edu

## Abstract

A system, WOLFIE, that acquires a mapping of words to their semantic representation is presented and a preliminary evaluation is performed. Tree least general generalizations (TLGGs) of the representations of input sentences are performed to assist in determining the representations of individual words in the sentences. The best guess for a meaning of a word is the TLGG which overlaps with the highest percentage of sentence representations in which that word appears. Some promising experimental results on a non-artificial data set are presented.

## 1 Introduction

Computer language learning is an area of much potential and recent research. One goal is to learn to map surface sentences to a deeper semantic meaning. In the long term, we would like to communicate with computers as easily as we do with people. Learning word meanings is an important step in this direction. Some other approaches to the lexical acquisition problem depend on knowledge of syntax to assist in lexical learning (Berwick and Pilato, 1987). Also, most of these have not demonstrated the ability to tie in to the rest of a language learning system (Hastings and Lytinen, 1994; Kazman, 1990; Siskind, 1994). Finally, unnatural data is sometimes needed (Siskind, 1994).

We present a lexical acquisition system that learns a mapping of words to their semantic representation, and which overcomes the above problems. Our system, WOLFIE (WOrd Learning From Interpreted Examples), learns this mapping from training examples consisting of sentences paired with their semantic representation. The representation used here is based on Conceptual Dependency (CD) (Schank, 1975). The results of our system can be used to assist a larger language acquisition system; in particular, we use the results as part of the input to CHILL (Zelle and Mooney, 1993). CHILL learns to parse sentences into case-role representations by analyzing a sample of sentence/case-role pairings. By extending the representation of each word to a CD representation, the problem faced by CHILL is made more difficult. Our hypothesis is that the output from WOLFIE can ease the difficulty.

In the long run, a system such as WOLFIE could be used to help learn to process natural language queries and translate them into a database query language. Also, WOLFIE could possibly assist in translation from one natural language to another.

## 2 Problem Definition and Algorithm

### 2.1 The Lexical Learning Problem

**Given:** A set of sentences, $S$ paired with representations, $R$.
**Find:** A pairing of a subset of the words, $W$ in $S$ with representations of those words.

Some sentences can have multiple representations because of ambiguity, both at the word and sentence level. The representations for a word are formed from subsets of the representations of input sentences in which that word occurred. This assumes that a representation for some or all of the words in a sentence is contained in the representation for that sentence. This may not be true with all forms of sentence representation, but is a reasonable assumption.

Tree least general generalizations (TLGGs) plus statistics are used together to solve the problem. We make no assumption that each word has a single meaning (i.e., homonymy is allowed), or that each meaning is associated with one word only (i.e., synonymy is allowed). Also, some words in $S$ may not have a meaning associated with them.

### 2.2 Background: Tree Least General Generalizations

The input to a TLGG is two trees, and the outputs returned are common subtrees of the two input trees.

*This research was supported by the National Science Foundation under grant IRI-9310819

Our trees have labels on their arcs; thus a tree with root `p`, one child `c`, and an arc label to that child `l` is denoted `[p,l:c]`. TLGGs are related to the LGGs of (Plotkin, 1970). Summarizing that work, the LGG of two clauses is the least general clause that subsumes both clauses. For example, given the trees

`[ate,agt:[person,sex:male,age:adult],`
`  pat:[food,type:cheese]]`
and `[hit,inst:[inst,type:ball],`
`  pat:[person,sex:male,age:child]]`

the TLGGs are `[person,sex:male]` and `[male]`. Notice that the result is not unique, since the algorithm searches all subtrees to find commonalities.

### 2.3 Algorithm Description

Our approach to the lexical learning problem uses TLGGs to assist in finding the most likely meaning representation for a word. First, a table, $T$ is built from the training input. Each word, $W$ in $S$ is entered into $T$, along with the representations, $R$ of the sentences $W$ appeared in. We call this the representation set, $W_R$. If a word occurs twice in the same sentence, the representation of that sentence is entered twice into $W_R$. Next, for each word, several TLGGs of pairs from $W_R$ are performed and entered into $T$. These TLGGs are the possible meaning representations for a word. For example, `[person,sex:male,age:adult]` is a possible meaning representation for `man`. More than one of these TLGGs could be the correct meaning, if the word has multiple meanings in $R$. Also, the word may have no associated meaning representation in $R$. "The" plays such a role in our data set.

Next, the main loop is entered, and greedy hill climbing on the best TLGG for a word is performed. A TLGG is a good candidate for a word meaning if it is part of the representation of a large percentage of sentences in which the word appears. The best word-TLGG pair in $T$, denoted $(w, t)$ is the one with the highest percentage of this overlap. At each iteration, the first step is to find and add to the output this best $(w, t)$ pair. Note that $t$ can also be part of the representation of a large percentage of sentences in which another word appears, since we can have synonyms in our input.

Second, one copy of each sentence representation that has $t$ somewhere in it is removed from $w$'s entry in $T$. The reason for this is that the meaning of $w$ for those sentences has been learned, and we can gain no more information from those sentences. If $t$ occurs $n$ times in one of these sentence representations, the sentence representation is removed $n$ times, since we add one copy of the representation to $w_R$ for each occurrence of $w$ in a sentence.

Finally, for each $word \in T$, if $word$ and $w$ appear in one or more sentences together, the sentence representations in $word$'s entry that correspond to such sentences are modified by eliminating the portion of the sentence representation that matches $t$, thus shortening that sentence representation for the next iteration. This prevents us from mistakenly choosing the same meaning for two different words in the same sentence. This elimination might not always succeed since $w$ can have multiple meanings, and it might be used in a different way than that indicated by $t$ in the sentence with both $w$ and $word$ in it. But if it does succeed the TLGG list for $word$ is modified or recomputed as needed, so as to still accurately reflect the (now modified) sentence representations for $word$. Loop iteration continues until all $W \in T$ have no associated representations.

### 2.4 Example

Let us illustrate the workings of WOLFIE with an example. Consider the following input:

1. The boy hit the window.
   [propel,agt:[person,sex:male,age:child],
   pat:[obj,type:window]]
2. The hammer hit the window.
   [propel,inst:[obj,type:hammer],
   pat:[obj,type:window]]
3. The hammer moved.
   [ptrans,pat:[obj,type:hammer]]
4. The boy ate the pasta with the cheese.
   [ingest,agt:[person,sex:male,age:child],
   pat:[food,type:pasta,accomp:[food,type:cheese]]]
5. The boy ate the pasta with the fork.
   [ingest,agt:[person,sex:male,age:child],
   pat:[food,type:pasta],inst:[inst,type:fork]]

A portion of the initial $T$ follows. The TLGGs for `boy` are [ingest, agt:[person, sex:male, age:child], pat:[food, type:pasta]], [person, sex:male, age:child], [male], [child], [food, type:pasta], [food], and [pasta]. The TLGGs for `pasta` are the same as for `boy`. The TLGGs for `hammer` are [obj, type:hammer] and [hammer].

In the first iteration, all the above words have a TLGG which covers 100% of the sentence representations. For clarity, let us choose `[person,sex:male,age:child]` as the meaning for `boy`. Since each sentence representation for `boy` has this TLGG in it, we remove all of them, and `boy`'s entry will be empty. Next, since `boy` and `pasta` appear in some sentences together, we modify the sentence representations for `pasta`. They are now as follows: [ingest,pat:[food,type:pasta,accomp:[food,type:cheese]]] and [ingest,pat:[food,type:pasta],inst:[inst,type:fork]]. We also have to modify the TLGGs, resulting in the list: [ingest,pat:[food,type:pasta]], [food,type:pasta], [food], and [pasta]. Since all of these have 100% coverage in this example set, any of them could be chosen as the meaning representation for `pasta`. Again, for clarity, we choose the correct one, and the final meaning representations for these examples would be: `(boy,[person,sex:male,` `age:child])`, `(pasta,[food,type:pasta])`, `(hammer,[obj,type:hammer])`, `(ate,[ingest])`, `(fork,[inst,type:fork])`, `(cheese,[food, type:cheese])`, and `(window,[obj,type: window])`. As noted above, in this example, there are some alternatives for the meanings for `pasta`, and also for `window` and `cheese`. In a larger example, some of these ambiguities would be eliminated, but those remaining are an area for future research.

## 3 Experimental Evaluation

Our hypothesis is that useful meaning representations can be learned by WOLFIE. One way to test this is by examining the results by hand. Another way to test this is to use the results to assist a larger learning system.

The corpus used is based on that of (McClelland and Kawamoto, 1986). That corpus is a set of 1475 sentence/case-structure pairs, produced from a set of 19 sentence templates. We modified only the case-structure portion of these pairs. There is still the basic case-structure representation, but instead of a single word for each filler, there is a semantic representation, as in the previous section.

The system is implemented in prolog. We chose a random set of training examples, starting with 50 examples, and incrementing by 100 for each of three trials. To measure the success of the system, the percentage of correct word meanings obtained was measured. This climbed to 94% correct after 450 examples, then went down to around 83% thereafter, with training going up to 650 examples. In one case, in going from 350 to 450 training examples, the number of word-meaning pairs learned went down by ten while the accuracy went up by 31%. This happened, in part, because the incorrect pair `(broke, [inst])` was hypothesized early in the loop with 350 examples, causing many of the instruments to have an incomplete representation, such as `(hatchet, [hatchet])`, instead of the correct `(hatchet, [inst,type:hatchet])`. This error was not made in cases where a higher percent of the correct word meanings were learned. It is an area for future research to discover why this error is being made in some cases but not in others.

We have only preliminary results on the task of using WOLFIE to assist CHILL. Those results indicate that CHILL, without WOLFIE's help cannot learn to parse sentences into the deeper semantic representation, but that with 450 examples, assisted by WOLFIE, it can learn parse up to 55% correct on a testing set.

## 4 Future Work

This research is still in its early stages. Many extensions and further tests would be useful. More extensive testing with CHILL is needed, including using larger training sets to improve the results. We would also like to get results on a larger, real world data set. Currently, there is no interaction between lexical and syntactic/parsing acquisition, which could be an area for exploration. For example, just learning `(ate, [ingest])` does not tell us about the case roles of `ate` (i.e., agent and optional patient), but this information would help CHILL with its learning process. Many acquisition processes are more incremental than our system. This is also an area of current research. In the longer term, there are problems such as adding the ability to: acquire one definition for multiple morphological forms of a word; work with an already existing lexicon, to revise mistakes and add new entries; map a multi-word phrase to one meaning; and many more. Finally, we have not tested the system on noisy input.

## 5 Conclusion

In conclusion, we have described a new system for lexical acquisition. We use a novel approach to learn semantic representations for words. Though in its early stages, this approach shows promise for many future applications, including assisting another system in learning to understand entire sentences.

## References


Berwick, Robert C., and Pilato, S. (1987). Learning syntax by automata induction. *Machine Learning*, 2(1):9–38.

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

Kazman, Rick (1990). Babel: A psychologically plausible cross-linguistic model of lexical and syntactic acquisition. In *Proceedings of the Eighth International Workshop on Machine Learning*, 75–79. Evanston, IL.

McClelland, James L., and Kawamoto, A. H. (1986). Mechanisms of sentence processing: Assigning roles to constituents of sentences. In Rumelhart, D. E., and McClelland, J. L., editors, *Parallel Distributed Processing, Vol. II*, 318–362. Cambridge, MA: MIT Press.

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

Schank, Roger C. (1975). *Conceptual Information Processing*. Oxford: North-Holland.

Siskind, Jeffrey M. (1994). Lexical acquisition in the presence of noise and homonymy. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 760–766.

Zelle, John M., and Mooney, Raymond J. (1993). Learning semantic grammars with constructive inductive logic programming. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 817–822. Washington, D.C.