Incremental Parameter Estimation of Stochastic Context-Free Grammars without Retention of the Sentence Corpus

Brent Heeringa
Department of Computer Science
University of Massachusetts, Amherst
140 Governor's Drive Amherst, MA 01003
heeringa@cs.umass.edu

Tim Oates
Artificial Intelligence Lab
Massachusetts Institute of Technology
545 Technology Square
Cambridge, MA 02139
oates@ai.mit.edu

Abstract

Stochastic context free grammars (SCFGs) are often used to represent the syntax of natural languages. Most algorithms for learning these grammars from data require a corpus of sentences to be stored and repearedly processed. We are interested in how embedded agents might learn the syntax of natural languages from exposure to utterances over long periods of time. In this context, the memory and computational requirements of existing algorithms for learning SCFGs from data are prohibitive. We present an online algorithm for learning the parameters of SCFGs that computes summary statistics from sentences as they are observed. The algorithm thus requires a fixed amount of space regardless of the number of sentences it processes. Despite the fact that it uses much less information than the Inside-Outside algorithm, our algorithm performs almost as well.

1 Introduction

How might an embedded agent, such as a mobile robot, acquire natural language from utterances describing its environment and activities paired with sensory information about the physical contexts in which they occurred? In other work we investigated the problem of identifying words in the speech signal and learning their denotations (Oates, 2001). In this paper we describe the next step in this line of research - learning syntax.

Although natural languages are not entirely context free, stochastic context free

grammars (SCFGs) are an effective representation for capturing much of their structure. However, for embedded agents, most algorithms for learning SCFGs from data have two shortcomings. First, they need access to a corpus of complete sentences, requiring that the agent retain every sentence it hears. Second, they are batch algorithms that make repeated passes over the data, often requiring significant computation in each pass.

To address these shortcomings we present an online algorithm called SPAN (Sample Parse Adjust Normalize) for learning the parameters of SCFGs using only summary statistics in combination with repeated sampling techniques. SCFGs contain both structure (i.e. rules) and parameters (i.e. rule probabilities). Many approaches to learning SCFGs from data alternate between improving the structure of the grammar given the parameters and then improving the parameters given the structure. Our algorithm assumes a fixed structure and attempts to find parameters that maximize the likelihood of the data. This approach is also often used to learn structure by pruning away rules with very low probabilities (e.g. (Lari & Young, 1990)).

Given a SCFG to be learned, SPAN has access to the grammar structure and a set of sentences generated by the grammar. The correct parameters are unknown. SPAN begins by parsing the sentence corpus using a chart parser. Note that the parse does not depend on the parameters but only on the structure. SPAN associates with each rule a histogram. Each histogram records the number of times its associated rule occurs in parse trees of individual sentences.

SPAN repeatedly generates a corpus of sample sentences from its grammar, a process that does depend on the parameters, and parses them. This results in a second set of histograms reflecting rule counts. Since the counts depend on the frequencies of rules which correspondingly depend on the parameters of the generating grammar, we can use the degree to which the two sets of histograms differ as a measure of the difference between SPAN's current parameter estimation and the target parameters. SPAN regenerates sentences and observes changes in its histograms, parameters are updated: How might an embedded agent, such as a mobile robot, acquire natural language from utterances describing its environment and activities paired with sensory information about the physical contexts in which they occurred? In other work we investigated the problem of identifying words in the speech signal and learning their denotations (Oates, 2001). In this paper we describe the next step in this line of research - learning syntax.

Although natural languages are not entirely context free, stochastic context free grammars (SCFGs) are an effective representation for capturing much of their structure. However, for embedded agents, most algorithms for learning SCFGs from data have two shortcomings. First, they need access to a corpus of complete sentences, requiring that the agent retain every sentence it hears. Second, they are batch algorithms that make repeated passes over the data, often requiring significant computation in each pass.

To address these shortcomings we present an online algorithm called SPAN (Sample Parse Adjust Normalize) for learning the parameters of SCFGs using only summary statistics in combination with repeated sampling techniques. SCFGs contain both structure (i.e. rules) and parameters (i.e. rule probabilities). Many approaches to learning SCFGs from data alternate between improving the structure of the grammar given the parameters and then improving the parameters given the structure. Our algorithm assumes a fixed structure and attempts to find parameters that maximize the likelihood of the data. This approach is also often used to learn structure by pruning away rules with very low probabilities (e.g. (Lari & Young, 1990)).

Given a SCFG to be learned, SPAN has access to the grammar structure and a set of

sentences generated by the grammar. The correct parameters are unknown. SPAN begins by parsing the sentence corpus using a chart parser. Note that the parse does not depend on the parameters but only on the structure. SPAN associates with each rule a histogram. Each histogram records the number of times its associated rule occurs in parse trees of individual sentences.

SPAN repeatedly generates a corpus of sample sentences from its grammar, a process that does depend on the parameters, and parses them. This results in a second set of histograms reflecting rule counts. Since the counts depend on the frequencies of rules which correspondingly depend on the parameters of the generating grammar, we can use the degree to which the two sets of histograms differ as a measure of the difference between SPAN's current parameter estimation and the target parameters. Emptical results show that this procedure yields parameters that are close to those found by the Inside-Outside algorithm.

The remainder of the paper is organized as follows. In sections 2 and 2.1 we formally present stochastic context-free grammars and discuss previous research into learning their structure and parameters. Section 3 presents SPAN and provides some algorithmic analysis. Section 5 explains some experimental results and section 6 concludes with a discussion of future work.

2 Stochastic Context Free Grammars

Stochastic Context-Free Grammars [1] (SCFGs) are the natural extension of Context-Free Grammars to the probabilistic domain (Sipser, 1997; Charniak, 1993). Said differently, they are context-free grammars with probabilities associated with each production. Formally, a SCFG is a four-tuple $M = \langle V, \Sigma, R, S \rangle$ where

1. V is a finite set of non-terminals
2. Σ is a finite set, disjoint from V, of terminals
3. R is a finite set of rules of the form $A \rightarrow w$ where A belongs to V, and w is a finite string composed of elements from V and Σ. We refer to A as the left-hand side (LHS) of the production and w as the right-hand side (RHS), or expansion, of the production. Additionally, each production r has an associated probability $p(r)$ such that the probabilities of productions with the same left-hand side sum to 1.
4. S is the start symbol.

Grammars can either be ambiguous or unambiguous. Ambiguous grammars can generate the same string in multiple ways. Unambiguous grammars cannot.

2.1 Learning Stochastic-Context Free Grammars

Learning context-free grammars is the problem of inducing a context-free structure (or model) from a corpus of sentences (i.e., data). When the grammars are stochastic, we have the additional problem of learning production probabilities (parameters) from the corpus. More formally, given a set of sentence observations $O = \{o_0 \ldots o_{n-1}\}$, the problem is to learn a parameterized model M that best fits the observations.

Typically the problem is framed in terms of search where the objective function is the likelihood of the data given the grammar. One usually starts with a sentence

[1] Stochastic Context-Free Grammars are often called Probabilistic Context-Free Grammars.

corpus consisting of only positive training examples and a grammar that can generate each of the sentences in the corpus. Positive training examples are sentences that provide positive evidence of the structure and parameters in a model whereas negative training examples illustrate what we don't want in our model.

While we have interest in learning the structure of stochastic context-free grammars, our main focus here is on learning parameters. For a thorough overview of techniques for learning structure, see (Stolcke, 1994) and (Chen, 1996).

2.2 Learning Parameters

The Inside-Outside algorithm (Lari & Young, 1990; Lari & Young, 1991) is the standard method for estimating parameters in SCFGs. The algorithm uses the general-purpose expectation-maximization (EM) procedure to iterate between calculating expectations of the parameters and then maximizing the likelihood given both the expectations and the sentence corpus. Almost all parameter learning is done batch style using some derivation of the Inside-Outside algorithm.

For example, in learning parameters, (Chen, 1995) initially estimates the production probabilities using the most probable parse of the sentences given the grammar (the Viterbi parse) and then uses a post-pass procedure that incorporates the Inside-Outside algorithm. He transforms the grammar into Chomsky Normal Form, runs Inside-Outside and then transforms it back.

In using EM to iterate between parameter estimation and maximization, the entire sentence corpus must be stored. While this storage may not be in the form of actual sentences, it is always in some representation that easily allows the reconstruction of the original corpus (e.g., the chart of a chart parse). Because we are interested in language acquisition in embedded agents over large periods of time, the prospect of memorizing entire sentence corpora is unpalatable.

This motivation also carries a desire to easily adjust our parameters when new sentences are encountered. That is, we want to learn production probabilities incrementally. While the Inside-Outside algorithm can be updated with the new sentences during the iterations, it still uses the entire sentence corpus for estimation. Hence, adding a new sentence is no different than simply starting from scratch with the new sentence included in the original corpus.

3 The Algorithm

To address some of the concerns given in the previous section, we have developed SPAN, an unsupervised, incremental algorithm for finding parameter estimates in stochastic context-free grammars.

SPAN is incremental in two ways. First, in the classical sense, at every iteration it uses the previously learned parameter estimation as a stepping stone to the new one. Second, it naturally allows new data to contribute to learning without restarting the entire process.

SPAN uses only a statistical summary of the observation data for learning. It stores the summary information in histograms. SPAN also records histogram information about its current parameter estimates. So the addition of new sentences typically does not increase the memory requirements. Furthermore, the histograms play a crucial role in learning. If the parameter estimates are accurate, the histograms of the observed data should resemeble the histograms of the current parameterizaton. When the histograms do not resemble each other, we use the difference to guide the

Table 1: A grammar in CNF that generates palindromes

S	$\rightarrow$	A	A
S	$\rightarrow$	B	B
S	$\rightarrow$	A	C
S	$\rightarrow$	B	D
C	$\rightarrow$	S	A
D	$\rightarrow$	S	B
A	$\rightarrow$	Y	
B	$\rightarrow$	Z	
Y	$\rightarrow$	y	
Z	$\rightarrow$	z	

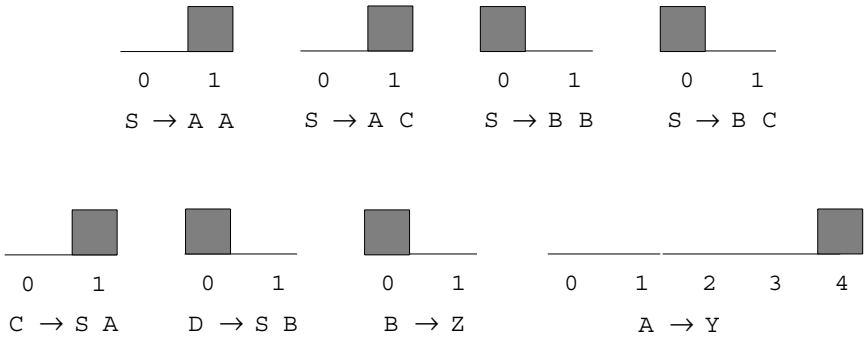

Figure 1: The palindrome grammar rule histograms after only one parse of the sentence $y\ y\ y\ y$. Because only one sentence has been parse, the mass of the distribution is concentrated in a single bin.

learning process.

3.1 Algorithm Description

Suppose we have n sentence observations, $O = \{o_0 \ldots o_{n-1}\}$, generated by a target grammar T and a SCFG, $G = \langle V, \Sigma, R, S \rangle$, in Chomsky Normal Form with random parameter values that meet the constraints given in Section 2. Suppose further that G can generate each observation in our corpus O and that it is identical in structure to T. We call G our learning grammar because we want to learn the parameters of T using only a statistical summary of the observations.

Let each rule in our grammar have two associated histograms called H_r^O and H_r^L. H_r^O is constructed by parsing each sentence in O and recording the number of times rule r appeared in the parse tree of the sentence. Histograms constructed in this way are called observation histograms. The indices of the histogram range from 0 to k where k is the maximum number of times a rule was used in a particular sentence parse. In many cases, k remains small, and more importantly, when a sentence parse does not increase k, the storage requirements remain unchanged.

Each H_r^L is a histogram identical in nature to H_r^O but is used during the learning process, so we call it a learning histogram. Like the observation histograms, SPAN uses each learning histogram to record the number of times each rule occurs in single sentence parses during the learning process. Since SPAN is an iterative algorithm, the learning histograms are repeatedly updated during each iteration. Specifically

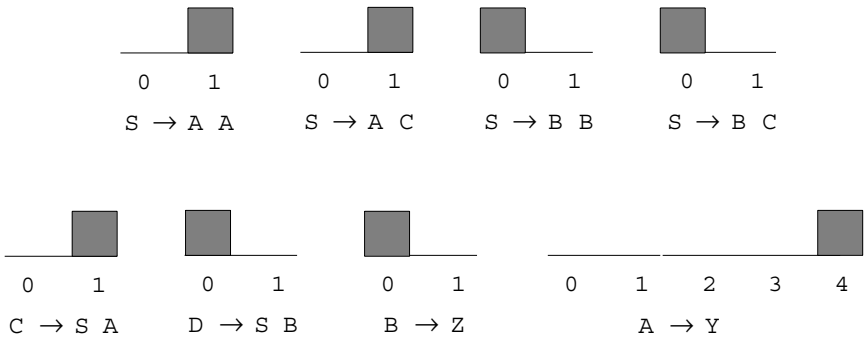

Figure 2: The palindrome grammar rule histograms after parsing $y\ y\ y\ y$ and $y\ y$. Notice that the mass of rules $S \rightarrow A\,C$, and $C \rightarrow S\,A$ are now evenly distributed between 0 and 1. Similarly the mass of rule $S \rightarrow Y$ is evenly distributed between 2 and 4.

they reflect the current parameters of the learning grammar.

Using G and a standard chart parsing algorithm (e.g., (Charniak, 1993) or (Allen, 1995)) we parse each observation o from the sentence corpus. Chart parsing a sentence allows us to count how many times a particular rule was used in deriving that sentence. For example, suppose we have the palindrome-generating grammar given in table 1 and the sentence $y\ y\ y\ y$. Chart parsing the sentence reveals that rule $A \rightarrow Y$ has frequency 4, rules $S \rightarrow A\,A$, $S \rightarrow A\,C$ and $S \rightarrow A$ have frequency 1, and the remaining non-terminals have frequency 0. Figure 1 depicts graphically the histograms for each rule after parsing the sentence. Parsing the sentence $y\,y$, means that rule $S \rightarrow A\,A$ is used once, $A \rightarrow Y$ is used twice and the other rules are not used. Figure 2 shows how the histograms in figure 1 change after additionally parsing $y\,y$.

After every sentence parse, we update the observation histograms and discard the sentence along with its parse. We are left with only a statistical summary of the corpus. As a result, we cannot reconstruct the observation corpus or any single sentence within it. From this point forward the observation histograms are updated only when new data is encountered.

SPAN now begins the iterative process. First, it randomly generates a small sentence corpus of prespecified constant size s from its learning grammar M. Each sentence in the sample is parsed using a chart parser. Using the chart, SPAN records summary statistics exactly as it did for the observation corpus except the statistics for each rule r are added to the learning histograms. After discarding the sentences, we normalize the histograms to some fixed size h. If we don't normalize, the information provided by the new statistics would have decreasing impact on the histograms' distributions since our sample size is constant and the bin counts typically increase linearly. In the future we will examine the role of normalization factor, however, for this work we have kept it fixed throughout the duration of the algorithm.

So, for each rule r we now have two distributions: H_r^O, reflecting the observed corpus, and H_r^L reflecting the overall corpora generated from M. Comparing H_r^L to H_r^O seems a natural predictor of the likelihood of the observation corpus given SPAN's learning grammar. *Relative entropy* (also known as the Kullback-Leibler distance) is commonly used to compare two distributions p and q (Cover & Thomas,

1991). It is defined as:

$$D(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

Specifically, we look at the change in relative entropy from one iteration to the next. If the relative entropy decreases it means SPAN updated the parameters in a helpful manner, if it increases, the distributions have become more dissimilar and so we should try something different. This is best explained by examining SPAN's update rule:

$$P_{t+1}(r) = P_t(r) + \alpha * sgn(P_t(r) - P_{t-1}(r)) * sgn(\Delta RE_t) * f(\Delta RE_t) + \beta * (P_t(r) - P_{t-1})$$

Here, sgn is the "sign" function that returns -1 if its argument is negative, 0 if its argument is 0 and +1 if its argument is positive. The first sign function determines the direction of our previous update. That is, it determines whether we increased the probability or decreased the probability. The second sign function determines if the relative entropy has increased or decreased. If it has decreased, then the difference is positive, if it increased, the difference is negative. Together these sign functions determine the direction of the step. The function $f(\Delta RE_t)$ returns the magnitude of the step. This, intuitively, is an estimation of the gradient since the magnitude of the change in relative entropy is reflective of the slope. The α parameter is a step-size. Finally the $\beta * (P_t(r) - P_{t-1})$ expression is a momentum term. In the next section we explain why the momentum term is needed and other outline some ideas for improving the algorithm.

Once the probability updates are performed we start another iteration beginning with the generation of a small sentence corpus from our learning grammar. The algorithm stops iterating when the relative entropy falls below a threshold, or some prespecified number of iterations has completed.

4 Algorithm Analysis

In this section we show why SPAN is space efficient and also outline some deficiencies of the algorithm and ideas for improving them.

4.0.1 Time and Space Analysis

In this section we analyze both the time and space requirements of SPAN. Comparing the results with the time and space requirements of the Inside-Outside algorithm shows that SPAN is asymptotically equivalent in time but nearly constant (as opposed to linear with Inside-Outside) in space.

The Inside-Outside algorithm runs in $O(\updownarrow^3 |V|^3)$ time where $\updownarrow$ is the length of sentence corpus and $|V|$ is the number of non-terminal symbols (Stolcke, 1994). The complexity arises directly from the chart parsing routines used to estimate probabilities. Note that the number of iterations used by the Inside-Outside algorithm is dominiated by the computational complexity of chart parsing.

SPAN chart parses the observation corpus once but repeatedly chart parses the fixed size samples it generates during the learning process. Taken as a whole, this iterative process typically dominates the single parse of the observations sentences, so the computational complexity is $O(|^3 |V|^3)$ where j is the length of of the maximum sample of any iteration.

Every iteration of the Inside-Outside algorithm requires the complete sentence corpus. So, using the algorithm in the context of embedded agents, where the sentence corpus increases continuously with time, means a corresponding continuous increase in memory. With our algorithm, the memory requirements remain effectively constant.

While SPAN continually updates its learning histograms through the learning process, the number of bins increases only when a sentence parse contains an occurrence count larger than any encountered previously. The sample is representative of the grammar parameters and structure, so typically after a few iterations, the number of bins becomes stable. This means that when new sentences are encountered there is typically corresponding increase in space.

Computing these expectations though, is as difficult as determining the ambiguity of a grammar.

Theorem.

Finding the expected number of occurrences of an expansion in the parse of a language is at least as difficult as determining whether the language is ambiguous.

Proof.

Suppose we can determine the expected number of occurrences of an expansion in the parses of the sentences in the language of a grammar G. Without loss of generality, suppose G is in Chomsky Normal Form. Augment G with two new symbols S' and S'' such that $S'' \rightarrow \varepsilon$ and $S' \rightarrow SS''$ where S is the original start symbol of G.

blah blah blah

4.0.2 Algorithmic Improvements

Talk about how the need to overcome the sample error.

5 Experiments

The previous section described an online algorithm for learning the parameters of SCFGs given summary statistics computed from a corpus of sentences. The question that remains is whether anything is sacrificed in terms of the quality of the learned grammar by using these statistics rather than the complete corpus. This section presents the results of experiments that compare the grammars learned by our algorithm with those learned by the Inside-Outside algorithm.

Let M^T be the target grammar whose parameters are to be learned. Let M^L be a grammar that has the same structure as M^T but whose production probabilities have been assigned random values so as to obey the constraint that the sum of these probabilities for all productions with the same left-hand side must be one. Let O^T be a set of sentences generated stochastically from M^T. The performance of the two algorithms can be compared by running them both on M^L and O^T and computing the log likelihood of O^T given the final grammar.

Because we are interested in learning parameters for a fixed structure, the above procedure involved using a number of different target grammars, each with the same structure but different production probabilities. The goal was to determine whether there were any regions of parameter space for which one algorithm was significantly better. We did this by stochastically sampling from this space. Note

Table 2: A grammar generating simple English phrases

S	$\rightarrow$	NP	VP
NP	$\rightarrow$	Det	N
VP	$\rightarrow$	Vt	NP
VP	$\rightarrow$	Vc	PP
VP	$\rightarrow$	Vi	
PP	$\rightarrow$	P	NP
Det	$\rightarrow$	A	
Det	$\rightarrow$	THE	
Vt	$\rightarrow$	$TOUCHES$	
Vt	$\rightarrow$	$COVERS$	
Vc	$\rightarrow$	IS	
Vi	$\rightarrow$	$ROLLS$	
Vi	$\rightarrow$	$BOUNCES$	
N	$\rightarrow$	$CIRCLE$	
N	$\rightarrow$	$SQUARE$	
N	$\rightarrow$	$TRIANGLE$	
P	$\rightarrow$	$ABOVE$	
P	$\rightarrow$	$BELOW$	
A	$\rightarrow$	a	
THE	$\rightarrow$	the	
$TOUCHES$	$\rightarrow$	$touches$	
IS	$\rightarrow$	is	
$ROLLS$	$\rightarrow$	$rolls$	
$BOUNCES$	$\rightarrow$	$bounces$	
$CIRCLE$	$\rightarrow$	$circle$	
$SQURE$	$\rightarrow$	$square$	
$TRIANGLE$	$\rightarrow$	$triangle$	
$ABOVE$	$\rightarrow$	$above$	
$BELOW$	$\rightarrow$	$below$	

that a new corpus was generated for each new set of parameters as they influence which sentences are generated.

The grammar shown in table 2 (Stolcke, 1994) was used in this manner with 50 different target parameter settings and 500 sentences in O^T for each setting. The mean and standard deviation of the log likelihoods for the online algorithm with $h = s = 100$ (histogram size and learning corpus size respectively) were $\mu = -962.58$ and $\sigma = 241.25$. These values for the Inside-Outside algorithm were $\mu = -959.83$ and $\sigma = 240.85$. Recall that equivalent performance would be a significant accomplishment because the online algorithm has access to much less information about the data. We began by assuming the means of both empirical distributions were equal. With our assumption as the null hypothesis, we ran a two-tailed t-test resulting in $p = 0.95$. This means that if we reject the null hypothesis, the probability of making an error is 0.95.

Unfortunately, the above result does not sanction the conclusion that the two distributions are the same. We can, however, look at the power of the test in this case. If the test's power is high then it is likely that a true difference in the means would be detected. If the power is low then it is unlikely that the test would detect a real difference. The power of a test depends on a number of factors, including the sample size, the standard deviation, the significance level of the test, and the actual difference between the means. Given a sample size of 50, a standard deviation of 240.05, a significance level of 0.05, and an actual delta of 174.79, the power of the

Table 3: An ambiguous grammar

S	$\rightarrow$	A	
S	$\rightarrow$	B	A
A	$\rightarrow$	C	
A	$\rightarrow$	C	C
B	$\rightarrow$	S	
B	$\rightarrow$	D	
B	$\rightarrow$	E	
B	$\rightarrow$	F	
C	$\rightarrow$	z	
D	$\rightarrow$	y	
E	$\rightarrow$	x	
F	$\rightarrow$	w	

t-test is 0.95. That is, with probability 0.95 the t-test will detect a difference in means of at least 174.79 at the given significance level. Because the mean of the two distributions is minute, we need a more powerful test.

Since we ran both algorithms on the same problems, we can also use a paired sample t-test, which is more powerful than the standard t-test. We again set the null hypothesis to our earlier assumption that the means of the two distributions are equal. Running the paired sample t-test yielded $p < 0.01$. That is, the probability of making an error in rejecting the null hypothesis is less than 0.01. Closer inspection of the data reveals why this is the case. Inside-Outside performed better than the online algorithm on each of the 50 grammars. However, as is evident from the means and standard deviation, the absolute difference in each case was quite small. We know that problem variance is minimized when using the paired sample t-test (Cohen, 1995). So, because the unpaired t-test reported no significance and the paired t-test did report significance, we can conclude that the variance in log-likelihood is due almost entirely to problem instance rather than choice of algorithm. So, as a purely practical matter, the performance of the algorithms is virtually identical.

The same experiment was conducted with the ambiguous grammar shown in table 3. The grammar is ambiguous, for example, because $z\ z$ could be generated by $S \rightarrow A \rightarrow C\,C \rightarrow z, z$ or $S \rightarrow BA$ with $B \rightarrow S \rightarrow C \rightarrow z$ and $A \rightarrow C \rightarrow z$. The mean and standard deviation of the log likelihoods for the online algorithm were $\mu = -1983.15$ and $\sigma = 250.95$. These values for the Inside-Outside algorithm were $\mu = -1979.37$ and $\sigma = 250.57$. The standard t-test returned a p value of 0.94 and the paired sample t-test was significant at the 0.01 level. Again, inside-outside performed better on every one of the 50 grammars, but the differences were very small.

6 Discussion and Conclusion

Because our work is motivated by developmental psychology and the unsupervised acquisition of language by embedded agents, we are interested in learning structural models of SCFG's. Remember that one appeal of our algorithm is the ability to add new sentence data naturally and incrementally during the learning process. But if we encounter a new sentence that our grammar cannot parse, it is not clear how we should handle the situation. One idea is to add a completely new set of productions so the sentence can be parsed, and then use structural learning frameworks such

as *model merging* (Stolcke & Omohundro, 1994) or *move selection* (Chen, 1995) to generalize the grammar.

Most parameter learning algorithms for stochastic context-free grammars retain the entire sentence corpus throughout the learning process. Incorporating a complete memory of sentence corpora seems ill-suited for language acquisition in embedded agents. We have introduced an incremental algorithm for learning parameters in stochastic context-free grammars using only summary statistics of the observed data. Empirical results demonstrate that the algorithm performs as well as the Inside-Outside algorithm despite using only a statistical summary of the data.

Acknowledgements

This research is supported by DARPA/USASMDC under contract number DASG60-99-C-0074. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of DARPA/USASMDC or the U.S. Government.

We would like to acknowledge Mark Johnson for the use of his Inside-Outside algorithm implementation.

References

Allen, J. (1995). *Natural language understanding.* The Benjamins/Cummings Publishing Company, Inc. 2 edition.

Charniak, E. (1993). *Statistical language learning.* Cambridge: The MIT Press.

Chen, S. F. (1995). *Bayesian grammar induction for language modeling* (Technical Report TR-01-95). Center for Research in Computing Technology, Harvard University, Cambridge, MA.

Chen, S. F. (1996). *Building probabilistic models for natural language.* Doctoral dissertation, The Division of Applied Sciences, Harvard University.

Cohen, P. R. (1995). *Empirical methods for artificial intelligence.* The MIT Press.

Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory.* John Wiley and Sons, Inc.

Lari, K., & Young, S. J. (1990). The estimation of stochastic context-free grammars using the inside-outiside algorithm. *Computer Speech and Language, 4,* 35–56.

Lari, K., & Young, S. J. (1991). Applications of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language, 5,* 237–257.

Oates, T. (2001). *Grounding knowledge in sensors: Unsupervised learning for language and planning.* Doctoral dissertation, The University of Massachusetts, Amherst.

Sipser, M. (1997). *Introduction to the theory of computation.* Boston: PWS Publishing Company.

Stolcke, A. (1994). *Bayesian learning of probabilistic language models.* Doctoral dissertation, Division of Computer Science, University of California, Berkeley.

Stolcke, A., & Omohundro, S. (1994). Inducing probabilistic grammars by bayesian model merging. *Grammatical Inference and Applications* (pp. 106–118). Berlin, Heidelberg: Springer.