Learning Stochastic Categorial Grammars

Miles Osborne and Ted Briscoe
Computer Laboratory, Cambridge University
Cambridge CB2 3QG, UK
{Miles.Osborne,Ted.Briscoe}@cl.cam.ac.uk

Abstract

Stochastic categorial grammars (SCGs) are introduced as a more appropriate formalism for statistical language learners to estimate than stochastic context free grammars. As a vehicle for demonstrating SCG estimation, we show, in terms of crossing rates and in coverage, that when training material is limited, SCG estimation using the Minimum Description Length Principle is preferable to SCG estimation using an indifferent prior.

1 Introduction

Stochastic context free grammars (SCFGs), which are standard context free grammars extended with a probabilistic interpretation of the generation of strings, have been shown to model some sources with hidden branching processes more efficiently than stochastic regular grammars (Lari and Young, 1990). Furthermore, SCFGs can be automatically estimated using the Inside-Outside algorithm, which is guaranteed to produce a SCFG that is (locally) optimal (Baker, 1990). Hence, SCFGs appear to be suitable formalisms for the estimation of wide-covering grammars, capable of being used as part of a system that assigns logical forms to sentences.

Unfortunately, from a Natural Language Processing perspective, SCFGs are not appropriate grammars to learn. Firstly, as Collins demonstrates (Collins, 1996), accurate parse selection, which is important for ambiguity resolution, requires lexical statistics. SCFGs, as standardly used in the Inside-Outside algorithm, are in Chomsky Normal Form (CNF), which restricts rules to being at most binary branching. Such rules are not lexicalised, and hence, to lexicalise (CNF) CFGs requires adding a complex statistical model that simulates the projection of head items up the parse tree. Given the embryonic status of grammatical statistical models and the difficulties of accurately estimating the parameters of such a model, it seems more prudent to prefer whenever possible simpler statistical models with fewer parameters, and treat lexicalisation as part of the grammatical formalism, and not as part of the statistical framework (for example (Schabes, 1992)). Secondly, (stochastic) CFGs are well-known as being linguistically inadequate formalisms for problems such as non-constituent coordination. Hence, a learner using a SCFG will not have an appropriate formalism with which to construct an adequate grammar.

Stochastic categorial grammars (SCGs), which are classical categorial grammars extended with a probabilistic component, by contrast, have a grammatical component that is naturally lexicalised. Furthermore, Combinatory Categorial Grammars have been shown to account elegantly for problematic areas of syntax such as non-constituent co-ordination (Steedman, 1989), and so it seems likely that SCGs, when suitably extended, will be able to inherit this linguistic adequacy. We therefore believe that SCGs are more useful formalisms for statistical language learning than SCFGs. Future work will reinforce the differences between SCFGs and SCGS, but in this paper, we instead concentrate upon the estimation of SCGs.

Stochastic grammars (of all varieties) are usually estimated using the *Maximum Likelihood Principle*, which assumes an indifferent prior probability distribution. When there is sufficient training material, *Maximum Likelihood Estimation* (MLE) produces good results. More usually however, with many thousands of parameters to estimate, there will be insufficient training material for MLE to produce an optimal solution. If, instead, an *informative* prior is used in place of the indifferent prior, better results can be achieved. In this paper we show how using an informative prior probability distribution

Miles Osborne and Ted Briscoe (1997) **Learning Stochastic Categorial Grammars.** In T.M. Ellison (ed.) *CoNLL97: Computational Natural Language Learning,* ACL pp 80-87.

leads to the estimation of a SCG that is more accurate than a SCG estimated using an indifferent prior. We use the *Minimum Description Length Principle* (MDL) as the basis of our informative prior. To our knowledge, we know of no other papers comparing MDL to MLE using naturally occurring data and learning probabilistic grammars. For example, Stolcke's MDL-based learner was trained using artificial data (Stolcke, 1984); Chen's similar learner mixes smoothing techniques with MDL, thereby obfuscating the difference between MDL and MLE (Chen, 1996).

The structure of the rest of this paper is as follows. In section 2 we introduce SCGs. We then in section 3 present a problem facing most statistical learners known as *overfitting*. Section 4 gives an overview of the MDL principle, which we use to deal with overfitting[1]; in section 5 we present our learner. Following this, in section 6 we give some experiments comparing use of MDL, with a MLE-style learner. The paper ends with some brief comments.

2 Grammar formalism and statistical models

An SCG is a classical categorial grammar (one using just functional application, see, for example, Wood (Wood, 1993)) such that each category is augmented with a probability, which is used to model the choices made when constructing a parse. Categories are conditioned on the lexical item they are assigned to.

More formally, a categorial lexicon G is the tuple $\langle A, C, V, L \rangle$, where:

- A is a non-empty set of atomic categories.
- C is a non-empty set of complex categories.
- V is a non-empty set of lexical items (words).
- L is the function $L\colon \forall v \in V \mapsto 2^C$. That is, L assigns sets of categories to lexical items.

Complex categories are defined as follows:

- Any member of A is a complex category.
- If a and b are complex categories, then so is $a\backslash b$.
- If a and b are complex categories, then so is a/b.
- Nothing else is a complex category.

[1]A fuller discussion of MDL and statistical language learning can be found in (Rissanen and Ristad, 1994, de Marcken, 1996).

A categorial grammar consists of a categorial lexicon augmented with the rule of left functional application: $a \ b\backslash a \mapsto b$ and the rule of right functional application: $b/a \ a \mapsto b$.

A probabilistic categorial grammar is a categorial grammar such that the sum of the probabilities of all derivations is one. Since in our variant of a categorial grammar, where there are no variables in categories, directional information is encoded into each category, and we only use functional application, the actual derivation of any sentence mechanically follows from the assignment of categories to lexical items, and so it follows that the choices available when parsing with a categorial grammar arise from the particular assignment of categories to any given lexical item. Within a stochastic process, probabilities model these choices, so in a stochastic categorial grammar, we need to ensure that the probabilities of all categories assigned to a particular lexical item sum to one. That is, for all categories c in lexicon C assigned to lexical item w:

$$\sum_{c \in C} P(c \mid w) = 1 \tag{1}$$

We estimate $P(c \mid w)$ as being:

$$P(c \mid w) \approx \frac{f(c)}{\sum_{x \in C} f(x)} \tag{2}$$

for some distinct category c, occurring with frequency $f(c)$, that can be assigned to lexical item w, and for all categories x, with frequency $f(x)$, that can also be assigned to w.

For the derivation space to actually sum to one, all possible assignments of categories to lexical items must be legal. Clearly, only assignments of lexical items that combine to form a valid parse constitute legal category assignments, and so there will be a probability loss. That is, the sum of all derivations will be less than, or equal to one. We can either scale the probabilities so that the derivations do sum to one, or alternatively, we can assume that (illegal) assignments of categories are never seen, with the relative probabilities between the legal category assignments being unaffected, and so give a zero probability to the illegal category assignments [2].

Because categories are normalised with respect to the lexical item they are associated with, the resulting statistical model is lexicalised. However, in this paper, we learn lexica of *part-of-speech tag sequences*, and not lexica for actual words. That is, the set of simple categories is taken as being a part-of-speech tag set and the set of words is also a set of

[2]Thanks to Eirik Hektoen for pointing this point out.

part-of-speech tags. This greatly reduces the number of parameters to be acquired, than would be the case if the lexicon contained a set of words, but incurs the obvious cost of a loss of accuracy. In future experiments, we plan to learn fully-lexicalised SCGs.

Having now introduced SCGs, we now turn to the problem of *overfitting*.

3 Overfitting

Bayesian inference forms the basis of many popular language learning systems, examples of which include the Baum-Welch algorithm for estimating hidden Markov models (Baum, 1972) and the Inside-Outside algorithm for estimating CFGs (Baker, 1990). As is well known, Bayes' theorem takes the following form:

$$P(H \mid D) = \frac{P(H)P(D \mid H)}{P(D)} \tag{3}$$

Here, the term $P(H)$ is the *prior probability*, $P(D \mid H)$ is the *likelihood probability*, and $P(H \mid D)$ is the *posterior probability*. The prior probability of H can be interpreted as quantifying one's belief in H. If the prior is accurate, hypotheses that are closer to the target hypothesis will have a higher prior probability assigned to them than hypotheses that are further away from the target hypothesis. The likelihood probability describes how well the training material can be encoded in the hypothesis. For example, one would hope that the training corpus would receive a high likelihood probability, but a set of ungrammatical sentences would receive a low likelihood probability. Finally, the posterior probability can be considered to be the combination of these two probability distributions: we prefer hypotheses that accord with our prior belief in them (have a high prior probability) and model the training material well (have a high likelihood probability). When learning in a Bayesian framework, we try to find some hypothesis that maximises the posterior probability. For example, we might try to find some maximally probable grammar H given some corpus D.

The usual setting is for the learner to assume an *uninformative* (indifferent) prior, yielding MLE. Usually, with sufficient data, MLE give good results. However, with insufficient data, which is the standard case when there are many thousands of parameters to estimate, MLE, unless checked, will lead to the estimation of a large theory whose probability mass is concentrated upon the training set, with a consequential poor prediction of future, unseen events. This problem is known as *over-fitting*. Over-fitting affects all Bayesian learners that assume an uninformative prior and are given insufficient training data. An over-fitted theory poorly predicts future events not seen in the training set. Clearly, good prediction of unseen events is the central task of language learners, and so steps need to be taken to avoid over-fitting.

Over-fitting is generally tackled in two ways:

- *Restrict* the learner such that it cannot express the maximally likely hypothesis, given some hypothesis language.
- *Smooth* the resulting parameters in the hope that they back-off from the training data and apportion more of the probability mass to account for unseen material.

Examples of the first approach can be seen most clearly with the usage of CNF grammars by the Inside-Outside algorithm (Pereira and Schabes, 1992, Lari and Young, 1990). A grammar in CNF does not contain rules of an arbitrary arity, and so when learning CNF grammars, the Inside-Outside algorithm cannot find the maximal likelihood estimation of some training set. The problem with this language restriction is that there is no a priori reason why one should settle with any particular limit on rule arity; some grammars mainly contain binary rules, but others (for example those implicitly within tree-banks) sometimes contain rules with many right-hand side categories. Any language restriction, in lieu of some theory of rule arity, must remain ad hoc. Note that SCGs, whilst assigning binary branching trees to sentences, contain categories that may naturally be of an arbitrary length, without violating linguistic intuitions about what constitutes a plausible analysis of some sentence.

Examples of the second approach can be found in language modelling (for example (Church and Gale, 1991, Katz, 1987)). Smoothing a probability distribution tends to make it 'closer' (reduces the Kullback-Liebler distance) to some other probability distribution (for example, the uniform distribution). Unfortunately, there is no guarantee that this other distribution is closer to the target probability distribution than was the original, un-smoothed distribution, and so smoothing cannot be relied upon always to improve upon the un-smoothed theory. Smoothing is also a post-hoc operation, unmotivated by details of what is actually being learnt, or with properties (problems) of the estimation process. Instead of selecting some language restriction or resorting to smoothing, a better solution to the over-fitting problem would be to use an *informative* prior. One such prior is in terms of *theory minimisation*, the pursuit of which leads to the *Minimum Description Length Principle* (MDL) (Rissanen, 1989).

In this paper we demonstrate that using MDL gives better results than when using an uninformative prior. Elsewhere, we demonstrated that (Good-Turing) smoothing does improve upon the accuracy of a SCG estimated using MLE, but still, the best results were obtained when using MDL (Osborne, 1997).

4 The MDL Principle

Learning can be viewed as *compression* of the training data in terms of a *compact hypothesis*. It can be shown that, under very general assumptions, the hypothesis with the minimal, or nearly minimal complexity, which is consistent with the training data, will with high probability predict future observations well (Blumer et al., 1987). One way of finding a good hypothesis is to use a prior that favours hypotheses that are consistent with the training data, but have minimal complexity. That is, the prior should be construed in terms of how well the hypothesis can be compressed (since significant compression is equivalent to a low stochastic complexity).

We can compress the hypothesis by replacing it with code words, such that when measured in bits of information, the total length of the encoding is less than, or equal to, the length of the hypothesis, also when measured in bits. To achieve this aim, objects in the hypothesis that occur frequently should be assigned shorter length code words than objects that occur infrequently. Let $l(H)$ be the total length of the code words for some set of objects H, as assigned by some optimal coding scheme. It turns out that:

$$2^{-l(H)} \tag{4}$$

can be used as a prior probability for H. The smaller $l(H)$, the greater the compression, and so the higher the prior probability.

There is an equivalence between description lengths, as measured in bits, and probabilities: the Shannon Complexity of some object x, with probability $P(x)$, is $-\log(P(x))$ (all logarithms are to the base 2). This gives the minimal number of bits required to encode some object. Hence, we can give a description length to both the prior and likelihood probabilities. Using these description lengths, we have the MDL Principle: we should select some hypothesis H that:

- Minimises the length of the hypothesis (when measured in bits) and
- Minimises the length of the data encoded in the hypothesis (measured in bits).

The first part says prefer hypotheses that are compact; the second part says prefer hypotheses that fit the data well. Both aspects of a theory are taken into consideration to arrive at a proper balance between overly favouring a compact hypothesis (which will model the training data badly) and overly favouring the likelihood probability (which leads to overfitting).

To use the MDL principle when learning grammar, we need to compute the prior and likelihood probabilities. One way to compute the prior is as follows.

We give each category r in lexicon H an *encoding* probability $P(r)$. If r was used $f(r)$ times in the parse trees of the training set,

$$P(r) = \frac{f(r)}{\sum_{x \in H} f(x)} \tag{5}$$

That is, categories used frequently in the training set have a high probability, and categories used infrequently have a low probability.

The intuition behind this particular coding scheme is to imagine that we are transmitting, in the shortest possible way, a set of parse trees across some channel. We conceptually use a two-part, dictionary-based coding scheme: one part for word-category pairs with their associated code words, and another part for an encoding of the trees in terms of the code words. Since the total length of the encoding of the trees will be much larger than the total length of the word-category pairs and associated code words, we can assume the dictionary length is just a constant, smaller than the total length of the encoded parse trees, and just consider, without an undue loss in accuracy, the cost of transmitting the trees. Hence, when we evaluate various lexica, we determine how much it costs to transmit the training material in terms of the particular dictionary-based encoding of the lexicon in question. Equation 5 is used to give the length, in bits, of the code word we would assign to each category in a parse tree.

Our encoding scheme treats each category as being independent and clearly, we could have used more of the context within the parse trees to construct a more efficient encoding scheme (see, for example (Ristad and Thomas, 1995)). For the purposes of this paper, our simple encoding scheme is sufficient.

The length of a lexicon is the sum of the lengths of all the categories used in the grammar:

$$l(H) = \sum_{x \in H} -\log(P(x)) \tag{6}$$

The prior is therefore:

$$P(H) = 2^{-l(H)} \tag{7}$$

The likelihood probability, $P(D \mid H)$, is defined as simply the product of the probabilities of the categories used to parse the corpus.

We approximate the probability of the data, $P(D)$, using a linear interpolated trigram model (Jelinek, 1990). Our trigram model is used to assign probabilities to substrings: substrings denoting phrases will be assigned higher probabilities than substrings that do not form natural phrases. It should be pointed out that most work in statistical language learning ignores $P(D)$. However, the implementation reported in this paper is greedy, and tries to build parse trees for sentences incrementally. Hence, we need to determine if the substring dominated by a local tree forms a phrase (has a high $P(D)$), and is not some non-phrasal word grouping (has a low $P(D)$). Clearly, using trigrams as an approximation of $P(D)$ may undermine the estimation process. In our more recent, non-greedy work, we can, and do, ignore $P(D)$, and so do not resort to using the trigram model.

5 Implementation

Having shown how MDL can be applied to the estimation of SCG, we now turn to a description of an implemented system. We learn categorial grammars in a greedy, bottom-up, incremental manner. In summary:

- For each part-of-speech tag sequence in some corpus, we create a labelled binary tree spanning that sequence.
- We then read-off from the tree those categories that would have generated that tree in the first place, placing them in the lexicon for subsequent usage.

In more detail, to create a labelled binary tree, we firstly assign unary trees to each tag in the tag sequence. As far as the current implementation is concerned, the only element in a unary local tree is the tag. For example, assuming the following tagged sentence:

`We_prp love_vbp categorial_jj grammars_nns`

we would generate the forest of local trees:

`(prp) (vbp) (jj) (nns)`

We ignore words and only work with the part-of-speech tags.

Next, we consider all pairwise ways of joining adjacent local trees together. For example, given the previous forest of local trees, we would consider joining the following local trees together:

`(prp) (vbp)`

`(vbp) (jj)`

and

`(jj) (nns)`

Each putative local tree is evaluated using Bayes' theorem: the prior is taken as being the probability assigned to an encoding of just the categories contained within the local tree (with respect to all the categories in the lexicon)[3]; the likelihood is taken as being the geometric mean of the probabilities of the categories contained within the local tree[4]; the probability of the data is taken as being the probability assigned by the ngram model to the tag sequence dominated by that local tree. The mother of a local tree is defined using a small table of what constitutes a mother given possible heads. Mothers are always either the left or right daughter, representing either left or right functional application.

After evaluating each putative local tree, the tree with the highest posterior probability is chosen. This tree replaces the two local trees from which it was created.

Continuing our example, if we assume the putative local tree:

`(nns (jj) (nns))`

has a higher posterior probability than the putative local tree:

`(vbp (vbp) (jj))`

we would replace the local trees:

`(jj) (nns)`

with the local tree:

`(nns (jj) (nns))`

The whole process of tree evaluation, selection and replacement is then repeated until a single tree remains.

To read categories off a labelled local tree, the following recursive process is applied:

- The category of the root of a tree is the category dominating that tree.
- Given a local tree of the form (A (A B)), the category assigned to the daughter node labelled A is α/B, where α is the category assigned to the root of the tree. The category assigned to node B is B.

[3] This differs from taking the length of all the categories in the lexicon. We do this for efficiency purposes.

[4] We take the geometric mean, and not the product, as this normalises the likelihood probability of arbitrary numbers of categories.

- Given a local tree of the form (A (B A)), the category assigned to the daughter node labelled A is $\alpha\backslash$B, where α is the category assigned to the root of the tree. The category assigned to B is B.

Note other methods of reading categories off a tree might exist. We make no claim that this is necessarily the best method.

So, if we assume the following tree:

`(vbp (prp) (vbp (vbp) (nns (jj) (nns))))`

we would extract the following categories:

Tag	Category
nns	nns\jj
jj	jj
vbp	vbp\prp/nns
prp	prp

Our categories are With each category, we also keep a frequency count of the number of times that category was added to the lexicon. This frequency information is used to estimate the probabilities of the lexicon.

Finally, when learning, we ignore sentences shorter than three words (these are likely to be ungrammatical fragments), or, for computational reasons, sentences longer than 50 words.

6 Experiments

Here, we report on a number of experiments showing that when there is a danger of overfitting taking place, MDL produces a quantitatively better SCG than does MLE.

To evaluate the various lexica produced, we used the following metrics:

- To measure a grammar's coverage, we note the number of tag sequences, drawn from a corpus of naturally occurring language, some grammar generates. The higher the number, the better the grammar.

- To measure a grammar's overgeneration, we note the number of ungrammatical strings, drawn from a source that generates all strings up to some length randomly, a grammar generates. The lower the number, the better the grammar. That is, random sequences of tags, of a sufficient length, will have a low probability of being grammatically well-formed.

- To measure the accuracy of the parses produced, we use the Grammar Evaluation Interest Group scheme (GEIG) (Harrison et al., 19). This compares unlabelled, manually produced parses with automatically produced parses in terms of *recall* (the ratio of matched brackets over all brackets in the manually produced parses), *precision* (the ratio of matched brackets in the manually produced parse over all brackets found by the parser) and *crossing rates* (the number of times a bracketed sequence produced by the parser overlaps with one in the manually produced parse, but neither is properly contained in the other). The higher the precision and recall, and the lower the crossing rates, the better the grammar.

Throughout our experiments, we used the Brill part-of-speech tagger to create testing and training material (Brill, 1993). Our trigram model was created using seven million words of tagged material drawn from the British National Corpus (BNC); training material consisted of 43,000 tagged sentences also taken from the BNC. For test material, we took 429 sentences taken from the Spoken English Corpus (SEC). To compute crossing rates, recall and precision figures, we used a program called Parseval to compare most probable parses with manually produced parses (232 trees in total taken from the SEC) (Harrison et al., 19). To measure overgeneration, we randomly generated 250 strings. From a manual inspection, these do appear to be ungrammatical. Here is an example randomly generated tag sequence:

1 *NP MD WP$ LS POS POS VBD NN WDT SYM*

We started with no initial lexica.

Training constructed the lexica outlined in figure 1. Note the difference in the size of the lexica. All things being equal, we prefer the lexicon to be as small as possible. The larger the lexicon, the slower the parsing. As predicted by theory, the lexicon learnt using MLE is larger than the one learnt using MDL.

Lexicon	How learnt	Size (categories)
A	MDL	24829
B	MLE	31091

Figure 1: Sizes of various lexica

Testing for coverage, we produced the results shown in figure 2. Again as predicted, lexicon A is

Lexicon	Percentage generated
A	95
B	93

Figure 2: Undergeneration

closer to convergence (better coverage) than lexicon B.

Lexicon	Percentage generated
A	0
B	0

Figure 3: Overgeneration

Turning now to figure 3, we see that, with respect to the test set, neither lexicon overgenerates.

Lexicon	Crossing rate	Precision	Recall
A	2.84	51.13	36.04
B	3.39	46.46	32.05

Figure 4: Crossing rates

Figure 4 shows the crossing rate results. Again, MDL has lead to the estimation of a better lexicon than has MLE. Note that the actual figures are not as great as they might be. This follows from the fact that although categorial grammars assigned binary-branching trees to sentences, the test parses used to compute crossing rates were not restricted to being binary branching. Also, our learner used virtually no supervision (for example parsed corpora), and did not start with a given lexicon: learning using parsed corpora is substantially easier than learning from just a tagged text, whilst starting with a given, manually constructed lexicon is equivalent to learning with a good initial estimation of the target lexicon, which greatly increases the chance of successful learning. However, the figures are sufficient for the purposes of our demonstration.

7 Discussion

In this paper, we introduced SCGs, and argued that they are more appropriate formalisms with which to estimate grammars than are SCFGs. We then showed how the Minimum Description Length Principle provides a way of reducing the problem of overfitting when estimating SCGs.

In more recent work, we are using a version of the Expectation-Maximisation algorithm to estimate SCGs. We use bracketed training material, and an MDL-style prior to aid in the estimation process. A later publication will report on this research.

The current state-of-the-art parsers trained on treebank data using fully lexicalised probabilistic models achieve crossing rates of around 1.0 per sentence. We achieve 2.84 using more heterogeneous and unannotated training material and learning SCGs from scratch. In future work we will attempt to rival the state-of-the-art through full lexicalisation and utilising bracketed training material.

Acknowledgments

We would like to thank the two anonymous reviewers for valuable comments on this paper. This work was supported by the EU Project *Sparkle* LE-21

References

J. K. Baker. Trainable grammars for speech recognition. In D. H. Klatt and J. J. Wolf, editors, *Speech Communication Papers for the 97th Meeting of the Acoustical Society of America*, pages 547–550. 1979.

L. E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. *Inequalities*, III:1–8, 1972.

A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Occam's Razor. *Information Processing Letters*, 24:377–380, 1987.

Eric Brill. *A Corpus-Based Approach to Language Learning.* PhD thesis, University of Pennsylvania, 1993.

Stanley F. Chen. *Building Probabilistic Language Models for Natural Language.* PhD thesis, Harvard University, 1996.

Kenneth W. Church and William A. Gale. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, 5:19–54, 1991.

Michael John Collins. A new statistical parser based on bigram lexical dependencies. In 34th *Annual Meeting of the Association for Computational Linguistics.* University of California, Santa Cruz, California, USA, June 1996.

Carl de Marcken. *Unsupervised Language Acquisition.* PhD thesis, MIT, 1996.

Philip Harrison, Steven Abney, Ezra Black, Dan Flickinger, Ralph Grishman Claudia Gdaniec, Donald Hindle, Robert Ingria, Mitch Marcus, Beatrice Santorini, and Tomek Strzalkowski. Evaluating Syntax Performance of Parser/Grammars of English. In Jeannette G. Neal and Sharon M. Walter, editors, *Natural Language Processing Systems Evaluation Workshop, Technical Report RL-TR-91-362*, 1991.

Fred Jelinek. Self-organised language modelling for speech recognition. In Alex Waibel and Kai-Fu Lee, editors, *Readings in Speech Recognition*, pages 450–560. Morgan-Kaufmann, 1990.

Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. In *IEEE Transactions on Acoustics, Speech and Signal Processing*, volume 35, pages 400–401, March 1987.

K. Lari and S. J. Young. The estimation of stochastic context-free grammars using the Inside-Outside Algorithm. *Computer Speech and Language*, 4:35–56, 1990.

Miles Osborne. Minimisation, indifference and statistical language learning. In *Empirical Learning of Natural Language Processing Tasks*, pages 113–124, Prague, Czech Republic, April 1997. ECML-97 MLNET Familiarisation Workshop.

Fernando Pereira and Yves Schabes. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the* 30^{th} *ACL, University of Delaware, Newark, Delaware*, pages 128–135, 1992.

Jorma Rissanen. *Stochastic Complexity in Statistical Inquiry.* Series in Computer Science -Volume 15. World Scientific, 1989.

Jorma Rissanen and Eric Sven Ristad. Language Acquisition in the MDL Framework. In Eric Sven Ristad, editor, *Language Computation.* American Mathemtatical Society, Philedelphia, 1994.

Eric Sven Ristad and Robert G. Thomas, III. Context Models in the MDL Framework. In *Proceedings of the Data Compression Conference*, pages 62–71, Snowbird, Utah, March 1995. IEEE, IEEE Computer Society Press.

Yves Schabes. Stochastic lexicalized tree-adjoining grammars. In *Proceedings of the* 14^{th} *International Conference on Computational Linguistics*, 1992.

M. J. Steedman. Dependency and Coordination in the Grammar of Dutch and English. *Language*, 61:523–568, 1989.

Andreas Stolcke. *Bayesian Learning of Probabilistic Language Models.* PhD thesis, University of California, Berkley, 1994.

Mary McGee Wood. *Categorial Grammars.* Routledge, 1993. Linguistic Theory Guides.