Unsupervised Lexical Learning with Categorial Grammars

Stephen Watkinson and Suresh Manandhar,
Department of Computer Science,
University of York,
York YO10 5DD,
UK.

Abstract

In this paper we report on an unsupervised approach to learning Categorial Grammar (CG) lexicons. The learner is provided with a set of possible lexical CG categories, the forward and backward application rules of CG and unmarked positive only corpora. Using the categories and rules, the sentences from the corpus are probabilistically parsed. The parses and the history of previously parsed sentences are used to build a lexicon and annotate the corpus. We report the results from experiments on a number of small generated corpora, that contain examples from subsets of the English language. These show that the system is able to generate reasonable lexicons and provide accurately parsed corpora in the process. We also discuss ways in which the approach can be scaled up to deal with larger and more diverse corpora.

1 Introduction

In this paper we discuss a potential solution to two problems in Natural Language Processing (NLP), using a combination of statistical and symbolic machine learning techniques. The first problem is learning the syntactic roles, or categories, of words of a language *i.e.* learning a lexicon. Secondly, we discuss a method of annotating a corpus with parses.

The aim is to learn Categorial Grammar (CG) lexicons, starting from a set of lexical categories, the functional application rules of CG and an unannotated corpus of positive examples. The CG formalism (discussed in Section 2) is chosen because it assigns distinct categories to words of different types, and the categories describe the exact syntactic role each word can play in a sentence.

This problem is similar to the unsupervised part of speech tagging work of, for example, Brill (Brill, 1997) and Kupiec (Kupiec, 1992). In Brill's work a lexicon containing the parts of speech available to each word is provided and a simple tagger attaches a complex tag to each word in the corpus, which represents all the possible tags that word can have. Transformation rules are then learned which use the context of a word to determine which simple tag it should be assigned. The results are good, generally achieving around 95% accuracy on large corpora such as the Penn Treebank.

Kupiec (Kupiec, 1992) uses an unsupervised version of the Baum-Welch algorithm, which is a way of using examples to iteratively estimate the probabilities of a Hidden Markov Model for part of speech tagging. Instead of supplying a lexicon, he places the words in equivalence classes. Words in the same equivalence class must take one of a specific set of parts of speech. This improves the accuracy of this algorithm to about the same level as Brill's approach.

In both cases, the learner is provided with a large amount of background knowledge – either a *complete* lexicon or set of equivalence classes. In the approach presented here, the most that is provided is a *small* partial lexicon. In fact the system learns the lexicon.

The second problem – annotating the corpus – is solved because of the approach we use to learn the lexicon. The system uses parsing to determine which are the correct lexical entries for a word, thus annotating the corpus with the parse derivations (also providing less probable parses if desired). An example of another approach to doing this is the Fidditch parser of Hindle (Hindle, 1983) (based on the deterministic parser of Marcus (Marcus, 1980)), which was used to annotate the Penn Treebank (Marcus et al., 1993). However, instead of learning the lexicon, a complete grammar and lexicon must be supplied to the Fidditch parser.

Our work also relates to CG induction, which has been attempted by a number of people. Osborne (Osborne, 1997) has an algorithm that learns a grammar for sequences of *part-of-speech tags* from a *tagged* corpora, using the Minimum Description Length (MDL) principle – a well-defined form of compression. While this is a supervised setting of the problem, the use of the more formal approach to compression is of interest for future work. Also, results of 97% coverage are impressive, even though the problem is rather simpler. Kanazawa (Kanazawa, 1994) and Buszkowski (Buszkowski, 1987) use a unification based approach with a corpus annotated with semantic structure, which in CG is a strong indicator of the syntactic structure. Unfortunately, they do not present results of experiments on natural language corpora and again the approach is essentially supervised.

Two unsupervised approaches to learning CGs are presented by Adriaans (Adriaans, 1992) and Solomon (Solomon, 1991). Adriaans, describes a purely symbolic method that uses the context of words to define their category. An oracle is required for the learner to test its hypotheses, thus providing negative evidence. This would seem to be awkward from a engineering view point *i.e.* how one could provide an oracle to achieve this, and implausible from a psychological point of view, as humans do not seem to receive such evidence (Pinker, 1990). Unfortunately, again no results on natural language corpora seem to be available.

Solomon's approach (Solomon, 1991) uses unannotated corpora, to build lexicons for simple CG. He uses a simple corpora of sentences from children's books, with a slightly *ad hoc* and non-incremental, heuristic approach to developing categories for words. The results show that a wide range of categories can be learned, but the current algorithm, as the author admits, is probably too naive to scale up to working on full corpora. No results on the coverage of the CGs learned are provided.

In Section 3 we discuss our learner. In Section 4 we describe experiments on three corpora containing examples of a subset of English and Section 5 contains the results, which are encouraging with respect to both problems. Finally, in Section 6, we compare the results with the systems mentioned above and discuss ways the system can be expanded and larger scale experiments may be carried out. Next, however, we describe Categorial Grammar.

2 Categorial Grammar

Categorial Grammar (CG) (Wood, 1993; Steedman, 1993) provides a functional approach to lexicalised grammar, and so, can be thought of as defining a syntactic *calculus*. Below we describe the basic (AB) CG, although in future it will be necessary to pursue a more flexible version of the formalism.

There is a set of *atomic* categories in CG, which are usually nouns (n), noun phrases (np) and sentences (s). It is then possible to build up *complex* categories using the two slash operators "/" and "\". If A and B are categories then A/B is a category and A\B is a category. With basic CG there are just two rules for combining categories: the forward (FA) and backward (BA) *functional application* rules. Following Steedman's notation (Steedman, 1993) these are:

$$\begin{array}{lcll} X/Y\ Y & \Rightarrow & X & \qquad (FA) \\ Y\ X\backslash Y & \Rightarrow & X & \qquad (BA) \end{array}$$

Therefore, for an intransitive verb like "run" the complex category is s\np and for a transitive verb like "take" it is (s\np)/np. In Figure 1 the parse derivation for "John ate the apple" is presented.

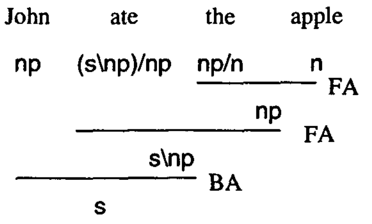

Figure 1: A Example Parse in Pure CG

The CG described above has been shown to be weakly equivalent to context-free phrase structure grammars (Bar-Hillel et al., 1964). While such expressive power covers a large amount of natural language structure, it has been suggested that a more flexible and expressive formalism may capture natural language more accurately (Wood, 1993; Steedman, 1993).

This has led to some distinct branches of research into usefully extending CG, which will be investigated in the future.

CG has at least the following advantages for our task.

- Learning the lexicon and the grammar is one task.
- The syntax directly corresponds to the semantics.

The first of these is vital for the work presented here. Because the syntactic structure is defined by the complex categories assigned to the words, it is not necessary to have separate learning procedures for the lexicon and for the grammar rules. Instead, it is just one procedure for learning the lexical assignments to words.

Secondly, the syntactic structure in CG parallels the semantic structure, which allows an elegant interaction between the two. While this feature of CG is not used in the current system, it could be used in the future to add semantic background knowledge to aid the learner (*e.g.* Buszkowski's discovery procedures (Buszkowski, 1987)).

3 The Learner

The system we have developed for learning lexicons and assigning parses to unannotated sentences is shown diagrammatically in Figure 2. In the following sections we explain the learning setting and the learning procedure respectively.

3.1 The Learning Setting

The input to the learning setting has five parts: the corpus, the lexicon, the CG rules, the set of legal categories and a probabilistic parser, which are discussed below.

The Corpus The corpus is a set of unannotated positive examples represented in Prolog as facts containing a list of words *e.g.*

```
ex([mary,loved,a,computer]).
```

The Lexicon The lexicon is a set of Prolog facts of the form:

```
lex(Word, Category, Frequency).
```

Where `Word` is a word, `Category` is a Prolog representation of the CG category assigned to that word and `Frequency` is the number of times this category has been assigned to this word up to the current point in the learning process.

The Rules The CG functional application rules (see Section 2) are supplied to the learner. Extra rules may be added in future for fuller grammatical coverage.

The Categories The learner has a complete set of the categories that can be assigned to a word in the lexicon. The complete set is shown in Table 1.

The Parser The system employs a probabilistic chart parser, which calculates the N most probable parses, where N is the beam set by the user. The probability of a word being assigned a category is based on the relative frequency, which is calculated from the current lexicon. This probability is smoothed (for words that have not been given fixed categories prior to execution) to allow the possibility that the word may appear as other categories. For all categories for which the word has not appeared, it·is given a frequency of one. This is particularly useful for new words, as it ensures the category of a word is determined by its context.

Each non-lexical edge in the chart has a probability calculated by multiplying the probabilities of the two edges that are combined to form it. Edges between two vertices are not added if there are N edges labelled with the same category and a higher probability, between the same two vertices (if one has a lower probability it is replaced). Also, for efficiency, edges are not added between vertices if there is an edge already in place with a much higher probability. The chart in Figure 3 shows examples of edges that would not be added. The top half of the chart shows one parse and the bottom half another. If N was set to 1 then the dashed edge spanning all the vertices would not be added, as it has a lower probability than the other `s` edge covering the same vertices. Similarly, the dashed edge between the first and third vertices would not be added, as the probability of the `n` is so much lower than the probability of the `np`.

It is important that the parser is efficient, as it is used on every example and each word in an example may be assigned any category. As will be seen it is also used extensively in selecting the best parses. In future we hope to investigate the possibility of using more restricted parsing techniques, *e.g.* deterministic parsing technology such as that described by Marcus (Marcus, 1980), to increase efficiency and allow

Syntactic Role	CG Category	Example
Sentence	s	*the dog ran*
Noun	n	*dog*
Noun Phrase	np	*the dog*
Intransitive Verb	s\np	*ran*
Transitive Verb	(s\np)/np	*kicked*
Ditransitive Verb	((s\np)/np)/np	*gave*
Sentential Complement Verb	(s\np)/s	*believe*
Determiner	np/n	*the*
Adjective	n/n	*hungry*
Auxiliary Verb	(s\np)/(s\np)	*does*
That complementizer	np/s	*that*
Preposition	(n\n)/np ((s\np)\(s\np))/np	*to*

Table 1: The categories available to the learner

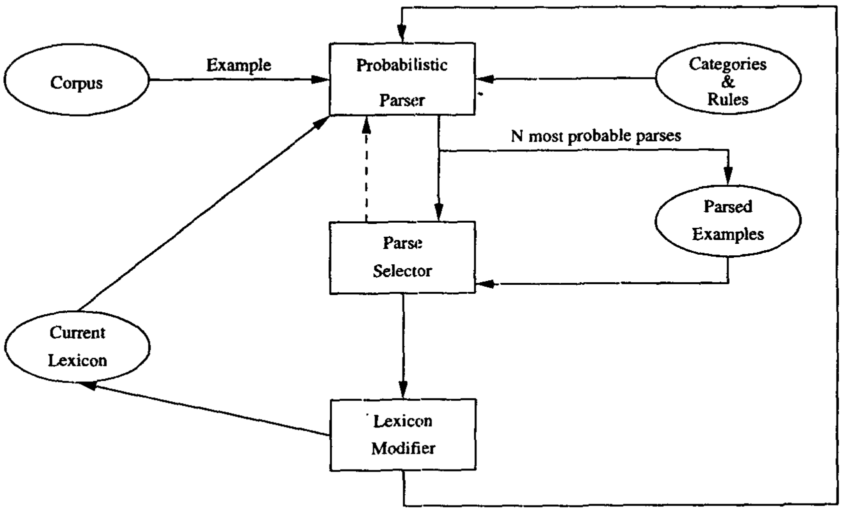

Figure 2: A Diagram of the Structure of the Learner

larger scale experiments.

3.2 The Learning Procedure

Having described the various components with which the learner is provided, we now describe how they are used in the learning procedure.

Parsing the Examples Examples are taken from the corpus one at a time and parsed. Each example is stored with the group of parses generated for it, so they can be efficiently accessed in future. The parse that is selected (see below) as the current correct parse is maintained at the head of this group. The head parse contributes information to the lexicon and annotates the corpus. The parses are also used extensively for the efficiency of the parse selection module, as will be described below. When the parser fails to find an analysis of an example, either because it is ungrammatical, or because of the incompleteness of the coverage of the grammar, the system skips to the next example.

The Parse Selector Once an example has been parsed, the N most probable parses are considered in turn to determine which can be used to make the most compressive lexicon (by a given measure), following the compression as learning approach of, for example, Wolff (Wolff, 1987). The current size measure for the lexicon

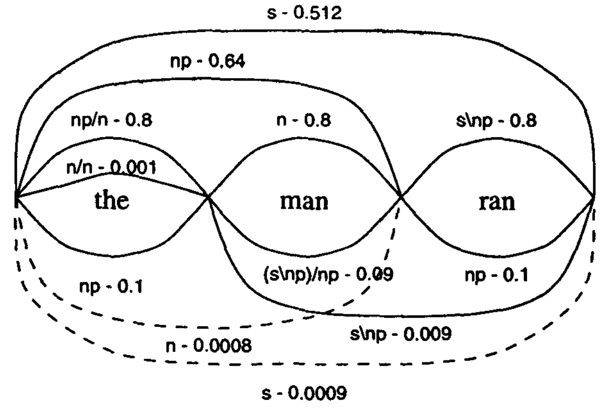

Figure 3: Example chart showing edge pruning

is the sum of the sizes of the categories for each lexical entry. The size of a category is the number of atomic categories within it. However, it is not enough to look at what a parse would add to the lexicon. The effect of changing the lexicon on the parses of previous examples must be considered. Changes in the frequency of assignments can cause the probabilities of previous parses to change and thus correct mistakes made earlier when the evidence from the lexicon was too weak to assign the correct parse. This correction is affected by reparsing previous examples that may be affected by the addition of the new parse to the lexicon. Not reparsing those examples that will not be affected, saves a great deal of time. In this way a new lexicon is built from the reparsed examples for each hypothesised parse of the current example. The parse leading to the most compressive of these is chosen. The amount of reparsing is also reduced by using stored parse information.

This may appear an expensive way of determining which parse to select, but it enables the system to calculate the most compressive lexicon and keep an up-to-date annotation for the corpus. Also, the chart parser works in polynomial time and it is possible to do significant pruning, as outlined, so few sentences need to be reparsed each time. However, in the future we will look at ways of determining which parse to select that do not require complete reparsing.

Lexicon Modification The final stage takes the current lexicon and replaces it with the lexicon built with the selected parse. The whole process is repeated until all the examples have been parsed. The final lexicon is left after the final modification. The most probable annotation of the corpus is the set of top-most parses after the final parse selection.

4 Experiments

Experiments were performed on three different corpora all containing only positive examples. Experiments were performed with and without a partial lexicon of closed-class words (words of categories with a finite number of members) with fixed categories and probabilities, *e.g.* determiners and prepositions. All experiments were carried out on a SGI Origin 2000.

Experiments on Corpus 1 The first corpus was built from a context-free grammar (CFG), using a simple random generation algorithm. The CFG (shown in Figure 4) covers a range of simple declarative sentences with intransitive, transitive and ditransitive verbs and with adjectives. The lexicon of the CFG contained 39 words with an example of noun-verb ambiguity. The corpus consisted of 500 such sentences (Figure 5 shows examples). As the size of the lexicon was small and there was only a small amount of ambiguity, it was unnecessary to supply the partial lexicon, but the experiment was carried out for comparison. We also performed an experiment on 100 unseen examples to see how accurately they were parsed with the learned lexicon. The results were manually verified to determine how many sentences were parsed correctly.

S → NP VP	VP → Vbar
Vbar → IV	Vbar → TV NP
Vbar → DV NP NP	NP → PN
NP → Nbar	Nbar → Det N
N → Adj N	
PN → john	Det → the
N → boy	Adj → small
IV → ran	TV → timed
DV → gave	

Figure 4: The CFG used to generate Corpus 1 with example lexical entries

Experiments on Corpus 2 The second corpus was generated in the same way, but using extra rules (see Figure 6) to include prepositions, thus making the fragment of English

```
ex([mary, ran]).
ex([john, gave, john, a, boy]).
ex([a, dog, called, the, fish, a, small,
ugly, desk]).
```

Figure 5: Examples from Corpus 1

more complicated. The lexicon used for generating the corpus was larger – 44 words in total. Again 500 examples were generated (see Figure 7 for examples) and experiments were carried out both with and without the partial lexicon. Again we performed an experiment on 100 unseen examples to see how accurately they are parsed.

NP → Nbar PP   VP → Vbar PP
PP → P NP

P → on

Figure 6: The extra rules required for generating Corpus 2 with example lexical entries

```
ex([the, fish, with, a, elephant, gave,
banks, a, dog, with, a, bigger, statue]).
ex([a, elephant, with, jim,
walked, on, a, desk]).
ex([the, girl, kissed, the, computer,
on, a, fish]).
```

Figure 7: Examples from Corpus 2

Experiments on Corpus 3 (The LLL Corpus) Finally, we performed experiments using the LLL corpus (Kazakov et al., 1998). This is a corpus of generated sentences for a substantial fragment of English. It is annotated with a certain amount of semantic information, which was ignored. The corpus contains 554 sentences, however, because of the restricted set of categories and CG rules, we limited the experiments to the 157 declarative sentences (895 words, with 152 unique words) in the corpus. Examples are shown in Figure 8. While our CG rules can handle a reasonable variety of declarative sentences it is by no means complete, not allowing any movement (*e.g.* topicalised sentences) or even any adverbs yet. This was, unsurprisingly, something of a limitation. Also, this corpus is very small and sparse, making learning difficult. It was determined to experiment to see how well the system performed under these conditions. Again we performed experiments with and without fixed closed-class words. Due to the lack of examples it was not possible to perform a test on unseen examples, which need to be pursued in the future.

```
ex([no, manager, in, sandy,
reads, every, machine]).
ex([the, manual, isnt, continuing]).
ex([no, telephone, sees, the, things]).
```

Figure 8: Examples from Corpus 3

All experiments were performed with the minimum number of categories needed to cover the corpus, so for example, in the experiments on Corpus 1 the categories for prepositions were not available to the parser. This will obviously affect the speed with which the learner performs. Also, the parser was restricted to two possible parses in each case.

5 Results

In Table 2 we report the results of these experiments. The CCW Preset column indicates whether the closed-class words were provided or not. The lexicon accuracy column is a measure, calculated by manual analysis, of the percentage of lexical entries *i.e.* entries that have word-category pairs that can plausibly be accepted as existing in English. This should be taken together with the parse accuracy, which is the percentage of correctly parsed examples *i.e.* a linguistically correct syntactic analysis. The results for the first two corpora are extremely encouraging with 100% accuracy in both measures. While these experiments are only on relatively simple corpora, these results strongly suggest that the approach can be effective. It

Corpus	CCW Preset	Lexicon Acc. (%)	Parse Acc. (%)	Exec. Time (s)
1	×	100	100	5297
1	√	100	100	625
2	√	100	100	10524
3	×	14.7	0.6	164151
3	√	77.7	58.9	361

Table 2: Accuracies and timings for the different learning experiments

should be noted that any experiment on corpus 2 without the closed-class words being set did not terminate, as the sentences in that corpus are significantly longer and each word may be a large number of categories. It is therefore clear, that setting the closed-class words greatly increases speed and that we need to consider methods of relieving the strain on the parser if the approach is to be useful on more complex corpora.

The results with the LLL corpus are also encouraging in part. A lexical accuracy of 77.7% and a parse accuracy of nearly 60% (note this measure of accuracy is strict) on such a small sparse corpus is a good result and analysis suggests most errors were made due to the small coverage of the grammar - especially not allowing any movement. Errors also suggest that adding some further linguistic constraints - for example not allowing words to be assigned the basic category s - and strengthening the compression heuristic may provide improvements. It was these problems, along with the sparseness of the corpus, that led to the poor results with the LLL corpus without preset words.

Table 3 shows predictably good results for parsing the test sets with the learned lexicons.

Corpus	Closed-Class	Parse Accuracy (%)
1	×	100
1	√	100
2	×	100
2	√	100

Table 3: Unseen example parsing accuracy

6 Conclusions

We have presented an unsupervised learner that is able to both learn CG lexicons and annotate natural language corpora, with less background knowledge than other systems in the literature. Results from preliminary experiments are encouraging with respect to both problems, particularly as the system appears to be reasonably effective on small, sparse corpora. It is encouraging that where errors arose this was often due only to incomplete background knowledge.

The results presented are encouraging with respect to the work that has already been mentioned - 100% can clearly not be improved upon and compares very favourable with the systems mentioned in Section 1. However, it is also clear that this was achieved on unrealistically simple corpora and when the system was used on the more diverse LLL corpus it did not fair as well. However, given the fact that the problem setting discussed here is somewhat harder than that attempted by other systems and the lack of linguistic background knowledge supplied, it is hoped that it will be possible to use the approach on wider coverage corpora more effectively in the future.

The use of CGs to solve the problem provides an elegant way of using syntactic information to constrain the learning problem and provides the opportunity for expansion to a full grammar learning system in the future by the development of a category hypothesizer. It is hoped that this will be part of future work.

We also hope to carry out experiments on larger and more diverse corpora, as the corpora used thus far are too small to be a an exacting test for the approach. We need to expand the grammar to cover more linguistic phenomena to achieve this, as well as considering other measures for compressing the lexicon (*e.g.* using an MDL-based approach). Larger experiments will lead to a need for increased efficiency in the parsing and reparsing processes. This could be done by considering deterministic parsing approaches (Marcus, 1980), or perhaps shallower syntactic analysis.

While many extensions may be considered for this work, the evidence thus far suggests that the approach outlined in this paper is effective and efficient for these natural language learning tasks.

References

Pieter Willem Adriaans. 1992. *Language Learning from a Categorial Perspective.* Ph.D. thesis, Universiteit van Amsterdam.

Y. Bar-Hillel, C. Gaifman, and E. Shamir. 1964. On categorial and phrase structure grammars. In *Language and Information* (Bar-Hillel, 1964), pages 99 - 115. First appeared in The Bulletin of the Research Council of Israel, vol. 9F, pp. 1-16, 1960.

Y. Bar-Hillel. 1964. *Language and Information.* Addison-Wesley.

Eric Brill. 1997. Unsupervised learning of disambiguation rules for part of speech tagging. In *Natural Language Processing Using Very Large Corpora.* Kluwer Academic Press.

Wojciech Buszkowski. 1987. Discovery procedures for categorial grammars. In Ewan Klein and Johan van Benthem, editors, *Categories, Polymorphism and Unification*, pages 35 – 64. Centre for Cognitive Science, University of Edinburgh and Institue for Language, Logic and Information, University of Amsterdam.

Donald Hindle. 1983. Deterministic parsing of syntactic non-fluencies. In Mitch Marcus, editor, *Proceedings of the 21st Annual Meeting of the Association for Computational Linguistics*, pages 123 – 128. Association for Computational Linguistics.

Makoto Kanazawa. 1994. *Learnable Classes of Categorial Grammars.* Ph.D. thesis, Institute for Logic, Language and Computation, University of Amsterdam.

Dimitar Kazakov, Stephen Pulman, and Stephen Muggleton. 1998. The FraCas dataset and the LLL challenge. Technical report, SRI International.

Julian Kupiec. 1992. Robust part-of-speech tagging using a hidden markov model. *Computer Speech and Language*, 6:225–242.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. Technical Report IRCS-93-47, Institution for Research in Cognitive Science.

Mitchell P. Marcus. 1980. *A Theory of Syntactic Recognition.* The MIT Press Series in Artificial Intelligence. The MIT Press.

Miles Osborne. 1997. Minimisation, indifference and statistical language learning. In *Workshop on Empirical Learning of Natural Language Processing Tasks, ECML'97*, pages 113 – 124.

Steven Pinker. 1990. Language acquisition. In Daniel N. Oshershon and Howard Lasnik, editors, *An Invitation to Cognitive Science: Language*, volume 1, pages 199–241. The MIT Press.

W. Daniel Solomon. 1991. Learning a grammar. Technical Report UMCS-AI-91-2-1, Department of Computer Science, Artificial Intelligence Group, University of Manchester.

Mark Steedman. 1993. Categorial grammar. *Lingua*, 90:221 – 258.

J.G. Wolff. 1987. Cognitive development as optimisation. In Leonard Bolc, editor, *Computational Models of Learning*, Symbolic computation-artificial intelligence. Springer Verlag.

Mary McGee Wood. 1993. *Categorial Grammars.* Linguistic Theory Guides. Routledge. General Editor Richard Hudson.