Learning Probabilistic Context-Free Grammars from Treebanks

Jose L. Verdú-Mas, Jorge Calera-Rubio, and Rafael C. Carrasco *

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, E-03071 Alicante, Spain.
{verdu,calera,carrasco}@dlsi.ua.es

Abstract. This paper describes the application of a new model to learn probabilistic context-free grammars (PCFGs) from a tree bank corpus. The model estimates the probabilities according to a generalized k-gram scheme for trees. It allows for faster parsing, decreases considerably the perplexity of the test samples and tends to give more structured and refined parses. In addition, it also allows several smoothing techniques such as *backing-off* or *interpolation* that are used to avoid assigning zero probability to any sentence.

1 Introduction

Context-free grammars may be considered to be the customary way of representing syntactical structure in natural language sentences. In many natural-language processing applications, obtaining the correct syntactical structure for a sentence is an important intermediate step before assigning an interpretation to it. But ambiguous parses are *very* common in real natural-language sentences (e.g., those longer than 15 words). A set of rather radical hypotheses as to how humans select the best parse tree [1] propose that a great deal of syntactic disambiguation may actually occur without the use of any semantic information; that is, just by selecting a preferred parse tree. It may be argued that the preference of a parse tree with respect to another is largely due to the relative frequencies with which those choices have lead to a successful interpretation. This sets the ground for a family of techniques which use a probabilistic scoring of parses to the correct parse in each case.

Probabilistic scorings depend on parameters which are usually estimated from data, that is, from parsed text corpora such as the Penn Treebank [2]. The most straightforward approach is that of *treebank grammars*, [3]. Treebank grammars are probabilistic context-free grammars in which the probability that a particular nonterminal is expanded according to a given rule is estimated as the relative frequency of that expansion by simply counting the number of times it appears in a manually-parsed corpus. This is the simplest probabilistic scoring

* The authors wish to thank the Generalitat Valenciana project CTIDIB/2002/173 and the Spanish CICyT project TIC2000-1599 for supporting this work.

A. Sanfeliu and J. Ruiz-Shulcloper (Eds.): CIARP 2003, LNCS 2905, pp. 537–544, 2003.

scheme, and it is not without problems; we will show how a set of approximate models, which we will call *offspring-annotated* models, in which expansion probabilities are dependent on the future expansions of children, may be seen as a generalization of the classic k-gram models to the case of trees, and include treebank grammars as a special case; other models, such as Johnson's [4] *parent-annotated* models (or more generally, *ancestry annotated* models) and IBM history-based grammars [5, p. 423],[6] offer an alternate approach in which the probability of expansion of a given nonterminal is made dependent on the previous expansions. An interesting property of many of these models is that, even though they may be seen as context-dependent, they may still be easily rewritten as context-free models in terms of specialized versions of the original nonterminals.

The next section proposes our generalization of the classic k-gram models to the case of trees, which is shown to be equivalent to having a specialized context-free grammar. A simplication of this model, called the *child-annotated* model or $k = 3$, for short, is also presented in that section.

2 The Model

Let $\Omega = \{\tau_1, \tau_2, \ldots, \tau_{|\Omega|}\}$ be a treebank, that is, a sample of parse trees.

For all $k > 0$ and for all trees $\tau = \sigma(t_1 t_2 \ldots t_m) \in \Omega$ we define the *k-root* of τ as the tree

$$r_k(\sigma(t_1 \ldots t_m)) = \begin{cases} \sigma & \text{if } k = 1 \\ \sigma(r_{k-1}(t_1) \ldots r_{k-1}(t_m)) & \text{otherwise} \end{cases} \tag{1}$$

The sets $f_k(t)$ of *k-forks* and $s_k(t)$ of *k-subtrees* are defined for all $k > 0$ as follows:

$$f_k(\sigma(t_1 \ldots t_m)) = \cup_{j=1}^{m} f_k(t_j) \cup \begin{cases} \emptyset & \text{if } 1 + \text{depth}(\sigma(t_1 \ldots t_m)) < k \\ r_k(\sigma(t_1 \ldots t_m)) & \text{otherwise} \end{cases} \tag{2}$$

$$s_k(\sigma(t_1 \ldots t_m)) = \cup_{j=1}^{m} s_k(t_j) \cup \begin{cases} \sigma(t_1 \ldots t_m) & \text{if } 0 < \text{depth}(\sigma(t_1 \ldots t_m)) < k \\ \emptyset & \text{otherwise} \end{cases} \tag{3}$$

where $depth(t)$ denotes the depth of the tree t having in own that in a single node tree it is zero.

We define the treebank probabilistic k testable grammar $G = (\mathcal{N}, \Sigma, \mathcal{S}, \mathcal{P})$ through:

- $\mathcal{N} = r_{k-1}(f_k(\Omega)) \cup s_{k-1}(\Omega) \cup \{\mathcal{S}\}$;
- Σ is the set of labels in Ω;
- $\mathcal{S}$ is the start symbol;

- $\mathcal{P} = \{(r, p(r)) \mid r \in R \wedge p(r) \in [0,1]\}$ where $R \subset \mathcal{N} \times (\mathcal{N} \cup \Sigma)^+$ is a set of production rules (usually written as $A \rightarrow \alpha$, where $A \in \mathcal{N}$ and $\alpha \in (\mathcal{N} \cup \Sigma)^+$) and $p(r)$ is the emission probability associated with the rule r. The set $\mathcal{P}$ is built as follows:
  - for every tree $t \in r_k(\Omega)$ add to $\mathcal{P}$ the rule $\mathcal{S} \longrightarrow t$ with probability

$$p(\mathcal{S} \longrightarrow t) = \frac{\sum_{\tau \in \Omega} \delta_{t\, r_{k-1}(\tau)}}{|\Omega|} \tag{4}$$

    where $\delta_{a\,b} = 1$ if $a = b$ and zero otherwise;
  - for every tree $\sigma(t_1 t_2 \ldots t_m) \in f_k(\Omega)$ add to $\mathcal{P}$ the rule $r_{k-1}(\sigma(t_1 t_2 \ldots t_m)) \longrightarrow t_1\ t_2\ \ldots\ t_m$ with probability

$$p(r_{k-1}(\sigma(t_1 t_2 \ldots t_m)) \longrightarrow t_1\ t_2\ \ldots\ t_m) = \frac{\sum_{\tau \in \Omega} C(\sigma(t_1 t_2 \ldots t_m), \tau)}{\sum_{\tau \in \Omega} C(r_{k-1}(\sigma(t_1 t_2 \ldots t_m)), \tau)} \tag{5}$$

    Here $C(t, \tau)$ counts the number of times that the fork t appers in the tree τ;
  - for every tree $\sigma(t_1 t_2 \ldots t_m) \in s_k(\Omega)$ add to $\mathcal{P}$ the rule $\sigma(t_1 t_2 \ldots t_m) \longrightarrow t_1\ t_2\ \ldots\ t_m$ with probability

$$p(\sigma(t_1 t_2 \ldots t_m) \longrightarrow t_1\ t_2\ \ldots\ t_m) = 1 \tag{6}$$

Defined in this way, these probabilities satisfy the normalization constraint

$$\text{for each } A \in \mathcal{N}: \sum_{\alpha : A \longrightarrow \alpha \in \mathcal{P}} p(A \longrightarrow \alpha) = 1 \tag{7}$$

and the consistency constraint. PCFGs estimated from treebanks using the relative frequency estimator always satisfy those constraints [7] [8].

Note that in this kind of models, the expansion probability for a given node is computed as a function of the subtree of depth $k-2$ that the node generates, i.e., every non-terminal symbol stores a subtree of depth $k-2$. In the particular case $k = 2$, only the label of the node is taken into account (this is analogous to the standard bigram model for strings) and the model coincides with the simple rule-counting approach used in treebank grammars by Charniak [9].

However, in the case $k = 3$, we get a *child-annotated* model, that is, non-terminal symbols $\sigma(\sigma_1\ \sigma_2\ \ldots\ \sigma_m)$ are defined by:

- the node label σ,
- the number m of descendents (if any) and
- the labels in the descendents $\sigma_1, \sigma_2, \ldots, \sigma_m$ (if any) and their ordering.

As an ilustration, consider a very simple sample with only the tree in the figure 1. If we choose $k = 2$, then

- $r_1(\mathrm{S(NP\ VP)}) = \mathrm{S}$;

- $f_2(\mathrm{S(NP\ VP)}) = \{\mathrm{S(NP\ VP)}, \mathrm{NP(N)}, \mathrm{VP(V\ NP)}, \mathrm{NP(NP\ PP)}, \mathrm{PP(P\ NP)}\}$
- $s_1(\mathrm{S(NP\ VP)}) = \emptyset$

and the CFG is

$$G^{[2]} = (\{\mathrm{S}, \mathrm{NP}, \mathrm{VP}, \mathrm{PP}\}, \{\mathrm{N}, \mathrm{V}, \mathrm{P}\}, \mathrm{S}, \mathcal{P}),$$

with $\mathcal{P}$ containing the rules

$$\begin{aligned}
\mathrm{S} &\rightarrow \mathrm{NP\ VP}\\
\mathrm{NP} &\rightarrow \mathrm{NP\ PP}\\
\mathrm{NP} &\rightarrow \mathrm{N}\\
\mathrm{VP} &\rightarrow \mathrm{VP\ PP}\\
\mathrm{VP} &\rightarrow \mathrm{V\ NP}\\
\mathrm{PP} &\rightarrow \mathrm{P\ NP}
\end{aligned}$$

However, for $k = 3$ we obtain

- $r_2(\mathrm{S(NP\ VP)}) = \mathrm{S(NP\ VP)}$
- $f_3(\mathrm{S(NP\ VP)}) = \{\mathrm{S(NP(N)\ VP(V\ NP))}, \mathrm{VP(V\ NP(NP\ PP))},$
  $\mathrm{NP(NP(N)\ PP(P\ NP))}, \mathrm{PP(P\ NP(N))}\}$
- $s_2(\mathrm{S(NP\ VP)}) = \{\mathrm{NP(N)}\}$

and the CFG is

$$G^{[3]} = (\{\mathrm{S}, \mathrm{S(NP\ VP)}, \mathrm{NP(N)}, \mathrm{VP(V\ NP)}, \mathrm{NP(NP\ PP)}, \mathrm{PP(P\ NP)}\}, \{\mathrm{N}, \mathrm{V}, \mathrm{P}\}, \mathrm{S}, \mathcal{P}),$$

with $\mathcal{P}$ containing the rules

$$\begin{aligned}
\mathrm{S} &\longrightarrow \mathrm{S(NP\ VP)}\\
\mathrm{S(NP\ VP)} &\longrightarrow \mathrm{NP(N)\ VP(V\ NP)}\\
\mathrm{VP(V\ NP)} &\longrightarrow \mathrm{V\ NP(NP\ PP)}\\
\mathrm{NP(NP\ PP)} &\longrightarrow \mathrm{NP(N)\ PP(P\ NP)}\\
\mathrm{PP(P\ NP)} &\longrightarrow \mathrm{P\ NP(N)}\\
\mathrm{NP(N)} &\longrightarrow \mathrm{N}
\end{aligned}$$

For comparison, if one uses a parent-annotated version of the grammar (following Johnson [4]), one gets the following rules [1] (where the superindex is the parent's label).

$$\begin{aligned}
\mathrm{S} &\longrightarrow {}^{\mathrm{S}}\mathrm{NP}\ {}^{\mathrm{S}}\mathrm{VP}\\
{}^{\mathrm{S}}\mathrm{NP} &\longrightarrow \mathrm{N}\\
{}^{\mathrm{S}}\mathrm{VP} &\longrightarrow \mathrm{V}\ {}^{\mathrm{VP}}\mathrm{NP}\\
{}^{\mathrm{VP}}\mathrm{NP} &\longrightarrow {}^{\mathrm{NP}}\mathrm{NP}\ {}^{\mathrm{NP}}\mathrm{PP}\\
{}^{\mathrm{NP}}\mathrm{NP} &\longrightarrow \mathrm{N}\\
{}^{\mathrm{NP}}\mathrm{PP} &\longrightarrow \mathrm{P}\ {}^{\mathrm{PP}}\mathrm{NP}\\
{}^{\mathrm{PP}}\mathrm{NP} &\longrightarrow \mathrm{N}
\end{aligned}$$

[1] As will be seen in section 3, parent-annotated grammars usually have *less* parameters than child-annotated grammars, contrary to what this example may suggest.

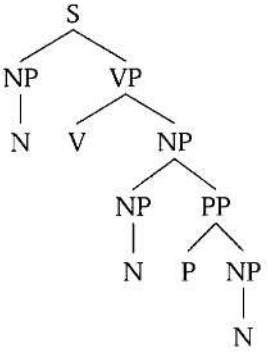

Fig. 1. A sample parse tree

3 Experiments

3.1 General Conditions

We have performed a series of experiments to assess the structural disambiguation performance of offspring-annotated models as compared to standard treebank grammars, that is, to compare their relative ability for selecting the best parse tree. To better put these comparisons in context, we have also evaluated Johnson's [4] parent annotation scheme. To build training corpora and test sets of parse trees, we have used English parse trees from the Penn Treebank, release 3. In all experiments the training corpus, consisted of all of the trees (41,532) in sections 02 to 22 of the *Wall Street Journal* portion of Penn Treebank, modified as above. This gives a total number of more than 600,000 subtrees. The test set contained all sentences in section 23 having no more than 40 words.

A Chappelier and Rajman's [10] probabilistic extended Cocke-Younger-Kasami parsing algorithm (which constructs a table containing generalized items like those in Earley's [11] algorithm) was used to obtain the most likely parse for each sentence in the training set; this parse was compared to the corresponding gold-standard tree in the test set using the customary PARSEVAL evaluation metric [12, 5, p. 432] after deannotating the most likely tree delivered by the parser. PARSEVAL gives partial credit to incorrect parses by establishing the *labeled precision* (P) and *labeled recall* (R) measures.

3.2 Structural Disambiguation Results

Here is a list of the models which were evaluated:

- A standard treebank grammar, with no annotation of node labels (NO or $k = 2$), with probabilities for 15,140 rules.

- A child-annotated grammar (CHILD or $k = 3$), with probabilities for 92,830 rules.
- A parent-annotated grammar (PARENT), with probabilities for 23,020 rules.
- A both parent- and child-annotated grammar (BOTH), with probabilities for 112,610 rules.

As expected, the number of rules obtained increases as more information is conveyed by the node label, although this increase is not extreme. On the other hand, as the generalization power decreases, some sentences in the test set become unparsable, that is, they cannot be generated by the grammar.

ANNOTATION	R	P	$f_{R=100\%}$	EXACT	PARSED	t
NO ($k = 2$)	70.7%	76.1%	10.4%	10.0%	100%	57
CHILD ($k = 3$)	79.2%	74.2%	19.4%	13.6%	94.6%	9
PARENT	80.0%	81.9%	18.5%	16.3%	100%	340
BOTH	80.1%	75.6%	20.5%	14.7%	79.6%	75

Table 1. Parsing results with different annotation schemes: labelled recall R, labelled precision P, fraction of sentences with total labelled recall $f_{R=100\%}$, fraction of exact matches, fraction of sentences parsed by the annotated model, and average time per sentence in seconds.

The results in table 1 show that

- The parsing performance of parent-annotated and child-annotated PCFG is similar and better than those obtained with the standard treebank PCFG. The performance is measured both with the customary PARSEVAL metrics and by counting the number of maximum-likelihood trees that (a) match their counterparts in the treebank exactly, and (b) contain all of the constituents in their counterpart (100% labeled recall, $f_{R=100\%}$). The fact that child-annotated grammars do not perform better than parent-annotated ones may be due to their larger number of parameters compared to parent-annotated PCFG. This makes it difficult to estimate them accurately from currently available treebanks (only about 6 subtrees per rule in the experiments).
- The average time to parse a sentence shows that child annotation leads to parsers that are much faster. This comes as no surprise because the number of possible parse trees considered is drastically reduced; this is, however, not the case with parent-annotated models.

It may be worth mentioning that parse trees produced by child-annotated models tend to be more structured and refined than parent-annotated and unannotated parses which tend to use rules that lead to flat trees.

On the other hand, child-annotated models, CHILD and BOTH, were unable to deliver a parse tree for all sentences in the test set (CHILD parses 94.6% of the sentences and BOTH, 79.6%). To be able to parse all sentences, those smoothed models, were evaluated:

- A linear interpolated model, M1, where the probability of a tree t is

$$p(t) = \lambda p_3(t) + (1 - \lambda) p_2(t) \tag{8}$$

here, $p_3(t)$ and $p_2(t)$ are the probabilities of the tree t in, respectively, the model $k = 3$ and $k = 2$. The value of λ was 0.7 (selected to minimize the perplexity).

- A tree-level back-off, M2, where the highest order model such that the probability of the event is greater than zero is selected. Some care has to be taken in order to preserve normalization.
- A rule-level back-off model, M3 that builds a new PCFG from the rules of the tree-k-grammar models and adding new rules which allow to switch among those models. In particular, the new PCFG consists of three different kinds of rules:
  1. $k = 3$ rules with modified probability in order to preserve normalization,
  2. back-off rules that allow to switch to the lower model, and,
  3. modified $k = 2$ rules to switch-back to the higher model.

  The new grammar has 92,830 $k = 3$ rules, 15,140 $k = 2$ rules and 10,250 back-off rules.

MODEL	R	P	EXACT	PARSED	t
M1	80.2%	78.6%	17.4%	100%	57
M2	78.9%	74.2%	17.1%	100%	9.3
M3	82.4%	81.3%	17.5%	100%	68

Table 2. Parsing results with different smoothed models.

The results in table 2 show that:

- M2 is the fastest but its performance is worse than that of M1 and M3.
- M1 and M3 parse sentences at a comparable speed but recall and precision are better using M3.

Compared to un-smoothed models, smoothed ones:

- Cover the whole test set ($k = 3$ did not).
- Parsed at reasonable speed (compared to PARENT).
- Achieved acceptable performance ($k = 2$ did not).

4 Conclusion

We have introduced a new probabilistic context-free grammar model, *offspring-annotated* PCFG in which the grammar variables are specialized by annotating them with the subtree they generate up to a certain level. In particular, we have studied child-annotated models (one level) and have compared their parsing performance to that of unannotated PCFG and of parent-annotated PCFG [4]. Offspring-annotated models may be seen as a special case of a very general probabilistic state-based model, which in turn is based on probabilistic bottom-up tree automata. The experiments show that:

- The parsing performance of parent-annotated and the proposed child-annotated PCFG is similar.
- Parsers using child-annotated grammars are, however, much faster because the number of possible parse trees considered is drastically reduced; this is, however, not the case with parent-annotated models.
- Child-annotated grammars have a larger number of parameters than parent-annotated PCFG which may make it difficult to estimate them accurately from currently available treebanks.
- Child-annotated models tend to give very structured and refined parses instead of flat parses, a tendency not so strong for parent-annotated grammars.

References

1. L. Frazier and K. Rayner. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14:178–210, 1982.
2. Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19:313–330, 1993.
3. Eugene Charniak. Treebank grammars. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, pages 1031–1036. AAAI Press/MIT Press, 1996.
4. Mark Johnson. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632, 1998.
5. Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.
6. Ezra Black, Frederick Jelinek, John D. Lafferty, David M. Magerman, Robert L. Mercer, and Salim Roukos. Towards history-based grammars: Using richer models for probabilistic parsing. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 31–37, 1992.
7. J.A. Sánchez and J.M. Benedí. Consistency of stochastic context-free grammars from probabilistic estimation based on growth transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(9):1052–1055, 1997.
8. Zhiyi Chi and Stuart Geman. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299–305, 1998.
9. Eugene Charniak. Tree-bank grammars. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and the Eighth Innovative Applications of Artificial Intelligence Conference*, pages 1031–1036, Menlo Park, 1996. AAAI Press/MIT Press.
10. J.-C. Chappelier and M. Rajman. A generalized CYK algorithm for parsing stochastic CFG. In *Actes de TAPD'98*, pages 133–137, 1998.
11. J. Earley. An efficient context-free parsing algorithm. *Communications of the ACM*, 13(2):94–102, 1970.
12. Ezra Black, Steven Abney, Dan Flickinger, Claudia Gdaniec, Ralph Grishman, Philip Harrison, Donald Hindle, Robert Ingria, Frederick Jelinek, Judith Klavans, Mark Liberman, Mitch Marcus, Salim Roukos, Beatrice Santorini, and Tomek Strzalkowski. A procedure for quantitatively comparing the syntatic coverage of English grammars. In *Proc. Speech and Natural Language Workshop 1991*, pages 306–311, San Mateo, CA, 1991. Morgan Kauffmann.