Tree-gram Parsing
Lexical Dependencies and Structural Relations

K. Sima'an
Induction of Linguistic Knowledge, Tilburg University &
Computational Linguistics, University of Amsterdam,
Spuistraat 134, 1012 VB Amsterdam, The Netherlands.
Email: khalil.simaan@hum.uva.nl

Abstract

This paper explores the kinds of probabilistic relations that are important in syntactic disambiguation. It proposes that two widely used kinds of relations, *lexical dependencies* and *structural relations*, have complementary disambiguation capabilities. It presents a new model based on structural relations, the *Tree-gram model*, and reports experiments showing that structural relations should benefit from enrichment by lexical dependencies.

1 Introduction

Head-lexicalization currently pervades in the parsing literature e.g. (Eisner, 1996; Collins, 1997; Charniak, 1999). This method extends every treebank nonterminal with its head-word: the model is trained on this *head lexicalized treebank*. Head lexicalized models extract probabilistic relations between *pairs of lexicalized nonterminals* ("bilexical dependencies"): every relation is between a parent node and one of its children in a parse-tree. Bilexical dependencies generate parse-trees for input sentences via Markov processes that generate Context-Free Grammar (CFG) rules (hence Markov Grammar (Charniak, 1999)).

Relative to Stochastic CFGs (SCFGs), bilexical dependency models exhibit good performance. However, bilexical dependencies capture many but not all relations between words that are crucial for syntactic disambiguation.We give three examples of kinds of relations not captured by bilexical-dependencies. Firstly, relations between non-head words of phrases, e.g. the relation between "more" and "than" in "more apples than oranges" or problems of PP attachments as in "he ate pizza (with mushrooms)/(with a fork)". Secondly, relations between three or more words are, by definition, beyond bilexical dependencies (e.g. between "much more" and "than" in "much more apples than oranges"). Finally, it is unclear how bilexical dependencies help resolve the ambiguity of idioms, e.g. "Time flies like an arrow" (neither "time" prefers to "fly", nor the fictitios beasts "Time flies" have taste for an "arrow").

The question that imposes itself is, indeed, *what relations might complement bilexical dependencies ?* We propose that bilexical dependencies can be complemented by *structural relations* (Scha, 1990), i.e. cooccurrences of syntactic structures, including actual words. An example model that employs one version of structural relations is Data Oriented Parsing (DOP) (Bod, 1995). DOP's parameters are "subtrees", i.e. connected subgraphs of parse-trees that constitute combinations of CFG rules, including terminal rules.

Formally speaking, "bilexical dependencies" and "structural relations" define two disjoint sets of probabilistic relations. Bilexical dependencies are relations defined over direct dominance head lexicalized nonterminals (see (Satta, 2000)); in contrast, structural relations are defined over words and arbitrary size syntactic structures (with non-lexicalized nonterminals). Apart from formal differences, they also have complementary advantages. Bilexical-dependencies capture influential lexical relations between heads and dependents. Hence, all bilexical dependency probabilities are *conditioned on lexical information* and lexical information is available at every point in the parse-tree. Structural relations, in contrast, capture many relations not captured by bilexical-dependencies (e.g. the examples above). However, structural relations do not always percolate lexical information up the parse-tree since their probabilities are not always *lexicalized.* This is a serious disadvantage when parse-trees are generated for *novel* input sentences since e.g. subcat frames are hypothesized for nodes high in the parse-tree without reference to their head words.

So, theoretically speaking, bilexical dependencies and structural relations have complementary aspects. But, *what are the empirical merits and limitations of structural relations ?* This paper presents a new model based on *structural relations*, the Tree-gram model, which allows head-driven parsing. It studies the effect of percolating head categories on performance and compares the performance of structural relations to bilexical dependencies. The comparison is conducted on the Wall Street Journal (WSJ) corpus (Marcus et al., 1993). In the remainder, we introduce the Tree-gram model in section 2, discuss practical issues in section 3, exhibit and discuss the results in section 4, and in section 5 we give our conclusions.

2 The Tree-gram model

For observing the effect of percolating information up the parse-tree on model behavior, we introduce *pre-head enrichment*, a structural variant of head-lexicalization. Given a training treebank TB, for every non-leaf node μ we mark one of its children as the *head-child*, i.e. the child that dominates the head-word[1] of the constituent under μ. We then enrich this treebank by attaching to the label of every phrasal node (i.e. nonterminal that is not a POS-tag) a *pre-head* representing its head-word. The pre-head of node μ is extracted from the constituent parse-tree under node μ. In this paper, the pre-head of μ consists of 1) the POS-tag of the head-word of μ (called 1^{st} order pre-heads or 1^{PH}), and possibly 2) the label of the mother node of that POS-tag (called 2^{nd} order or 2^{PH}). Pre-heads here also include other information defined in the sequel, e.g. subcat frames. The complex categories that result from the enrichment serve as the nonterminals of our training treebank; we refer to the original treebank symbols as "WSJ labels".

[1]Head-identification procedure by (Collins, 1997).

2.1 Generative models

A probabilistic model assigns a probability to every parse-tree given an input sentence S, thereby distinguishing one parse $T^* = argmax_T\ P(T|S) = argmax_T\ P(T, S)$. The probability $P(T, S)$ is usually estimated from cooccurrence statistics extracted from a treebank. In generative models, the tree T is generated in top down derivations that rewrite the start symbol TOP into the sentence S. Each rewrite-step involves a "rewrite-rule" together with its estimated probability. In the present model, the "rewrite-rules" differ from the CFG rules and combinations thereof that can be extracted from the treebank. We refer to them as *Tree-grams* (abbreviated T-grams). T-grams provide a more general-form for Markov Grammar rules (Collins, 1997; Charniak, 1999) as well as DOP subtrees. In comparison with DOP subtrees, T-grams capture more structural relations, allow head-driven parsing and are easier to combine with bilexical-dependencies.

2.2 T-gram extraction

Given a parse T from the training treebank, we extract three disjoint T-gram sets, called *roles*, from every one of its non-leaf nodes[2] μ: the head-role $\mathcal{H}(\mu)$, the left-dependent role $\mathcal{L}(\mu)$ and the right-dependent role $\mathcal{R}(\mu)$. The role of a T-gram signifies the T-gram's contribution to stochastic derivations: $t \in \mathcal{H}$ carries a head-child of its root node label, $t \in \mathcal{L}$ ($t \in \mathcal{R}$) carries left (resp. right) dependents for other head T-grams that have roots labeled the same as the root of t. Like in Markov Grammars, a head-driven derivation generates first a head-role T-gram and attaches to it left- and right-dependent role T-grams. We discuss

[2]Assuming that every node has a unique address.

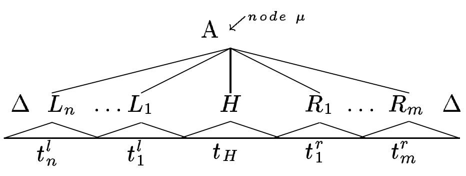

Figure 1: Constituent under node μ: $d > 1$.

these derivations right after we specify the T-gram extraction procedure.

Let d represent the depth[3] of the constituent tree-structure that is rooted at μ, H represent the label of the head-child of μ, and Δ represent the special *stop* symbol that encloses the children of every node (see figure 1). Also, for convenience, let δ^n_k be equal to Δ iff $k = n$ and $NILL$ (i.e. the empty tree-structure) otherwise. We specify the extraction for $d = 1$ and for $d > 1$. When $d = 1$, the label of μ is a POS-tag and the subtree under μ is of the form $pt \rightarrow \Delta w \Delta$, where w is a word. In this case $\mathcal{H}(\mu) = \{pt \rightarrow \Delta w \Delta\}$ and $\mathcal{L}(\mu) = \mathcal{R}(\mu) = \emptyset$. When $d > 1$: the subtree under μ has the form $A \rightarrow \Delta L_n(t^l_n) \ldots L_1(t^l_1)\ H(t_H)\ R_1(t^r_1) \ldots R_m(t^r_m)\Delta$ (figure 1), where every t^l_i, t^r_j and t_H is the subtree dominated by the child node of μ (labeled respectively L_i, R_j or H) whose address we denote respectively with $child_L(\mu, i)$, $child_R(\mu, j)$ and $child_H(\mu)$. We extract three sets of T-grams from μ:

$\mathcal{H}(\mu)$: contains $\forall\ 1 \leq i < n$ and $1 \leq j < m$, $A \rightarrow \delta^n_i L_i(X^l_i) \ldots H(X_h) \ldots R_j(X^r_j)\delta^m_j$, where X_h is either in $\mathcal{H}(child_H(\mu))$ or $NILL$, and every X^l_z (resp. X^r_z) is either a T-gram from $\mathcal{H}(child_L(\mu, z))$ (resp. $\mathcal{H}(child_R(\mu, z))$) or $NILL$.

$\mathcal{L}(\mu)$: contains $A \rightarrow \delta^n_k L_k(X_k) \ldots L_i(X_i)$, for all $1 \leq i \leq k < n$, where every X_z, $i \leq z \leq k$, is either a T-gram from $\mathcal{H}(child_L(\mu, z))$ or $NILL$,

$\mathcal{R}(\mu)$: contains $A \rightarrow R_i(X_i) \ldots R_k(X_k)\delta^m_k$, for all $1 \leq i \leq k < m$, where every X_z, $i \leq z \leq k$, is either a T-gram from $\mathcal{H}(child_R(\mu, z))$ or $NILL$,

[3]The depth of a (sub)tree is the number of edges in the longest path from its root to a leaf node.

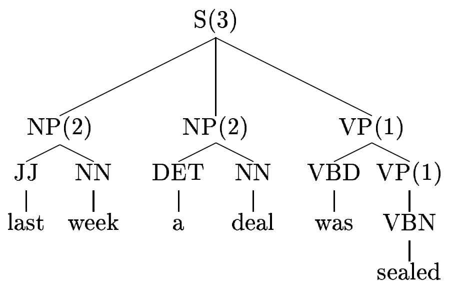

Figure 2: An example parse-tree.

Note that every T-gram's non-root and non-leaf node dominates a *head-role* T-gram (specified by $\mathcal{H}(child \cdots)$).

A non-leaf node μ labeled by nonterminal A is called *complete*, denoted "$[A]$", iff Δ delimits its sequence of children from both sides; when Δ is to the left (right) of the children of the node, the node is called *left (resp. right) complete*, denoted "$[A$" (resp. "$A]$"). When μ is not left (right) complete it is *open from the left* (resp. *right*); when μ is left and right open, it is called *open*.

Figure 2 exhibits a parse-tree[4]: the number of the head-child of a node is specified between brackets. Figure 3 shows some of the T-grams that can be extracted from this tree.

Having extracted T-grams from all non-leaf nodes of the treebank, we obtain $\mathcal{H} = \bigcup_{\mu \in TB}\mathcal{H}(\mu)$, $\mathcal{L} = \bigcup_{\mu \in TB}\mathcal{L}(\mu)$ and $\mathcal{R} = \bigcup_{\mu \in TB}\mathcal{R}(\mu)$. $\mathcal{H}_A$, $\mathcal{L}_A$ and $\mathcal{R}_A$ represent the subsets of resp. $\mathcal{H}$, $\mathcal{L}$ and $\mathcal{R}$ that contain those T-grams that have roots labeled A. $X_A(B) \in \{\mathcal{L}_A(B), \mathcal{R}_A(B), \mathcal{H}_A(B)\}$ specifies that the extraction took place on some treebank B other than the training treebank.

2.3 T-gram generative processes

Now we specify T-gram derivations assuming that we have an estimate of the probability of a T-gram. We return to this issue right after this. A stochastic derivation starts from the start nonterminal TOP. TOP is a single node partial parse-tree which is simultaneously the root and the only leaf node. A derivation terminates when two conditions are met (1) every non-leaf node in the generated parse-tree is *complete* (i.e. Δ delimits its children from

[4]Pre-heads are omitted for readability.

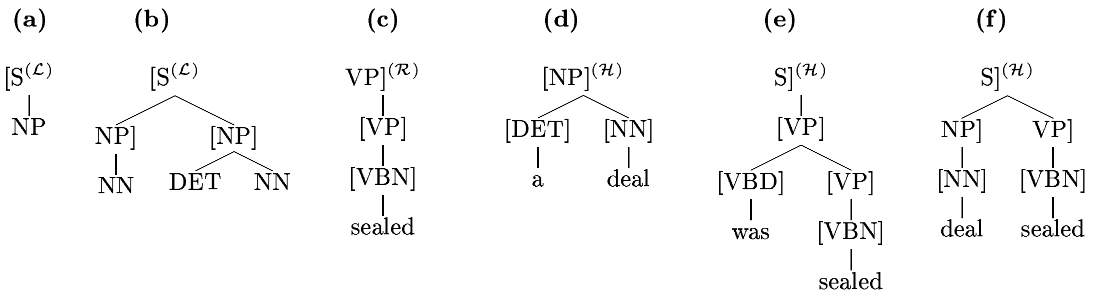

Figure 3: Some T-grams extracted from the tree in figure 2: the superscript on the root label specifies the *T-gram role*,. e.g. the left-most T-gram is in the left-dependent role. Non-leaf nodes are marked with "[" and "]" to specify whether they are complete from the left/right or both (leaving open nodes unmarked).

both sides) and (2) all leaf nodes are labeled with terminal symbols. Let Π represent the current partial parse-tree, i.e. the result of the preceding generation steps, and let $\mathcal{C}_\Pi$ represent that part of Π that influences the choice of the next step, i.e. the conditioning history. The generation process repeats the following steps in some order, e.g. head-left-right:

Head-generation: Select a leaf node μ labeled by a nonterminal A, and let A generate a head T-gram $t \in \mathcal{H}_A$ with probability $P_H(t|A, \mathcal{C}_\Pi)$. This results in a partial parse-tree that extends Π at μ with a copy of t (as in CFGs and in DOP).

Modification: Select from Π a non-leaf node μ that is *not complete*. Let A be the label of μ and $T = A \rightarrow X_1(x_1) \cdots X_b(x_b)$ be the tree dominated by μ (see figure 4):

Left: if μ is not left-complete, let μ generate to the left of T a left-dependent T-gram $t = A^{(\mathcal{L})} \rightarrow L_1(l_1) \cdots L_a(l_a)$ from $\mathcal{L}_A$ with probability $P_L(t|A, \mathcal{C}_\Pi)$ (see figure 4 (L)); this results in a partial parse-tree that is obtained by replacing T in Π with $A \rightarrow L_1(l_1) \cdots L_a(l_a) X_1(x_1) \cdots X_b(x_b)$,

Right: this is the mirror case (see figure 4 (R)). The generation probability is $P_R(t|A, \mathcal{C}_\Pi)$.

Figure 5 shows a derivation using T-grams (e), (a) and (d) from figure 3 applied to T-gram $TOP \rightarrow S$. Note that each derivation-step probability is conditioned on A, the label of node μ in Π where the current rewriting is taking place, on the role ($\mathcal{H}$, $\mathcal{L}$ or $\mathcal{R}$) of the T-gram involved, and on the relevant history $\mathcal{C}_\Pi$. Assuming beyond this that stochastic independence between the various derivation steps holds, *the probability of a derivation is equal to the multiplication of the conditional probabilities of the individual rewrite steps.*

Unlike SCFGs and Markov grammars but like DOP, a parse-tree may be generated via different derivations. The probability of a parse-tree T is equal to the *sum of the probabilities of the derivations that generate it (denoted $der \Rightarrow T$)*, i.e. $P(T, S) = \sum_{der \Rightarrow T} P(der, S)$. However, because computing $argmax_T\ P(T, S)$ can not be achieved in deterministic polynomial time (Sima'an, 1996), we apply estimation methods that allow tractable parsing.

2.4 Estimating T-gram probabilities

Let $count(Y_1, \cdots Y_m)$ represent the occurrence count for joint event $\langle Y_1 \cdots Y_m \rangle$ in the training treebank. Consider a T-gram $t \in X_A$, $X_A \in \{\mathcal{L}_A, \mathcal{R}_A, \mathcal{H}_A\}$, and a conditioning history $\mathcal{C}_\Pi$. The estimate $\frac{count(t, X_A, \mathcal{C}_\Pi)}{\sum_{x \in X_A} count(x, X_A, \mathcal{C}_\Pi)}$ assumes no hidden elements (different derivations per parse-tree), i.e. it estimates the probability $P_X(t|A, \mathcal{C}_\Pi)$ directly from the treebank trees (henceforth *direct-estimate*). This estimate is employed in DOP and is not Maximum-Likelihood (Bonnema et al., 1999). We argue that the bias of the direct estimate allows approximating the preferred parse by the one generated by the Most Probable Derivation (MPD). This is beyond the scope

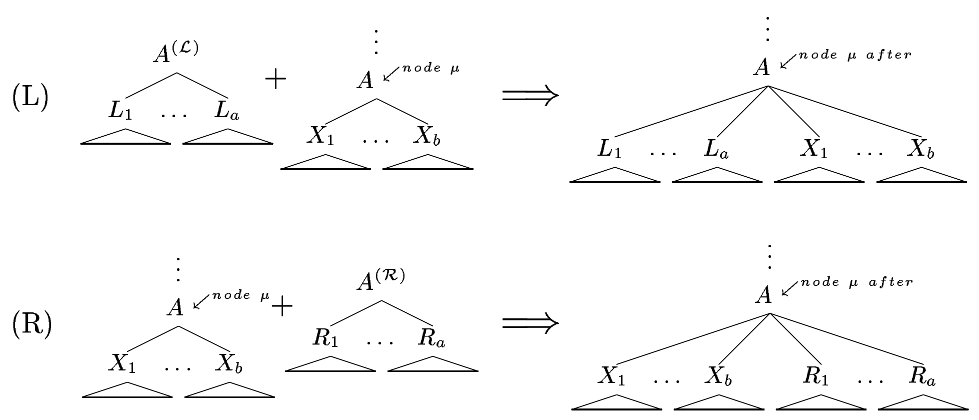

Figure 4: T-gram t is generated at μ: (L) $t \in \mathcal{L}_A$, (R) $t \in \mathcal{R}_A$

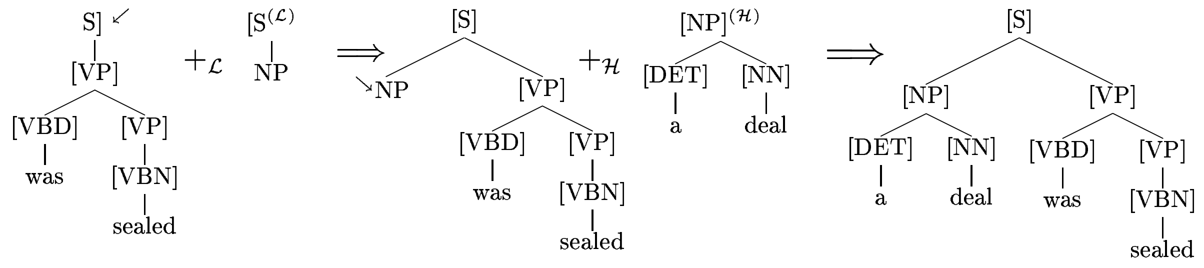

Figure 5: A T-gram derivation: the rewriting of TOP is not shown. An arrow marks the node where rewriting takes place. Following the arrows: 1. A left T-gram with root $[S$ is generated at node $S]$: S is complete. 2. A head-role T-gram is generated at node NP: all nodes are either complete or labeled with terminals.

of this paper and will be discussed elsewhere.

2.5 WSJ model instance

Up till now $\mathcal{C}_\Pi$ represented conditioning information anonymously in our model. For the WSJ corpus, we instantiate $\mathcal{C}_\Pi$ as follows: **1.** *Adjacency:* The flag $F_L(t)$ ($F_R(t)$) tells whether a left-dependent (right-dependent) T-gram t extracted from some node μ dominates a surface string that is adjacent to the head-word of μ (detail in (Collins, 1997)). **2.** *Subcat-frames:* (Collins, 1997) subcat frames are adapted: with every node μ that dominates a rule $A \rightarrow \Delta L_n \ldots L_1\ H\ R_1 \ldots R_m \Delta$ in the treebank (figure 1), we associate two (possibly empty) multisets of complements: SC^μ_L and SC^μ_R. Every complement in SC^μ_L (SC^μ_R) represents some left (right) complement-child of μ. This changes T-gram extraction as follows: *with every non-leaf node in a T-gram* that is extracted from a tree in this enriched treebank we have now a left and a right subcat frame associated. Consider the root node x in a T-gram extracted from node μ and let the children of x be $Y_1 \cdots Y_f$ (a subsequence of $\Delta L_n, \cdots, H, \cdots R_m \Delta$). The left (right) subcat frame of x is subsumed by SC^μ_L (resp. SC^μ_R) and contains those complements that correspond to the left-dependent (resp. right-dependent) children of μ that are *not* among $Y_1 \cdots Y_f$. Tree-gram derivations are modified accordingly: whenever a T-gram is generated (together with the subcat frames of its nodes) from some node μ in a partial-tree, the complements that its root dominates are removed from the subcat frames of μ. Figure 6 shows a small example of a derivation. **3.** *Markovian generation:* When node μ has empty subcat frames, we assume 1*st*-order Markov processes in generating both $\mathcal{L}$ and $\mathcal{R}$ T-grams around its $\mathcal{H}$ T-gram: LM^μ and RM^μ denote resp. the left- and rightmost children of node μ. Let XRM^μ and

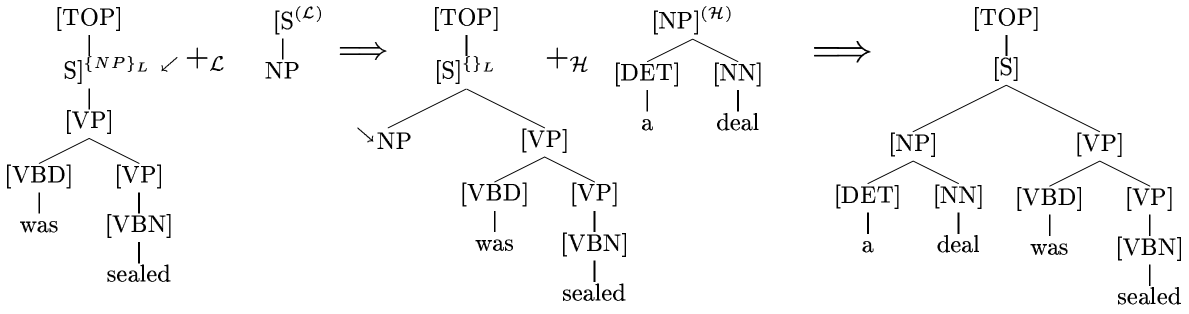

Figure 6: $S]^{\{NP\}_L}$ is a (left-open right-complete) node labeled S with a left subcat frame containing an NP. After the first rewriting, the subcat frame becomes empty since the NP complement was generated resulting in $[S]^{\{\}_L}$. The Other subcat frames are empty and are not shown here.

XLM^μ be equal to resp. RM^μ and LM^μ if the name of the T-gram system contains the word $+Markov$ (otherwise they are empty).

Let μ, labeled A, be the node where the current rewrite-step takes place, P be the WSJ-label of the parent of μ, and H the WSJ-label of the head-child of μ. Our probabilities are: $P_H(t|A,\mathcal{C}_\Pi) \approx P_H(t|A,P)$, $P_L(t|A,\mathcal{C}_\Pi) \approx P_L(t|A,H,SC_L^\mu,F_L(t),XRM^\mu)$, $P_R(t|A,\mathcal{C}_\Pi) \approx P_R(t|A,H,SC_R^\mu,F_R(t),XLM^\mu)$.

3 Implementation issues

Sections 02-21 WSJ Penn Treebank (Marcus et al., 1993) (release 2) are used for training and section 23 is held-out for testing (we tune on section 24). The parser-output is evaluated by "evalb"[5], on the PARSEVAL measures (Black et al., 1991) comparing a proposed parse P with the corresponding treebank parse T on Labeled Recall (LR = $\frac{number\ of\ correct\ constituents\ in\ P}{number\ of\ constituents\ in\ T}$), Labeled Precision (LP = $\frac{number\ of\ correct\ constituents\ in\ P}{number\ of\ constituents\ in\ P}$), and Crossing Brackets (CB = number of constituents in P that violate constituent boundaries in T).

T-gram extraction: The number of T-grams is limited by setting constraints on their form much like n-grams. One upperbound is set on the depth[6] (d), a second on the number of children of every node (b), a third on the sum of the number of nonterminal leafs with the number of (left/right) open-nodes (n), and a fourth (w) on the number of words in a T-gram. Also, a threshold is set on the frequency (f) of the T-gram. In the experiments $n \leq 4$, $w \leq 3$ and $f \geq 5$ are fixed while d changes. Unknown words and smoothing: We did not smooth the relative frequencies. Similar to (Collins, 1997), every word occurring less than 5 times in the training-set was renamed to CAP+UNKNOWN+SUFF, where CAP is 1 if its first-letter is capitalized and 0 otherwise, and SUFF is its suffix. Unknown words in the input are renamed this way before parsing starts. Tagging and parsing: An input word is tagged with all POS-tags with which it cooccurred in the training treebank. The parser is a two-pass CKY parser: the first pass employs T-grams that fulfill $d = 1$ in order to keep the parse-space under control before the second-pass employs the full Tree-gram model for selecting the MPD.

4 Empirical results

First we review the lexical-conditionings in previous work (other important conditionings are not discussed for space reasons). Magerman95 (Magerman, 1995; Jelinek et al., 1994) grows a decision-tree to estimate $P(T|S)$ through a history-based approach which conditions on actual-words. Charniak (Charniak, 1997) presents lexicalizations of SCFGs: the Minimal model conditions SCFG rule generation on the head-word of its left-hand side, while Charniak97 further con-

[5]http://www.research.att.com/ mcollins/.

[6]T-gram depth is the length of the longest path in the tree obtained by right/left-linearization of the T-gram around the T-gram nodes' head-children.

System	**LR%**	**LP%**	**CB**	**0CB%**	**2CB%**
Minimal (Charniak, 1997)	83.4	84.1	1.40	53.2	79.0
Magerman95 (Magerman, 1995)	84.6	84.9	1.26	56.6	81.4
Charniak97 (Charniak, 1997)	87.5	87.4	1.00	62.1	86.1
Collins97 (Collins, 1997)	88.1	88.6	0.91	66.4	86.9
Charniak99 (Charniak, 1999)	90.1	90.1	0.74	70.1	89.6
SCFG (Charniak, 1997)	71.7	75.8	2.03	39.5	68.1
T-gram ($d \leq 5$ (2^{PH}))	*82.9*	*85.1*	*1.30*	*58.0*	*82.1*

Table 1: Various results on WSJ section 23 sentences $\leq$ 40 words (2245 sentences).

ditions the generation of every constituent's head-word on the head-word of its parent-constituent, effectively using bilexical dependencies. Collins97 (Collins, 1997) uses a bilexicalized 0^{th}-order Markov Grammar: a lexicalized CFG rule is generated by projecting the head-child first followed by every left and right dependent, conditioning these steps on the head-word of the constituent. Collins97 extends this scheme to deal with subcat frames, adjacency, traces and wh-movement. Charniak99 conditions lexically as Collins does but also exploits up to 3^{rd}-order Markov processes for generating dependents. Except for T-grams and SCFGs, all systems smooth the relative frequencies with much care.

Sentences $\leq$ 40 words (including punctuation) in section 23 were parsed by various T-gram systems. Table 1 shows the results of some systems including ours. Systems conditioning mostly on lexical information are contrasted to SCFGs and T-grams. Our result shows that T-grams improve on SCFGs but fall short of the best lexical-dependency systems. Being 10-12% better than SCFGs, comparable with the Minimal model and Magerman95 and about 7.0% worse than the best system, it is fair to say that (depth 5) T-grams perform more like bilexicalized dependency systems than bare SCFGs.

Table 2 exhibits results of various T-gram systems. Columns 1-2 exhibit the traditional DOP observation about the effect of the size of subtrees/T-grams on performance. Columns 3-5 are more interesting: they show that even when T-gram size is kept fixed, systems that are pre-head enriched improve on systems that are not pre-head enriched (0^{PH}). This is supported by the result of column 1 in contrast to SCFG and Collins97 (table 1): the $D1$ T-gram system differs from Collins97 almost only in pre-head vs. head enrichment and indeed performs midway between SCFG and Collins97. This all suggests that allowing bilexical dependencies in T-gram models should improve performance. It is noteworthy that pre-head enriched systems are also more efficient in time and space. Column 6 shows that adding Markovian conditioning to subcat frames further improves performance suggesting that further study of the conditional probabilities of dependent T-grams is necessary. Now for any node in a gold / proposed parse, let node-height be the average path-length to a word dominated by that node. We set a threshold on node-height in the gold and proposed parses and observe performance. Figure 7 plots the F-score = (2*LP*LR)/(LP+LR) against node-height threshold. Clearly, performance degrades as the nodes get further from the words while pre-heads improve performance.

5 Conclusions

We started this paper wondering about the merits of structural-relations. We presented the T-gram model and exhibited empirical evidence for the usefulness as well as the shortcomings of structural relations. We also provided evidence for the gains from enrichment of structural relations with semi-lexical information. In our quest for better modeling, we still need to explore how structural-relations and bilexical dependencies can be combined. Probability estimation, smoothing and efficient implementations need special attention.

SYSTEM	$D^1(2^{PH})$	$D^4(2^{PH})$	$D^5(2^{PH})$	$D^5(1^{PH})$	$D^5(0^{PH})$	$D^5(2^{PH}) + Markov$
LR	80.03	82.42	82.57	82.85	81.35	82.93
LP	80.99	85.23	85.02	85.06	84.59	85.13
CB	1.70	1.32	1.44	1.43	1.48	1.30
#sens	2245		first 1000			2245

Table 2: Results of various systems: D^i $(d \leq i)$, i^{PH} (pre-head length is i), $+Markov$ (1^{st} order Markov conditioning on nodes with empty subcat frames for generating L and R T-grams).

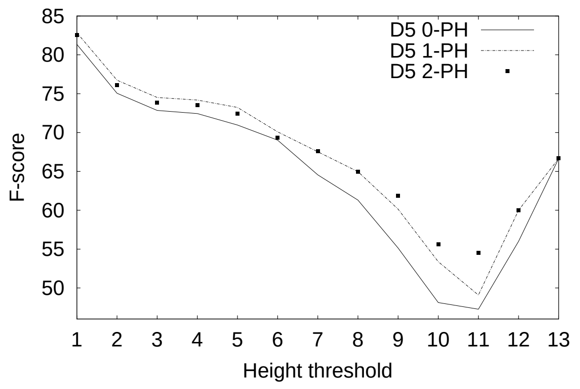

Figure 7: Heigher nodes are harder

Acknowledgements: I thank Remko Scha, Remo Bonnema, Walter Daelemans and Jakub Zavrel for illuminating discussions, Remko Bonnema for his pre- and post-parsing software, and the anonymous reviewers for their comments. This work is supported by the Netherlands Organization for Scientific Research.

References

E. Black et al. 1991. A procedure for quantitatively comparing the syntactic coverage of english grammars. In *Proceedings of the February 1991 DARPA Speech and Natural Language Workshop*.

R. Bod. 1995. *Enriching Linguistics with Statistics: Performance models of Natural Language*. PhD thesis, ILLC-dissertation series 1995-14, University of Amsterdam.

R. Bonnema, P. Buying, and R. Scha. 1999. A new probability model for data oriented parsing. In Paul Dekker and Gwen Kerdiles, editors, *Proceedings of the 12th Amsterdam Colloquium*, Amsterdam, The Netherlands, december. Institute for Logic, Language and Computation, Department of Philosophy.

E. Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the* 14^{th} *National Conference on Artificial Intelligence*, pages 598–603, Menlo Park. AAAI Press/MIT Press.

E. Charniak. 1999. A maximum-entropy-inspired parser. In *Report CS-99-12*, Providence, Rhode Island.

M. Collins. 1997. Three generative, lexicalized models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the ACL and the 8th Conference of the EACL*, pages 16–23, Madrid, Spain.

J. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING-96*, pages 340–245, Copenhagen, Denmark.

F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi, and S. Roukos. 1994. Decision tree parsing using a hidden derivation model. In *Proceedings of the 1994 Human Language Technology Workshop*. DARPA.

D. M. Magerman. 1995. Statistical Decision-Tree Models for Parsing. In *Proceedings of the 33th Annual Meeting of the ACL.*

M. Marcus, B. Santorini, and M. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2).

G. Satta. 2000. Parsing techniques for lexicalized context-free grammars. In *Proceedings of the* 6^{th} *IWPT*, Trento, Italy, Februari.

R. Scha. 1990. Language Theory and Language Technology; Competence and Performance (in Dutch). In Q.A.M. de Kort and G.L.J. Leerdam, editors, *Computertoepassingen in de Neerlandistiek*, Almere: LVVN-jaarboek.

K. Sima'an. 1996. Computational Complexity of Probabilistic Disambiguation by means of Tree Grammars. In *Proceedings of COLING'96*, volume 2, pages 1175–1180, Copenhagen, Denmark, August.