# Wide coverage incremental parsing by learning attachment preferences

Fabrizio Costa[1], Vincenzo Lombardo[2], Paolo Frasconi[1], and Giovanni Soda[1]

[1] DSI, Università di Firenze, ITALY
{costa, paolo, giovanni}@dsi.unifi.it
[2] DiSTA, Università del Piemonte Orientale, ITALY
vincenzo@di.unito.it

**Abstract.** This paper presents a novel method for wide coverage parsing using an incremental strategy, which is psycholinguistically motivated. A recursive neural network is trained on treebank data to learn first pass attachments, and is employed as a heuristic for guiding parsing decision. The parser is lexically blind and uses beam search to explore the space of plausible partial parses and returns the full analysis having highest probability. Results are based on preliminary tests on the WSJ section of the Penn treebank and suggest that our incremental strategy is a computationally viable approach to parsing.

## 1 Introduction

The most successful approaches to wide coverage parsing are the history-based algorithms, where the parse tree is viewed as a sequence of decisions (a derivation), and the probability of the tree is calculated by combining single decision probabilities. The resolution of structural ambiguity in probabilistic terms relies on a learning process over large text corpora annotated with syntactic structures (treebanks). History-based parsers are generally modeled upon probabilistic context-free grammars, and produce more accurate results if they learn about bilexical dependencies between head words of constituents [2, 3]. Though in general these approaches use specialized machine learning techniques, general learning frameworks are also applicable (ID3 algorithm [7], maximum entropy model [13]). The most common control structure is the chart-based (or dynamic programming) technique.

This paper explores the possibility that a psycholinguistically motivated parser can also perform well on freely occurring text. Our reference theories in the psycholinguistic literature are those theories that rely on learning parse decisions from the past experience (e.g., [12]). Specifically, we have developed an incremental history-based parsing model that relies on a dynamic grammar, and solves attachment ambiguities by using a general machine learning technique (recursive neural networks).

The incrementality hypothesis assumes that humans process language from left to right, and proceed by chunking partial parses that span the initial fragment of the sentence (we can call these partial parses *incremental trees*). Wide coverage parsing models are rarely based on the incrementality hypothesis. An exception to this claim is the work of [14], who have implemented a predictive parser, which selects the parses to expand with a probabilistic best-first method and a beam search algorithm. With the incremental strategy, the probabilistic model follows the chain of rule predictions that allow the linking of the next input word. A natural way to encode such linguistic knowledge is the dynamic grammar approach. A dynamic grammar characterizes the syntactic knowledge in terms of states, and transitions between states [11], thus forming a framework where competence and performance can be easily put in relation (derivation steps and parsing actions coincide). In our approach, the states of the dynamic grammar are the incremental trees, and the transitions between states are the attachment operations of partial structures that extend an incremental tree to include the next input word.

The probabilistic model that informs the parsing decisions relies on a learning technique that involves a neural network model, called *recursive neural network* [5]. Recursive neural networks can adaptively process labeled graphs, and exploit the supervised learning paradigm on structured data. The incremental trees are the instances in the learning domain, and the prediction task consists in estimating the probability that a given incremental tree is the correct one.

The paper is organized as follows: Section 2 introduces the basic notions of incremental processing, and illustrates the parsing strategy. In section 3 we formulate the learning task, and specialize the basic network architecture to solve the learning problem considered here. In section 4 we describe the experimental setup, and discuss the results.

## 2 Incremental parsing with connection paths

Incremental processing of natural language (incrementality for short) is a widely held hypothesis upon the parsing actions of the human language processor. According to incrementality, the semantic interpretation of some fragment of the sentence occurs as the scan of the input material proceeds from left to right. This incrementality hypothesis has received a large experimental support in the psycholinguistic community over the years: from the shadowing experiments [10], to the data about head-final language processing [1, 6, 8], to the eye-movement studies of visual object recognition [4]. In this section, we provide an operational definition of incrementality, which forms the core of the parsing process.

The operational account of the incrementality hypothesis we are going to pursue here is called *strong incrementality*, and is a parsimonious version of incrementality (see [15]). This proposal shares some commonalities with a number of psycholinguistic models [16, 17] and the broad coverage predictive parser described in [14]. Parsing proceeds from left to right through a sequence of incremental trees, each spanning one more word to the right. No input is stored in a disconnected state.

The connection of the next input word to the existing structure (what is called the *left context*) requires the construction of a substructure called the *connection path* (see Figure 1). In parsing naturally occurring text, this results in a high number of candidate connection paths, which yields a hard search problem.

Before describing the parsing algorithm, we provide some basic definitions. Given a sentence $s = w_0 w_1 \cdots w_i \cdots w_{|s|-1}$ and a tree $T$ for it, we define recursively the *incremental trees* $T_i (i = 0, 1, ..., |s| - 1)$ spanning $w_0 \cdots w_i$ as follows:

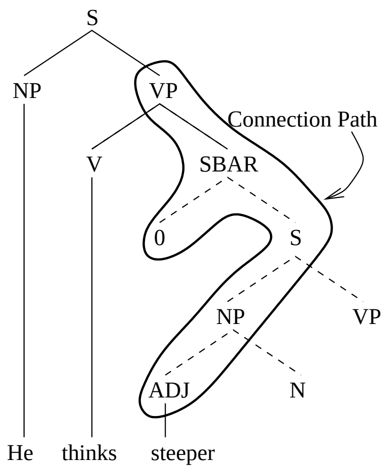

**Fig. 1.** The connection path for “steeper” in “He thinks steeper prices are to come.”

- $T_0$ consists of the node $w_0$;
- $T_i$ consists of all the nodes and edges in $T_{i-1}$ and the chain of nodes and edges from $w_i$ to $N$, where $N$ is
  - either a node of $T_{i-1}$,
  - or the lowest node of $T$ dominating both the root of $T_{i-1}$ and $w_i$ (in this case $T_i$ also includes the edge from N to the root of $T_{i-1}$).

Given two incremental trees $T_1$ and $T_2$, we define the *difference* between $T_1$ and $T_2$ as the tree formed by all the edges which are in $T_1$ and not in $T_2$, and all the nodes touched by such edges. Now, given a sentence $s = w_0 w_1 \cdots w_{|s|-1}$ and a tree $T$ for it, the *connection path* for $w_i$ is the difference between the incremental trees $T_i$ and $T_{i-1}$. Moreover,

- A node both in $T_i$ and in $T_{i-1}$, and touched by an edge only in $T_i$, is called an *anchor* (that is, a node where the connection path anchors to $T_{i-1}$).
- The node labeled by the POS tag of $w_i$ is called a *foot*.

In Figure 2a, we show the sequence of incremental trees for a sentence of the corpus. The notions introduced so far underlie a parsing schema that operates incrementally with a grammar composed of connection paths. The grammar of connection paths is a sort of *dynamic grammar* [11], that is a grammar that defines the linguistic knowledge in the form of possible transitions between parsing states.

In order to implement a wide coverage parser, we have collected a large basic grammar of connection paths. We have run a simulation algorithm on sections 2-21 of the Penn treebank (about 40,000 sentences). The simulation algorithm collects all the incremental trees associated with the trees in the sample, by simulating the building of the syntactic structure as it would be built by a perfectly informed incremental parser (a similar approach has been described in [9]). The data base (or universe) of connection paths extracted from the dataset

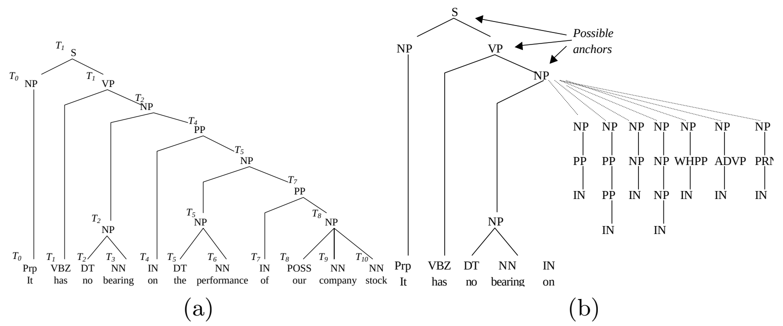

**Fig. 2.** (a) The incremental trees of the sentence "It has no bearing on the performance of our company stock." Nodes are labeled with the incremental tree that includes them for the first time. (b): Local ambiguity in incremental processing of the same sentence. The figure shows the incremental tree $T_3$, with its potential anchors and the connection paths compatible with one of the anchors (NP).

counts 5.348. The connection paths as defined here do not include any information beyond the structural commitments which are necessary to incorporate the new word in input (see Figures 1 and 2a). Linguistically informed connection paths would include further nodes required by constraints posed by the individual words. For example, in Figure 1 the node SBAR would be predicted by the sub-categorization frame of the word "thinks", and so it would not be part of the connection path for linking "steeper".

## 2.1 Parsing strategy

Let us consider the parsing search space $\mathcal{S}(s)$ associated with a given sentence $s = w_1, \ldots, w_n$. States in this space consist of incremental trees, and state transitions (search operators) correspond to legal attachments of connection paths to incremental trees. The initial state is the empty tree. Goals are all the states whose associated parse trees span the whole sentence under consideration. Clearly, the generic *parse* tree at depth $i$ of the *search* tree spans $w_1, \ldots, w_i$. The search space $\mathcal{S}(s)$ is a tree for every sentence $s$, and any incremental tree for $w_1, \ldots, w_i$ can be identified by the unique sequence of connection path attachments $j_1, j_2, \ldots, j_i$ in its derivation. We denote by $T_{j_1,\ldots,j_i}$ the resulting incremental tree. Note that only one derivation $j_1^*, \ldots, j_n^*$ leads to the correct tree for $w_1, \ldots, w_n$. Our method consists of defining a probability distribution over the space of possible derivations and then seeking the most likely derivation. The probability distribution over derivation is factorized using the psycholinguistic notion of preference in the first pass attachment, modeled as the conditional probability

$$P(T_{j_1,\ldots,j_i} | T_{j_1,\ldots,j_{i-1}}) \tag{1}$$

Given an incremental tree spanning $w_1, \ldots, w_{i-1}$, several candidate incremental trees can be built to accommodate the next word $w_i$. We denote by $F_{j_1,\ldots,j_{i-1}}$ the forest of candidate parse trees obtained by linking $w_i$ to $T_{j_1,\ldots,j_{i-1}}$. We assume this set of candidates is exhaustive and contains the correct incremental tree for $w_1, \dots w_i$. Hence:

$$\sum_{T_{j_1,\dots,j_i} \in F_{j_1,\dots,j_{i-1}}} P(T_{j_1,\dots,j_i} | T_{j_1,\dots,j_{i-1}}) = 1 \tag{2}$$

Since derivations are unambiguous, we can recursively apply equation 1, obtaining

$$P(T_{j_1,\dots,j_i}) = P(T_{j_1,\dots,j_i} | T_{j_1,\dots,j_{i-1}}) P(T_{j_1,\dots,j_{i-1}}). \tag{3}$$

where in the base step we assign probability one to the empty tree spanning the empty sentence. As a result, the probability of a full parse $T_{j_1,\dots,j_n}$ is

$$P(T_{j_1,\dots,j_n}) = \prod_{i=1}^{n} P(T_{j_1,\dots,j_i} | T_{j_1,\dots,j_{i-1}}). \tag{4}$$

Conditional probabilities in Eq. 1 are empirically estimated from a treebank and modeled by a recursive neural network, a machine learning architecture that can solve the supervised learning problem when instances are represented by labeled acyclic graphs [5]. The algorithm for learning first pass attachments is outlined in the next section.

Finding the most likely analysis in this framework is computationally expensive due to the large size of the state space. We found that the average branching factor (corresponding to the number of first pass attachment alternatives) is about 132 in the set of 40k sentences mentioned above. Since exhaustive search is computationally unfeasible, a beam search algorithm is employed for selecting a pool of candidate analyses receiving high probability according to the above model. The outer loop of the algorithm sweeps the sentence word by word, from left to right. At each stage $i$, we create a beam of $B_i$ best candidates by expanding and scoring each candidate selected in the previous stage. Branches of search space with low probability incremental trees are thus pruned, based on the rationale that they are unlikely to appear as the left fragment of the most likely full analysis.

## 3 Formulating parse decisions with a neural network

Ambiguity comes in two forms: the number of possible anchors on the right edge of $T_{i-1}$, and the number of different paths that can link $w_i$ with $T_{i-1}$ from some anchor (see Figure 2b). A selection procedure chooses the best connection path and anchor for continuation, and instantiates it to generate the new incremental tree $T_i$. Now we first formulate the learning task, and then we illustrate the network architecture.

### 3.1 The learning task

The instances of the learning domain are the incremental trees. We start from a core corpus of parsed sentences, which is denoted as $\boldsymbol{B} = \{(s^{(p)}, T(s^{(p)})), p = 1, \cdots, P\}$ where $s^{(p)}$ is a generic sentence and $T(s^{(p)})$ its parse tree. The *universe* *of connection paths* $U(\boldsymbol{B})$ is the set of all connection paths that can be extracted from $T(s^{(p)}), p = 1, \cdots, P$ by running the simulation algorithm described above. The universe $U(\boldsymbol{B})$ effectively plays the role of a dynamic grammar for the sentences in $\boldsymbol{B}$.

Given a sentence $s = w_0, \ldots, w_{|s|-1}$ (not in the corpus $\boldsymbol{B}$), at stage $i$ of parsing we know the correct incremental tree $T_{i-1}(s)$ spanning $w_0, \ldots, w_{i-1}$. The goal of an incremental parser is then to compute the next tree $T_i(s)$ in order to accommodate the next word $w_i$. $T_i(s)$ can be obtained by joining $T_{i-1}(s)$ to one of the connection paths in $U(\boldsymbol{B})$. However, other trees spanning $w_1, \cdots, w_i$ can be generated by legally attaching other connection paths. The set of trees obtained by legally attaching $T_{i-1}(s)$ to any path in $U(\boldsymbol{B})$ is called the *forest of candidates* for word $w_i$ within sentence $s$, denoted $\boldsymbol{F}_i(s) = \{T_{i,1}(s), \ldots, T_{i,m_i}(s)\}$. Of course, only one of the trees in $\boldsymbol{F}_i(s)$, $T_{i,j^\star}(s)$, is the correct one (and maybe none, in case of incompleteness of the grammar).

The learning algorithm relies on a statistical model that assigns probabilities of correctness to each candidate tree. Parameters of the model are estimated from examples. Then the model can be effectively employed to rank alternative trees, sorting them by increasing probability of correctness.

The learning task is formulated as follows: Each instance is the whole forest of candidate trees $\boldsymbol{F}_i(s)$. The task consists of learning the correct member of the forest, which can be identified by a multinomial variable with realizations in $\{1, \ldots, m_i(s)\}$. Training examples are pairs $(\boldsymbol{F}_i(s), j_i^\star(s))$, where the input portion is the candidate forest and the output portion (supervision) is the integer $j_i^\star(s) \in [1, m_i(s)]$ identifying the correct tree.

One learning instance is a "bag" of objects (trees), which contains exactly one positive instance, and such instance can be always identified in the training set. In a probabilistic setting this means that the learner should make decisions about *multinomial* output variables $O_i(s)$ whose realizations are integers in $[1, \cdots, m_i(s)]$ identifying the correct tree in the bag associated with word $w_i$ in sentence $s$. In particular, for each position $i$ within $s$ and for each alternative $j$, the learner should predict the quantity

$$y_{i,j}(s) = P(O_i(s) = j|\boldsymbol{F}_i(s)) \tag{5}$$

where $O_i(s) = j$ means that $T_{i,j}(s)$ is the correct tree. Predictions are conditional to the whole set of candidates thus introducing competition among trees in the forest. In the following subsection we shall explain how this learning task can be solved using a connectionist approach. To simplify notation, reference to the particular sentence $s$ will be omitted in the following discussion.

### 3.2 Neural network architecture

We use an architecture, called *recursive* neural network, which is suitable for dealing with labeled directed acyclic graphs [5]. The input $I$ in the present case is a labeled ordered forest of $m$-ary trees, where labeled means that each vertex has a label from a finite alphabet $\mathcal{I} = I_1, \ldots, I_N$ (namely, the current set of non-terminal symbols in the universe of connection paths), and ordered means that for each vertex $v$, a total order is defined on the $m$ children of $v$ ($ch[v]$ denotes the ordered $m$-tuple of vertices that are $v$'s children; $I(v)$ denotes the label attached to vertex $v$). The basic network computation is based on the following recursive state space representation:

$$\begin{array}{l} \boldsymbol{x}(v) = f(\boldsymbol{x}(ch[v]), I(v)) \\ a = g(\boldsymbol{x}(r)). \end{array} \tag{6}$$

$\boldsymbol{x}(v) \in I\!R^n$ denotes the state vector associated with node $v$; $\boldsymbol{x}(\mathrm{ch}[v]) \in I\!R^{m \cdot n}$ is a vector obtained by concatenating the components of the state vectors contained in $v$'s children; $f : \mathcal{I} \times I\!R^{m \cdot n} \to I\!R^n$ is the state transition function that maps states at $v$'s children and the label at $v$ into the state vector at $v$; $g : I\!R^n \to I\!R$ is the output function, that maps the state $\boldsymbol{x}(r)$ (at the root $r$ of the input tree) into a real number $a$. States in Eq. (6) are updated bottom-up, traversing the tree in post-order. If a child is missing, the corresponding entries in $\boldsymbol{x}(\mathrm{ch}[v])$ are filled with the *frontier* state $\overline{\boldsymbol{x}}$, which is associated with the base step of recursion. Functions $f$ and $g$ are implemented by two multilayered perceptrons. After all $m_i(s)$ trees in the forest $\boldsymbol{F}_i(s)$ have been processed by the same network, we obtain a vector of $m_i(s)$ real outputs $a_{i,j}(s)$. These real numbers are eventually passed through the softmax function (normalized exponentials) to obtain the probabilities that will be used to rank trees in the forest.

Supervised learning is based on the maximum likelihood principle as in many other neural network models. The cost function has the form of a cross-entropy (negative log-likelihood according to the multinomial model) and is written as follows:

$$J(\mathcal{D}) = -\sum_{p=1}^{P} \sum_{i=0}^{|s^{(p)}|-1} \log y_{i,j*} \tag{7}$$

where $\mathcal{D}$ denotes the training set, the first sum ranges over sentences in the training set, the second sum ranges over words within each sentence, and $j^\star$ is the index of the correct incremental tree in the candidate list (which is known in the training set). Optimization is solved by gradient descent. In this case, gradients are computed by a special form of backpropagation on the feedforward network obtained by unrolling the state transition network according to the topology of the input graph $I$ [5].

## 4 Implementation and results

In this section, we illustrate the experiments we carried out to demonstrate the viability of the method. We have carried on two experiments to evaluate two different subcomponents of our systems. The first experiment has been the verification that the network can generalize appropriately over incremental trees in order to predict the correct connection path for the expansion. In the second experiment we have combined neural network predictions with the simple search strategy described above.

## 4.1 Dataset

The reference grammar is the data base of 5,348 connection paths extracted from the sections 2-21 of the WSJ Collection of the Penn II Treebank (39,831 sentences, about 1 million words). For the training set section 2-21 was used (about 40,000 sentences), for the validation set section 24 (about 3,500 sentences), and for the test set section 23 (about 2,500 sentences). Each sentence of the three sets was processed in the following manner. For each word $w_i$, we have examined a forest of candidate incremental trees $\boldsymbol{F}_i$, formed by identifying the possible anchor nodes on the right frontier of $T_{i-1}$ and the foot category of $w_i$, and composing $T_{i-1}$ with all the connection paths in the universe $U(\boldsymbol{B})$ that are compatible with the anchor category and the foot category (see Figure 2b) [1]. we do not consider individual words and semantic affixes of non-terminal labels. For example, both NP-PRD and NP-MNR have been collapsed to NP. The total number of non-terminal symbols is 72.

In our runs, the average cardinality of a candidate set $\boldsymbol{F}_i$ is 133, and the maximum cardinality is 2542. This gives a precise idea of the amount of ambiguity that occurs for the connection of each word. In the training set (40k sentences), the total number of words is about 1 million and the total number of candidates is 126 millions. In the test set (2416 sentences), the total number of words is 51,433 and the total number of candidates is 7 millions.

The recursive network used for the experiments implements the model described in Section 3.2, and has the following specifications: $m = 15$ (maximum outdegree), $N = 72$ labels (one-hot encoded), and $n = 20$ state components, yielding a total of 7,440 adjustable parameters. Each tree in the forest is processed separately by a recursive network with one linear output unit. The resulting numbers are then normalized with the softmax function. All the trees (both the correct and the incorrect ones) are grouped into 1 million forests and are employed to train the network in each epoch.

In all the experiments, we estimated generalization (the average position $R$ of the correct candidate) by running the partially trained networks in recall mode on the 3,676 validation sentences. Gradient descent was stopped when the performance on the validation set was maximum. This early stopping is expected to be useful in order to prevent overfitting. Due to the redundancy in our large training set, 4 epochs were sufficient.

## 4.2 Results on learning first pass attachments

The output of the trained network is employed to rank candidate trees by increasing probability of correctness. Hence, the performance measure of our approach to parse decisions is based on the rank position assigned to the correct tree in each forest $\boldsymbol{F}_i$.

A synthesis of results on the 2,416 test sentences is reported in Table 1. Each time the parser tries to attach a word, it must select a connection path (actually

[1] Notice that the model does not make use of lexical and semantic information, which are considered very relevant in wide coverage parsing.

**Table 1.** Performance of the network in positioning the correct incremental tree.

| Bin | R | 1 pos % | Num | Bin | R | 1 pos % | Num |
|---|---|---|---|---|---|---|---|
| 2 – 7 | 1.18 | 88% | 2727 | 90 – 103 | 1.56 | 84% | 2885 |
| 8 – 14 | 1.22 | 90% | 2738 | 104 – 123 | 1.69 | 83% | 2974 |
| 15 – 20 | 1.27 | 88% | 2734 | 124 – 145 | 1.61 | 83% | 2909 |
| 21 – 29 | 1.27 | 88% | 2913 | 146 – 176 | 1.68 | 82% | 2995 |
| 30 – 39 | 1.57 | 84% | 2895 | 177 – 207 | 1.93 | 79% | 2984 |
| 40 – 49 | 1.43 | 84% | 2974 | 208 – 255 | 1.82 | 81% | 2969 |
| 50 – 61 | 1.51 | 85% | 2845 | 256 – 330 | 1.86 | 82% | 2998 |
| 62 – 75 | 1.70 | 79% | 2790 | 331 – 476 | 2.26 | 77% | 2989 |
| 76 – 89 | 1.64 | 82% | 2810 | 477 – 2542 | 3.74 | 67% | 2304 |

an incremental tree) out of a certain number that can range from 1 (unique selection) to 2542. In order to rate the performance of the network in selection, we have gathered the possible cases in bins: so, the first row includes the cases which require the selection of one connection path out of two, three, four and up to seven, respectively; the second row includes the cases which require the selection of one connection path out of eight to fourteen, respectively; etc. The actual bins originate from the effort of balancing their cardinality (around 3000, see the last column). $R$ is the average position of the correct incremental tree, after having sorted candidates according to network predictions. $R = 1$ would yield a perfect predictor that always ranks the correct tree in first position. The third column reports the percentage of times that the correct incremental tree was in the first position. The fourth column reports the cardinality of the bin. Globally, 82.67% of the trees are correctly assigned in first position and the global average position is 1.70.

## 4.3 Preliminary results on incremental parsing

We have run the parser on the 2,416 test sentences with a beam of 100. We are perfectly aware that this number is very low compared with the base beam factor (10,000) used by Roark and Johnson. 97% of the sentences were parsed (that is, reached the end)[2]. The performance of the parser on the 2,416 test sentences was 64.89% labeled precision and 57.84% labeled recall with an average crossing of 2.76 (that becomes precision/recall 70.27/62.62 if we consider only sentences shorter than 30 words). Of course these preliminary results still do not compete against the level of performance of many current history-based parsing systems (precision/recall 76%/76% and average crossing of 2.26 for parsers that operate only with POS tags, without lexical information). However, we believe that the results are encouraging, and we are currently investigating how to incorporate lexical dependency information and grammatical transformations.

[2] The fact that 3% were not parsed is to be accounted to the incapability of the recursive neural network to process trees containing nodes with outdegree greater than the maximum specified (in our case 15).

## 5 Conclusions

The paper has presented a novel methodology for parsing unrestricted text based upon the incrementality hypothesis. The results are quite preliminary. The method currently takes into account only the syntactic structure labeled with non terminal categories. The insertion of a number of linguistic heuristics should improve the performances presented here.

## References

1. M. Bader and I. Lasser. German verb-final clauses and sentence processing. In C. Clifton, L. Frazier, and K. Reyner, editors, *Perspectives on Sentence Processing*, pages –. Lawrence Erlbaum Associates, 1994.
2. E. Charniak. Statistical parsing with a context-free grammar and word statistics. In *Proc. of AAAI97*, 1997.
3. M. Collins. A new statistical parser based on bigram lexical dependencies. In *Proc. of 34th ACL*, pages 184–191, 1996.
4. K. M. Eberhard, M. J. Spivey-Knowlton, J.C. Sedivy, and M. K. Tanenhaus. Eye movements as a window into real-time spoken language comprehension in natural contexts. *Journal of Psycholinguistic Research*, 24:409–436, 1995.
5. P. Frasconi, M. Gori, and A. Sperduti. A general framework for adaptive processing of data structures. *IEEE Trans. on Neural Networks*, 9(5):768–786, 1998.
6. L. Frazier. Syntactic processing: Evidence from dutch. *Natural Language and Linguistic Theory*, 5:519–559, 1987.
7. U. Hermjakob and R. J. Mooney. Learning parse and translation decisions from examples with rich context. In *Proceedings of ACL97*, pages 482–489, 1997.
8. Y. Kamide and D. C. Mitchell. Incremental pre-head attachment in japanese parsing. *Language and Cognitive Processes*, 14(5–6):631–662, 1999.
9. V. Lombardo and P. Sturt. Incrementality and lexicalism: A treebank study. In S. Stevenson and P. Merlo, editors, *Lexical Representations in Sentence Processing*. John Benjamins, 1999.
10. W. Marslen-Wilson. Linguistic structure and speech shadowing at very short latencies. *Nature*, 244:522–533, 1973.
11. D. Milward. Dynamic dependency grammar. *Linguistics and Philosophy*, 17(6), 1994.
12. D.C. Mitchell, F. Cuetos, M.M.B. Corley, and M. Brysbaert. Exposure-based models of human parsing: evidence for the use of coarse-grained (nonlexical) statistical records. *Journal of Psycholinguistics Research*, 24, 1995.
13. A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of EMNLP97*, 1997.
14. B. Roark and M. Johnson. Efficient probabilistic top-down and left-corner parsing. In *Proc. of ACL99*, 1999.
15. E. P. Stabler. The finite connectivity of linguistic structure. In C. Clifton, L. Frazier, and K. Reyner, editors, *Perspectives on Sentence Processing*, pages 303–336. Lawrence Erlbaum Associates, 1994.
16. M. J. Steedman. Grammar, interpretation and processing from the lexicon. In W. M. Marslen-Wilson, editor, *Lexical Representation and Process*, pages 463–504. MIT Press, 1989.
17. P. Sturt and M. Crocker. Monotonic syntactic processing: a cross-linguistic study of attachment and reanalysis. *Language and Cognitive Processes*, 11(5):449–494, 1996.