# A Stochastic Parser
# Based on an SLM with Arboreal Context Trees

**Shinsuke MORI**

IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.
1623-14 Shimotsuruma Yamato-shi, 242-8502, Japan
mori@trl.ibm.co.jp

## Abstract

In this paper, we present a parser based on a stochastic structured language model (SLM) with a flexible history reference mechanism. An SLM is an alternative to an $n$-gram model as a language model for a speech recognizer. The advantage of an SLM against an $n$-gram model is the ability to return the structure of a given sentence. Thus SLMs are expected to play an important part in spoken language understanding systems. The current SLMs refer to a fixed part of the history for prediction just like an $n$-gram model. We introduce a flexible history reference mechanism called an ACT (arboreal context tree; an extension of the context tree to tree-shaped histories) and describe a parser based on an SLM with ACTs. In the experiment, we built an SLM-based parser with a fixed history and one with ACTs, and compared their parsing accuracies. The accuracy of our parser was 92.8%, which was higher than that for the parser with the fixed history (89.8%). This result shows that the flexible history reference mechanism improves the parsing ability of an SLM, which has great importance for language understanding.

## 1 Introduction

Currently, the state-of-the-art speech recognizers can take dictation with satisfactory accuracy. Although continuing attempts for improvements in predictive power are needed in the language modeling area for speech recognizers, another research topic, understanding of the dictation results, is coming into focus. Structured language models (SLMs) (Chelba and Jelinek, 1998; Charniak, 2001; Mori et al., 2001) were proposed for these purposes. Their predictive powers are reported to be slightly higher than an orthodox word tri-gram model if the SLMs are interpolated with a word tri-gram model. In contrast with word $n$-gram models, SLMs use the syntactic structure (a partial parse tree) covering the preceding words at each step of word prediction. The syntactic structure also grows in parallel with the word prediction. Thus after the prediction of the last word of a sentence, SLMs are able to give syntactic structures covering all the words of an input sentence (parse trees) with associated probabilities. Though the impact on the predictive power is not major, this ability, which is indispensable to spoken language understanding, is a clear advantage of SLMs over word $n$-gram models. With an SLM as a language model, a speech recognizer is able to directly output a recognition result with its syntactic structure after being given a sequence of acoustic signals.

The early SLMs refer to only a limited and fixed part of the histories for each step of word and structure prediction in order to avoid a data-sparseness problem. For example, in an English model (Chelba and Jelinek, 2000) the next word is predicted from the two right-most exposed heads. Also in a Japanese model (Mori et al., 2000) the next word is predicted from 1) all exposed heads depending on the next word and 2) the words depending on those exposed heads. One of the natural improvements in predictive power for an SLM can be achieved by adding some flexibility to the history reference mechanism. For a linear history, which is referred to by using word $n$-gram models, we can use a context tree (Ron et al., 1996) as a flexible history reference mechanism. In an $n$-gram model with a context tree, the length of each $n$-gram is increased selectively according to an estimate of the resulting improvement in predictive quality. Thus, in general, an $n$-gram model with a context tree has more predictive power in a smaller model.

In SLMs, the history is not a simple word sequence but a sequence of partial parse trees. For a tree-shaped context, there is also a flexible history reference mechanism called an arboreal context tree (ACT) (Mori et al., 2001).[1] Similar to a context tree, an SLM with ACTs selects, depending on the context, the region of the tree-shaped history to be referred to for the next word prediction and the next structure prediction. Mori et al. (2001) report that an SLM with ACTs has more predictive power than an SLM with a fixed history reference mechanism. Therefore, if a parser based on an SLM with ACTs outperforms an SLM without ACTs, an SLM with

[1] In the original paper, it was called an arbori-context tree.

ACTs is a promising language model as the next research milestone for spoken language understanding systems.

In this paper, first we describe an SLM with ACTs for a Japanese dependency grammar. Next, we present our stochastic parser based on the SLM. Finally, we report two experimental results: a comparison with an SLM without ACTs and another comparison with a state-of-the-art Japanese dependency parser. The parameters of our parser were estimated from 9,108 syntactically annotated sentences from a financial newspaper. We then tested the parser on 1,011 sentences from the same newspaper. The accuracy of the dependency relationships reported by our parser was 92.8%, higher than the accuracy of the parser based on an SLM without ACTs (89.8%). This proved experimentally that an ACT improves a parser based on an SLM.

## 2 Structured Language Model based on Dependency

The most popular language model for a speech recognizer is a word $n$-gram model, in which each word is predicted from the last $(n-1)$ words. This model works so well that the current recognizer can take dictation with an almost satisfactory accuracy. Now the research focus in the language model area is understanding the dictation results. In this situation, a structured language model (SLM) was proposed by Chelba and Jelinek (1998). In this section, we describe the dependency grammar version of an SLM.

### 2.1 Structured Language Model

The basic idea of an SLM is that each word would be better predicted from the words that may have a dependency relationship with the word to be predicted than from the proceeding $(n-1)$ words. Thus the probability $P$ of a sentence $\boldsymbol{w} = w_1 w_2 \cdots w_n$ and its parse tree $T$ is given as follows:

$$P(T) = \prod_{i=1}^{n} P(w_i|\boldsymbol{t}_{i-1}) P(\boldsymbol{t}_i|w_i, \boldsymbol{t}_{i-1}), \qquad (1)$$

where $\boldsymbol{t}_i$ is the $i$-th partial parse tree sequence. The partial parse tree depicted at the top of Figure 1 shows the status before the 9th word is predicted. From this status, for example, first the 9th word $w_9$ is predicted from the 8th partial parse tree sequence $\boldsymbol{t}_8 = t_{8,3} t_{8,2} t_{8,1}$, and then the 9th partial parse tree sequence $\boldsymbol{t}_9$ is predicted from the 9th word $w_9$ and the 8th partial parse tree sequence $\boldsymbol{t}_8$ to get ready for the 10th word prediction. The problem here is how to classify the conditional parts of the two conditional probabilities in Equation (1) in order to predict the next word and the next structure without encountering a data-sparseness problem. In an English model (Chelba and Jelinek, 2000) the next word is predicted from the two right-most exposed heads (for example $w_6$ and $w_8$ in Figure 1) as follows:

$$P(w_i|\boldsymbol{t}_{i-1}) \approx P(w_i|\mathit{root}(t_{i-1,2}), \mathit{root}(t_{i-1,1})),$$

where $\mathit{root}(t)$ is a function returning the root label word of the tree $t$. A similar approximation is adapted to the probability function for structure prediction. In a Japanese model (Mori et al., 2000) the next word is predicted from 1) all exposed heads depending on the next word and 2) the words depending on those exposed heads.

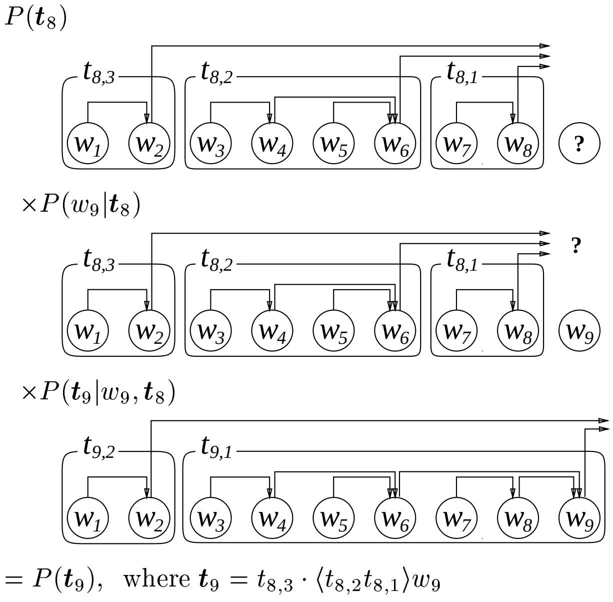
$P(\boldsymbol{t}_8)$
$t_{8,3}$
$t_{8,2}$
$t_{8,1}$
$w_1$ $w_2$ $w_3$ $w_4$ $w_5$ $w_6$ $w_7$ $w_8$ ?
$\times P(w_9|\boldsymbol{t}_8)$
?
$t_{8,3}$
$t_{8,2}$
$t_{8,1}$
$w_1$ $w_2$ $w_3$ $w_4$ $w_5$ $w_6$ $w_7$ $w_8$ $w_9$
$\times P(\boldsymbol{t}_9|w_9, \boldsymbol{t}_8)$
$t_{9,2}$
$t_{9,1}$
$w_1$ $w_2$ $w_3$ $w_4$ $w_5$ $w_6$ $w_7$ $w_8$ $w_9$
$= P(\boldsymbol{t}_9)$, where $\boldsymbol{t}_9 = t_{8,3} \cdot \langle t_{8,2} t_{8,1} \rangle w_9$

Figure 1: Word prediction from a partial parse

It is clear, however, that in some cases some child nodes of the tree $t_{i-1,2}$ or $t_{i-1,1}$ are useful for the next word prediction and in other cases even the consideration of an exposed head (root of the tree $t_{i-1,1}$ or $t_{i-1,2}$) suffers from a data-sparseness problem because of the limitation of the learning corpus size. Therefore a more flexible mechanism for history classification should improve the predictive power of the SLM.

### 2.2 SLM for Dependency Grammar

Since in a dependency grammar of Japanese, every dependency relationship is in a unique direction as shown in Figure 1 and since no two dependency relationships cross each other, the structure prediction model only has to predict the number of trees. Thus, the second conditional probability in the right hand side of Equation (1) is rewritten as $P(l_i|w_i, \boldsymbol{t}_{i-1})$, where $l_i$ is the length (number of elements) of the tree sequence $\boldsymbol{t}_i$. Our SLM for the Japanese depen-

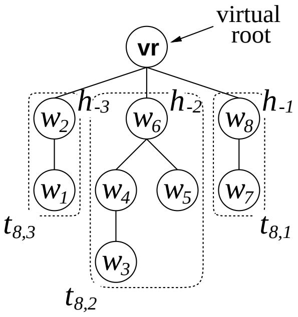

Figure 2: A history tree.

dency grammar is defined as follows:

$$P(T) = \prod_{i=1}^{n} P(w_i|\boldsymbol{t}_{i-1})P(l_i|w_i, \boldsymbol{t}_{i-1}). \qquad (2)$$

According to a psycholinguistic report on language structure (Yngve, 1960), there is an upper limit on $l_i$, the number of words whose modificands have not appeared yet. We set the upper limit to 9, the maximum number of slots in human short-term memory (Miller, 1956). With this limitation, our SLM becomes a hidden Markov model.

## 3 Arboreal Context Tree

A variable memory length Markov model (Ron et al., 1996), a natural extension of the $n$-gram model, is a flexible mechanism for a linear context (word sequence) which selects, depending on the context, the length of the history to be referred to for the next word prediction. This model is represented by a suffix tree, called a context tree, whose nodes are labeled with a suffix of the context. In this model, the length of each $n$-gram is increased selectively according to an estimate of the resulting improvement in predictive quality.

In SLMs, the history is not a simple word sequence but a sequence of partial parse trees. For a tree-shaped context, there is also a flexible history reference mechanism called an arboreal context tree (ACT) (Mori et al., 2001) which selects, depending on the context, the region of the tree-shaped history to be referred to for the next word prediction and for the next structure prediction. In this section, we explain ACTs and their application to SLMs.

### 3.1 Data Structure

As we mentioned above, in SLMs the history is a sequence of partial parse trees. This can be regarded as a single tree, called a history tree, by adding a virtual root node having these partial trees under it. For example, Figure 2 shows the history tree for the 9th word prediction based on the status depicted at the top of Figure 1. An arboreal context tree is a data structure for flexible history tree classification. Each node of an ACT is labeled with a subtree of the history tree. The label of the root is a null tree and if a node has child nodes, their labels are the series of trees made by expanding a leaf of the tree labeling the parent node. For example, each child node of the root in Figure 3 is labeled with a tree produced by adding the right-most child to the label of the root. Each node of an ACT has a probability distribution $P(x|t)$, where $x$ is an symbol and $t$ is the label of the node. For example, let $\langle a_k \cdots a_2 a_1 \rangle a_0$ represent a tree consisting of the root labeled with $a_0$ and $k$ child nodes labeled with $a_k, \cdots, a_2$, and $a_1$, so the right-most node at the bottom of the ACT in Figure 3 has a probability distribution of the symbol $x$ under the condition that the history matches the partial parse trees $\langle\langle z?\rangle a\rangle\langle b\rangle$, where "?" matches with an arbitrary symbol. Putting it in another way, the next word is predicted from the history having $b$ as the head of the right-most partial parse tree, $a$ as the head of the second right-most partial parse tree, and $z$ as the second right-most child of the second right-most partial parse tree. For example, in Figure 2 the subtree consisting of $w_4$, $w_6$, and $w_8$ is referred to for the prediction of the 9th word $w_9$ in Figure 1 under the following set of conditions: $a = w_6$, $b = w_8$, and $z = w_4$.

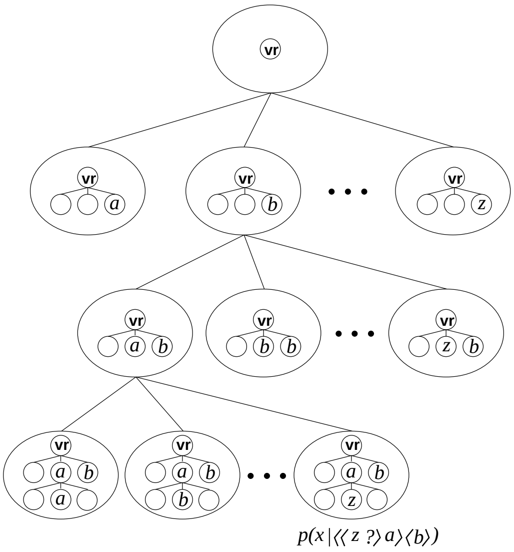

Figure 3: An arboreal context tree (ACT).

### 3.2 An SLM with ACTs

An ACT is applied to a classification of the condition parts of both of the two conditional probabilities in

Equation (2). Thus, an SLM with ACTs is:

$$P(T) = \prod_{i=1}^{n} P(w_i | ACT_w(\langle \boldsymbol{t}_{i-1} \rangle)) \times P(l_i | ACT_s(\langle \boldsymbol{t}_{i-1} w_i \rangle)), \quad (3)$$

where $ACT_w$ is an ACT for word prediction and $ACT_s$ is an ACT for structure prediction. Note that this is a generalization of the prediction from the two right-most exposed heads ($w_6$ and $w_8$) in the English model (Chelba and Jelinek, 2000). In general, SLMs with ACTs includes SLMs with fixed history reference mechanisms as special cases.

## 4 Parser

In this section, we explain our parser based on the SLM with ACTs we described in Sections 2 and 3.

### 4.1 Stochastic Parser Based on an SLM

A syntactic analyzer, based on a stochastic language model, calculates the parse tree with the highest probability $\hat{T}$ for a given sequence of words $\boldsymbol{w}$ according to

$$\begin{aligned}
\hat{T} &= \mathop{\textbf{argmax}}_{T} P(T|\boldsymbol{w}) \\
&= \mathop{\textbf{argmax}}_{T} P(T|\boldsymbol{w})P(\boldsymbol{w}) \\
&= \mathop{\textbf{argmax}}_{T} P(\boldsymbol{w}|T)P(T) \quad (\because \text{Bayes' formula}) \\
&= \mathop{\textbf{argmax}}_{T} P(T) \quad (\because P(\boldsymbol{w}|T) = 1),
\end{aligned}$$

where the concatenation of the words in the syntactic tree $T$ is equal to $\boldsymbol{w}$. $P(T)$ is an SLM. In our parser, $P(T)$ is the probability of a parse tree $T$ defined by the SLM based on the dependency with the ACTs (see Equation (3)).

### 4.2 Solution Search Algorithm

As shown in Equation (3), our parser is based on a hidden Markov model. It follows that the Viterbi algorithm is applicable to search for the best solution. The Viterbi algorithm is capable of finding the best solution in $O(n)$ time, where $n$ is the number of input words.

The parser repeats state transitions, reading words of the input sentence from beginning to end. So that the structure of the input sentence will be a single parse tree, the number of trees in the final state $\boldsymbol{t}_n$ must be 1 ($l_n = 1$). Among the final possible states that satisfy this constraint, the parser selects the state with the highest probability. Since our language model uses only a limited part of a partial parse tree to distinguish among states, the final state does not contain enough information to construct the parse tree. The parser can, however, calculate the parse tree from the sequence of states, or from the combination of the word sequence and the sequence of $l_i$, the number of words whose modificands have not appeared yet. Therefore our parser records these values at each prediction step. After the most probable last state has been selected, the parser constructs the parse tree by reading these sequences from beginning to end.

Table 1: Corpus.

| | #sentences | #words | #chars |
|---|---|---|---|
| learning | 9,108 | 260,054 | 400,318 |
| test | 1,011 | 28,825 | 44,667 |

Table 2: Word-based parsing accuracy.

| language model | parsing accuracy |
|---|---|
| SLM with ACTs | 92.8% (24,867/26,803) |
| SLM with fixed history | 89.8% (24,060/26,803) |
| baseline* | 79.4% (21,278/26,803) |

* Each word depends on the next one.

## 5 Evaluation

We developed an SLM with a constant history reference (Mori et al., 2000) and one with ACTs as explained in Section 3, and then implemented SLM-based parsers using the solution search algorithm presented in Section 4. In this section, we report the results of the parsing experiments and discuss them.

### 5.1 Conditions on the Experiments

The corpus used in our experiments consisted of articles extracted from a financial newspaper (*Nihon Keizai Shinbun*). Each sentence in the articles is segmented into words and each word is annotated with a part-of-speech (POS) and the word it depends on. There are 16 basic POSs in this corpus. Table 1 shows the corpus size. The corpus was divided into ten parts, and the parameters of the model were estimated from nine of them (learning) and the model was tested on the remaining one (test).

In parameter estimation and parsing, the SLM with ACTs distinguishes lexicons of function words (4 POSs) and ignores lexicons of content words (12 POSs) in order to avoid the data-sparseness problem. As a result, the alphabet of the SLM with ACTs consists of 192 function words, 4 symbols for unknown function words, and 12 symbols for content words. The SLM of the constant history reference selects words to be lexicalized referring to the accuracy of a withheld corpus (a small portion of the learning corpus).

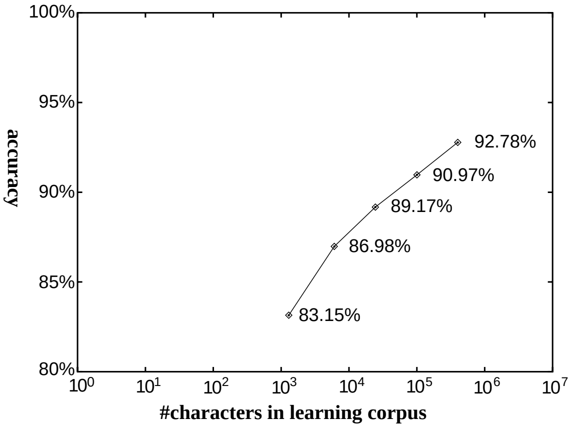

Figure 4: Relation between corpus size and parsing accuracy.

### 5.2 Evaluation

One of the major criteria for a dependency parser is the accuracy of its output dependency relationships. For this criterion, the input of a parser is a sequence of words, each annotated with a POS. The accuracy is the ratio of correct dependencies (matches in the corpus) to the number of the words in the input:

$$
\begin{aligned}
&accuracy \\
&= \frac{\#\text{words depending on the correct word}}{\#\text{words}}.
\end{aligned}
$$

The last word and the second-to-last word of a sentence are excluded, because there is no ambiguity. The last word has no word to depend on and the second-to-last word always depends on the last word.

Table 2 shows the accuracies of the SLM with ACTs, the SLM of the constant history reference, and a baseline in which each word depends on the next one. This result shows that the variable history reference mechanism based on ACTs reduces 30% of the errors of the SLM of a constant history reference. This proves experimentally that ACTs improve an SLM for use as a spoken language understanding engine.

We calculated the parsing accuracy of the models whose parameters were estimated from 1/4, 1/16, and 1/64 of the learning corpus and plotted them in Figure 4. The gradient of the accuracy curve at the point of the maximum learning corpus size is still important. It suggests that an accuracy of 95% should be achieved by annotating about 30,000 sentences.

Similar to most of the parsers for many languages, our parser is based on words. However, most other parsers for Japanese are based on a unique phrasal unit called a *bunsetsu*, a concatenation of one or more content words followed by some grammatical function words. In order to compare our parser with one of the state-of-the-art parsers, we calculated the *bunsetsu*-based accuracies of our model and KNP (Kurohashi and Nagao, 1994) on the first 100 sentences of the test corpus. First the sentences were segmented into words by JUMAN (Kurohashi et al., 1994) and the output word sequences are parsed by KNP. Next, the word-based dependencies output by our parser were changed into *bunsetsu* as used by KNP, where the *bunsetsu* which is depended upon by a *bunsetsu* is defined as the *bunsetsu* containing the word depended upon by the last word of the source *bunsetsu* (see Figure 5). Table 3 shows the *bunsetsu*-based accuracies of our model and KNP. In accuracy, our parser outperformed KNP, but the difference was not statistically significant. In addition, there were differences in the experimental environment:

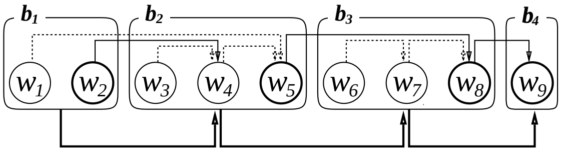

Figure 5: Conversion from word dependencies to *bunsetsu* dependencies.

Table 3: *Bunsetsu*-based parsing accuracy.

| language model | parsing accuracy |
|---|---|
| SLM with ACTs | 87.8% (674/768) |
| JUMAN+KNP | 85.3% (655/768) |
| baseline* | 62.4% (479/768) |

* Each *bunsetsu* depends on the next one.

- The test corpus size was limited.
- The POS system for the KNP input is detailed, so it has much more information than our SLM-based parser.
- KNP in this experiment was not equipped with commercial dictionaries.

As we mentioned above, our current model does not attempt to use lexical information about content words because of the data-sparseness problem. If we select the content words to be lexicalized by referring to the accuracy of the withheld corpus, the accuracy increases slightly to 92.9%. This means, however, our method is not able to efficiently use lexical information about the content words at this stage. Some model refinement should be explored for further improvements.

## 6 Related Work

Historically, the structures of natural languages have been described by a CFG and most parsers (Fujisaki et al., 1989; Pereira and Schabes, 1992; Charniak, 1997) are based on it. An SLM for English (Chelba and Jelinek, 2000), proposed as a language model for speech recognition, is also based on a CFG. On the other hand, an SLM for Japanese (Mori et al., 2000) is based on a Markov model by introducing a limit on language structures caused by our human memory limitations (Yngve, 1960; Miller, 1956). We introduced the same limitation into our language model and our parser is also based on a Markov model.

In the last decade, the importance of the lexicon has come into focus in the area of stochastic parsers. Nowadays, many state-of-the-art parsers are based on lexicalized models (Charniak, 1997; Collins, 1997). In these papers, they reported significant improvement in parsing accuracy by lexicalization. Our model is also lexicalized, the lexicalization is limited to grammatical function words because of the sparseness of data at the step of next word prediction. The greatest difference between our parser and many state-of-the-art parsers is that our parser is based on a generative language model, which works as a language model of a speech recognizer. Therefore, a speech recognizer equipped with our parser as its language model should be useful for a spoken language understanding system. The greatest advantage of our model over other structured language models is the ablity to refer to a variable part of the structured history by using ACTs.

There have been several attempts at Japanese parsers (Kurohashi and Nagao, 1994; Haruno et al., 1998; Fujio and Matsumoto, 1998; Kudo and Matsumoto, 2000). These Japanese parsers have all been based on a unique phrasal unit called a *bunsetsu*, a concatenation of one or more content words followed by some grammatical function words. Unlike these parsers, our model describes dependencies between words. Thus our parser can more easily be extended to other languages. In addition, since almost all pasers in other languages than Japanese output relationships between words, the output of our parser can be used by post-parser language processing systems proposed for many other languages (such as a word-level structural alignment of sentences in different languages).

## 7 Conclusion

In this paper we have described a structured language model (SLM) based on a dependency grammar. An SLM treats a sentence as a word sequence and predicts each word from beginning to end. The history at each step of prediction is a sequence of partial parse trees covering the preceding words. The problem is how to classify the tree-shaped histories to predict each word and structure while avoiding data-sparseness problems. As an answer, we propose to apply arboreal context trees (ACTs) to an SLM. An ACT is an extension of a context tree to a tree-shaped history. We built a parser based on an SLM with ACTs, whose parameters were estimated from 9,108 syntactically annotated sentences from a financial newspaper. We then tested the parser on 1,011 sentences from the same newspaper. The accuracy of the dependency relationships of the parser was 92.8%, higher than the accuracy of a parser based on an SLM without ACTs (89.8%). This proved experimentally that ACTs improve a parser based on an SLM.

## References

Eugene Charniak. 1997. Statistical parsing with a context-free grammar and word statistics. In *Proceedings of the 14th National Conference on Artificial Intelligence*, pages 598–603.

Eugene Charniak. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pages 124–131.

Ciprian Chelba and Frederic Jelinek. 1998. Exploiting syntactic structure for language modeling. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 225–231.

Ciprian Chelba and Frederic Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, 14:283–332.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, pages 16–23.

Masakazu Fujio and Yuji Matsumoto. 1998. Japanese dependency structure analysis based on lexicalized statistics. In *Proceedings of the Third Conference on Empirical Methods in Natural Language Processing*, pages 87–96.

T. Fujisaki, F. Jelinek, J. Cocke, E. Black, and T. Nishino. 1989. A probabilistic parsing method for sentence disambiguation. In *Proceedings of the International Parsing Workshop*.

Masahiko Haruno, Satoshi Shirai, and Yoshifumi Ooyama. 1998. Using decision trees to construct a practical parser. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 505–511.

Taku Kudo and Yuji Matsumoto. 2000. Japanese dependency structure analysis based on support vector machines. In *Proceedings of the 2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Sadao Kurohashi and Makoto Nagao. 1994. A syntactic analysis method of long Japanese sentences based on the detection of conjunctive structures. *Computational Linguistics*, 20(4):507–534.

Sadao Kurohashi, Toshihisa Nakamura, Yuji Matsumoto, and Makoto Nagao. 1994. Improvements of Japanese morphological analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, pages 22–28.

George A. Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, 63:81–97.

Shinsuke Mori, Masafumi Nishimura, Nobuyasu Itoh, Shiho Ogino, and Hideo Watanabe. 2000. A stochastic parser based on a structural word prediction model. In *Proceedings of the 18th International Conference on Computational Linguistics*, pages 558–564.

Shinsuke Mori, Masafumi Nishimura, and Nobuyasu Itoh. 2001. Improvement of a structured language model: Arbori-context tree. In *Proceedings of the Seventh European Conference on Speech Communication and Technology.*

Fernando Pereira and Yves Schabes. 1992. Inside-outside reestimation from partially bracketed corpora. In *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, pages 128–135.

Dana Ron, Yoram Singer, and Naftali Tishby. 1996. The power of amnesia: Learning probabilistic automata with variable memory length. *Machine Learning*, 25:117–149.

Victor H. Yngve. 1960. A model and a hypothesis for language structure. *The American Philosophical Society*, 104(5):444–466.