A Stochastic Parser Based on a Structural Word Prediction Model

Shinsuke MORI, Masafumi NISHIMURA, Nobuyasu ITOH, Shiho OGINO, Hideo WATANABE

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

Abstract

In this paper, we present a stochastic language model using dependency. This model considers a sentence as a word sequence and predicts each word from left to right. The history at each step of prediction is a sequence of partial parse trees covering the preceding words. First our model predicts the partial parse trees which have a dependency relation with the next word among them and then predicts the next word from only the trees which have a dependency relation with the next word. Our model is a generative stochastic model, thus this can be used not only as a parser but also as a language model of a speech recognizer. In our experiment, we prepared about 1,000 syntactically annotated Japanese sentences extracted from a financial newspaper and estimated the parameters of our model. We built a parser based on our model and tested it on approximately 100 sentences of the same newspaper. The accuracy of the dependency relation was 89.9%, the highest accuracy level obtained by Japanese stochastic parsers.

1 Introduction

The stochastic language modeling, imported from the speech recognition area, is one of the successful methodologies of natural language processing. In fact, all language models for speech recognition are, as far as we know, based on an n-gram model and most practical part-of-speech (POS) taggers are also based on a word or POS n-gram model or its extension (Church, 1988; Cutting et al., 1992; Merialdo, 1994; Dermatas and Kokkinakis, 1995). POS tagging is the first step of natural language processing, and stochastic taggers have solved this problem with satisfying accuracy for many applications. The next step is parsing, or that is to say discovering the structure of a given sentence. Recently, many parsers based on the stochastic approach have been proposed. Although their reported accuracies are high, they are not accurate enough for many applications at this stage, and more attempts have to be made to improve them further.

One of the major applications of a parser is to parse the spoken text recognized by a speech recognizer. This attempt is clearly aiming at spoken language understanding. If we consider how to combine a parser and a speech recognizer, it is better if the parser is based on a generative stochastic model, as required for the language model of a speech recognizer. Here, "generative" means that the sum of probabilities over all possible sentences is equal to or less than 1. If the language model is generative, it allows a seamless combination of the parser and the speech recognizer. This means that the speech recognizer has the stochastic parser as its language model and benefits richer information than a normal n-gram model. Even though such a combination is not possible in practices , the recognizer outputs N-best sentences with their probabilities, and the parser, taking them as input, parses all of them and outputs the sentence with its parse tree that has the highest probability of all possible combinations. As a result, a parser based on a generative stochastic language model may help a speech recognizer to select the most syntactically reasonable sentence among candidates. Therefore, it is better if the language model of a parser is generative.

In this paper, taking Japanese as the object language, we propose a generative stochastic language model and a parser based on it. This model treats a sentence as a word sequence and predicts each word from left to right. The history at each step of prediction is a sequence of partial parse trees covering the preceding words. To predict a word, our model first predicts which of the partial parse trees at this stage have dependency relation with the word, and then predicts the word from the selected partial parse trees. In Japanese each word depends on a subsequent word, that is to say, each dependency relation is left to right, it is not necessary to predict the direction of each dependency relation. So in order to extend our model to other languages, the model may have to predict the direction of each dependency. We built a parser based on this model, whose parameters are estimated from 1,072 sentences in a financial newspaper, and tested it on 119 sentences from the same newspaper. The accuracy of the depen-

dency relation was 89.9%, the highest obtained by any Japanese stochastic parsers.

2 Stochastic Language Model based on Dependency

In this section, we propose a stochastic language model based on dependency. Unlike most stochastic language models for a parser, our model is theoretically based on a hidden Markov model. In our model a sentence is predicted word by word from left to right and the state at each step of prediction is basically a sequence of words whose modificand has not appeared yet. According to a psycholinguistic report on language structure (Yngve, 1960), there is an upper limit on the number of the words whose modificands have not appeared yet. This limit is determined by the number of slots in short-term memory, 7 ± 2 (Miller, 1956). With this limitation, we can design a parser based on a finite state model.

2.1 Sentence Model

The basic idea of our model is that each word would be better predicted from the words that have a dependency relation with the word to be predicted than from the preceding two words (tri-gram model). Let us consider the complete structure of the sentence in Figure 1 and a hypothetical structure after the prediction of the fifth word at the top of Figure 2. In this hypothetical structure, there are three trees: one root-only tree (t_b composed of w_3) and two two-node trees (t_a containing w_1 and w_2, and t_c containing w_4 and w_5). If the last two trees (t_b and t_c) depend on the word w_6, this word may better be predicted from these two trees. From this point of view, our model first predicts the trees depending on the next word and then predicts the next word from these trees.

Now, let us make the following definitions in order to explain our model formally.

- $\boldsymbol{w} = w_1 w_2 \cdots w_n$: a sequence of words. Here a word is defined as a pair consisting of a string of alphabetic characters and a part of speech (e.g. the/DT).
- $\boldsymbol{t}_i = t_1 t_2 \cdots t_{k_i}$: a sequence of partial parse trees covering the i-prefix words ($w_1 w_2 \cdots w_i$).
- $\boldsymbol{t}_i^+$ and $\boldsymbol{t}_i^-$: subsequences of $\boldsymbol{t}_i$ having and not having a dependency relation with the next word respectively. In Japanese, like many other languages, no two dependency relations cross each other; thus $\boldsymbol{t}_i = \boldsymbol{t}_i^- \boldsymbol{t}_i^+$.
- $\langle \boldsymbol{t}\ w \rangle$: a tree with w as its root and $\boldsymbol{t}$ as the sequence of all subtrees connected to the root. After w_{i+1} has been predicted from the trees depending on it ($\boldsymbol{t}_i^+$), there are trees remaining ($\boldsymbol{t}_i^-$) and a newly produced tree ($\langle \boldsymbol{t}_i^+ w_{i+1} \rangle$); thus $\boldsymbol{t}_{i+1} = \boldsymbol{t}_i^- \cdot \langle \boldsymbol{t}_i^+ w_{i+1} \rangle$.

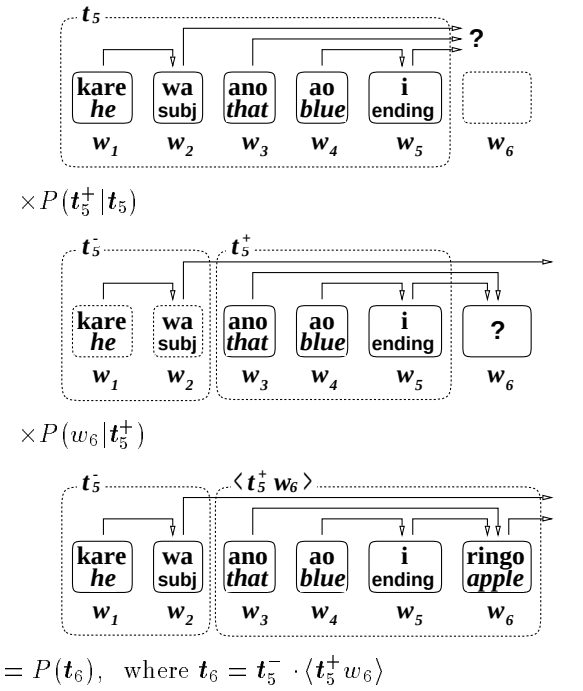

Figure 2: Word prediction from a partial parse

- y_{max} : upper limit on the number number of words whose modificands have not appeared yet.

Under these definitions, our stochastic language model is defined as follows:

$$P(\boldsymbol{w}) = \prod_{i=1}^{n} P(w_i | w_1 w_2 \cdots w_{i-1}) \approx \sum_{\boldsymbol{t}_n \in T_n} \prod_{i=1}^{n} P(w_i | \boldsymbol{t}_{i-1}^+) P(\boldsymbol{t}_{i-1}^+ | \boldsymbol{t}_{i-1}) \quad (1)$$

where T_n is all possible binary trees with n nodes. Here, the first factor, $(P(w_i|\boldsymbol{t}_{i-1}^+))$, is called the word prediction model and the second, $(P(\boldsymbol{t}_{i-1}^+|\boldsymbol{t}_{i-1}))$, the state prediction model. Let us consider Figure 2 again. At the top is the state just after the prediction of the fifth word. The state prediction model then predicts the partial parse trees depending on the next word among all partial parse trees, as shown in the second figure. Finally, the word prediction model predicts the next word from the partial parse trees depending on it.

As described above, there may be an upper limit on the number of words whose modificands have not yet appeared. To put it in another way, the length of the sequence of partial parse trees ($\boldsymbol{t}_i$) is limited.

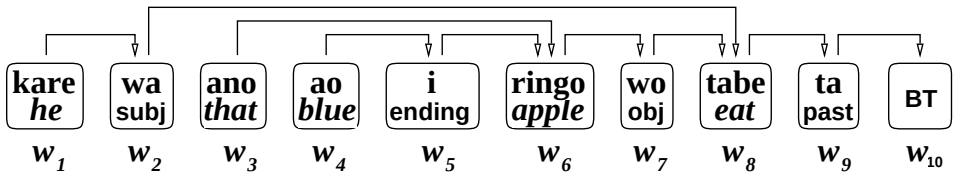

Figure 1: Dependency structure of a sentence.

Therefore, if the depth of the partial parse tree is also limited, the number of possible states is limited. Under this constraint, our model can be considered as a hidden Markov model. In a hidden Markov model, the first factor is called the output probability and the second, the transition probability.

Since we assume that no two dependency relations cross each other, the state prediction model only has to predict the number of the trees depending on the next word. Thus $P(\boldsymbol{t}_{i-1}^{+}|\boldsymbol{t}_{i-1}) = P(y|\boldsymbol{t}_{i-1})$ where y is the number of trees in the sequence $\boldsymbol{t}_{i-1}^{+}$. According to the above assumption, the last y partial parse trees depend on the i-th word. Since the number of possible parse trees for a word sequence grows exponentially with the number of the words, the space of the sequence of partial parse trees is huge even if the length of the sequence is limited. This inevitably causes a data-sparseness problem. To avoid this problem, we limited the number of levels of nodes used to distinguish trees. In our experiment, only the root and its children are checked to identify a partial parse tree. Hereafter, we represent P_{LL} to denote this model, in which the lexicon of the first level and that of the second level are considered. Thus, in our experiment each word and the number of partial parse trees depending on it are predicted by a sequence of partial parse trees that take account of the nodes whose depth is two or less. It is worth noting that if the dependency structure of a sentence is linear — that is to say, if each word depends on the next word, — then our model will be equivalent to a word tri-gram model.

We introduce an interpolation technique (Jelinek et al., 1991) into our model like those used in n-gram models. By loosening tree identification regulations, we obtain a more general model. For example, if we check only the POS of the root and the POS of its children, we will obtain a model similar to a POS tri-gram model (denoted P_{PP} hereafter). If we check the lexicon of the root, but not that of its children, the model will be like a word bi-gram model (denoted P_{NL} hereafter). As a smoothing method, we can interpolate the model P_{LL}, similar to a word tri-gram model, with a more general model, P_{PP} or P_{NL}. In our experiment, as the following formula indicates, we interpolated seven models of different generalization levels:

$$\begin{aligned} P(w_i|\boldsymbol{t}_{i-1}^{+}) &= \lambda_6 P_{LL}(w_i|\boldsymbol{t}_{i-1}^{+}) + \lambda_5 P_{PL}(w_i|\boldsymbol{t}_{i-1}^{+}) \\ &\quad +\lambda_4 P_{PP}(w_i|\boldsymbol{t}_{i-1}^{+}) + \lambda_3 P_{NL}(w_i|\boldsymbol{t}_{i-1}^{+}) \\ &\quad +\lambda_2 P_{NP}(w_i|\boldsymbol{t}_{i-1}^{+}) + \lambda_1 P_{NN}(w_i|\boldsymbol{t}_{i-1}^{+}) \\ &\quad +\lambda_0 P_{w,0-\mathrm{gram}} \end{aligned} \quad (2)$$

where X in P_{YX} is the check level of the first level of the tree (N: none, P: POS, L: lexicon) and Y is that of the second level, and $P_{w,0-\mathrm{gram}}$ is the uniform distribution over the vocabulary $\mathcal{W}$ ($P_{w,0-\mathrm{gram}}(w) = 1/|\mathcal{W}|$).

The state prediction model ($P(y|\boldsymbol{t}_{i-1})$) is also interpolated in the same way. In this case, the possible events are $y = 1, 2, \ldots, y_{max}$, thus; $P_{y,0-\mathrm{gram}} = 1/y_{max}$.

2.2 Parameter Estimation

Since our model is a hidden Markov model, the parameters of a model can be estimated from a row corpus by EM algorithm (Baum, 1972). With this algorithm, the probability of the row corpus is expected to be maximized regardless of the structure of each sentence. So the obtained model is not always appropriate for a parser.

In order to develop a model appropriate for a parser, it is better that the parameters are estimated from a syntactically annotated corpus by a maximum likelihood estimation (MLE) (Merialdo, 1994) as follows:

$$\begin{aligned} P(w|\boldsymbol{t}^{+}) &\overset{\mathrm{MLE}}{=} \frac{f(\langle \boldsymbol{t}^{+}\ w_i \rangle)}{\sum_w f(\langle \boldsymbol{t}^{+}\ w_i \rangle)} \\ P(\boldsymbol{t}^{+}|\boldsymbol{t}) &\overset{\mathrm{MLE}}{=} \frac{f(\boldsymbol{t}^{+}, \boldsymbol{t})}{f(\boldsymbol{t})} \end{aligned}$$

where $f(x)$ represents the frequency of an event x in the training corpus.

The interpolation coefficients in the formula (2) are estimated by the deleted interpolation method (Jelinek et al., 1991).

2.3 Selecting Words to be Lexicalized

Generally speaking, a word-based n-gram model is better than a POS-based n-gram model in terms of

predictive power; however lexicalization of some infrequent words may be harmful because it may cause a data-sparseness problem. In a practical tagger (Kupiec, 1989), only the most frequent 100 words are lexicalized. Also, in a state-of-the-art English parser (Collins, 1997) only the words that occur more than 4 times in training data are lexicalized.

For this reason, our parser selects the words to be lexicalized at the time of learning. In the lexicalized models described above (P_{LL}, P_{PL} and P_{NL}), only the selected words are lexicalized. The selection criterion is parsing accuracy (see section 4) of a held-out corpus, a small part of the learning corpus excluded from parameter estimation. Thus only the words that are predicted to improve the parsing accuracy of the test corpus, or unknown input, are lexicalized. The algorithm is as follows (see Figure 3):

1. In the initial state all words are in the class of their POS.
2. All words are sorted in descending order of their frequency, and the following process is executed for each word in this order:
   (a) The word is lexicalized provisionally and the accuracy of the held-out corpus is calculated.
   (b) If an improvement is observed, the word is lexicalized definitively.

The result of this lexicalization algorithm is used to identify a partial parse tree. That is to say, only lexicalized words are distinguished in lexicalized models. If no words were selected to be lexicalized, then $P_{NL} = P_{NP}$ and $P_{LL} = P_{PL} = P_{PP}$. It is worth noting that if we try to join a word with another word, then this algorithm will be a normal top-down clustering algorithm.

2.4 Unknown Word Model

To enable our stochastic language model to handle unknown words, we added an unknown word model based on a character bi-gram model. If the next word is not in the vocabulary, the model predicts its POS and the unknown word model predicts the string of the word as follows:

$$P(w|POS) = \prod_{i=1}^{m+1} P_{POS}(x_i|x_{i-1})$$
$$\text{where } w = x_1 x_2 \cdots x_m,\ x_0 = x_{m+1} = \mathrm{BT}.$$

BT, a special character corresponding to a word boundary, is introduced so that the sum of the probability over all strings is equal to 1.

In the parameter estimation described above, a learning corpus is divided into k parts. In our experiment, the vocabulary is composed of the words occurring in more than one partial corpus and the other words are used for parameter estimation of the unknown word model. The bi-gram probability of the unknown word model of a POS is estimated from the words among them and belonging to the POS as follows:

$$P_{\mathrm{POS}}(x_i|x_{i-1}) \overset{\mathrm{MLE}}{=} \frac{f_{\mathrm{POS}}(x_i, x_{i-1})}{f_{\mathrm{POS}}(x_{i-1})}.$$

The character bi-gram model is also interpolated with a uni-gram model and a zero-gram model. The interpolation coefficients are estimated by the deleted interpolation method (Jelinek et al., 1991).

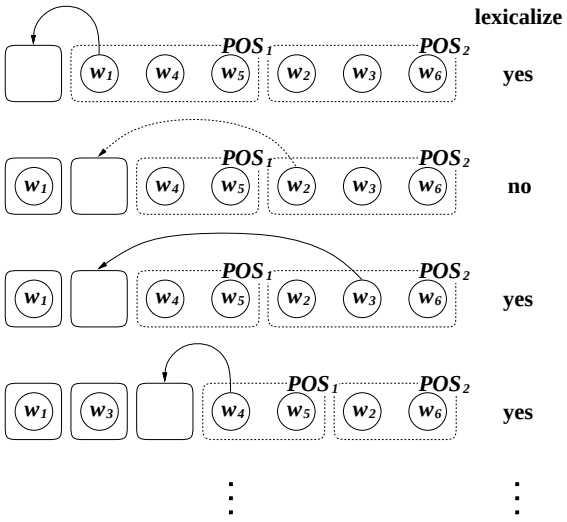

Figure 3: A conceptual figure of the lexicalization algorithm.

3 Syntactic Analysis

Generally, a parser may be considered as a module that receives a sequence of words annotated with a POS and outputs its structure. Our parser, which includes a stochastic unknown word model, however, is able to accept a character sequence as an input and execute segmentation, POS tagging, and syntactic analysis simultaneously[1] . In this section, we explain our parser, which is based on the language model described in the preceding section.

3.1 Stochastic Syntactic Analyzer

A syntactic analyzer, based on a stochastic language model, calculates the parse tree (see Figure 1) with the highest probability for a given sequence of characters $\boldsymbol{x}$ according to the following formula:

$$\hat{T} = \underset{\boldsymbol{w}(T)=\boldsymbol{x}}{\mathbf{argmax}}\, P(T|\boldsymbol{x})$$

[1] There is no space between words in Japanese

$$= \operatorname*{argmax}_{\boldsymbol{w}(T)=\boldsymbol{x}} P(T|\boldsymbol{x})P(\boldsymbol{x})$$
$$= \operatorname*{argmax}_{\boldsymbol{w}(T)=\boldsymbol{x}} P(\boldsymbol{x}|T)P(T) \quad (\because \text{Bayes' formula})$$
$$= \operatorname*{argmax}_{\boldsymbol{w}(T)=\boldsymbol{x}} P(T) \quad (\because P(\boldsymbol{x}|T) = 1),$$

where $\boldsymbol{w}(T)$ represents the concatenation of the word string in the syntactic tree T. $P(T)$ in the last line is a stochastic language model. In our parser, it is the probability of a parse tree T defined by the stochastic dependency model including the unknown word model described in section 2.

$$P(T) = \prod_{i=1}^{n} P(w_i|\boldsymbol{t}_{i-1}^{+})P(\boldsymbol{t}_{i-1}^{+}|\boldsymbol{t}_{i-1}), \qquad (3)$$

where $w_1 w_2 \cdots w_n = \boldsymbol{w}(T)$.

3.2 Solution Search Algorithm

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

The parser repeats a state transition, reading characters of the input sentence from left to right. In order that the structure of the input sentence may be a tree, the number of trees of the final state $\boldsymbol{t}_n$ must be 1 and no more. Among the states that satisfy this condition, the parser selects the state with the highest probability. Since our language model uses only the root and its children of a partial parse tree to distinguish states, the last state does not have enough information to construct the parse tree. The parser can, however, calculate the parse tree from the sequence of states, or both the word sequence and the sequence of y, the number of trees that depend on the next word. Thus it memorizes these values at each step of prediction. After the most probable last state has been selected, the parser constructs the parse tree by reading these sequences from top to bottom.

4 Evaluation

We developed a POS-based model and its lexicalized version explained in section 2 to evaluate their predictive power, and implemented parsers based on them that calculate the most probable dependency tree from a given character sequence, using the solution search algorithm explained in section 3 to observe their accuracy. In this section, we present and discuss the experimental results.

4.1 Conditions on the Experiments

The corpus used in our experiments consists of articles extracted from a financial newspaper (*Nihon Keizai Shinbun*). Each sentence in the articles is segmented into words and its dependency structure is annotated by linguists using an editor specially designed for this task at our site. The corpus was divided into ten parts; the parameters of the model were estimated from nine of them and the model was tested on the rest (see Table 1). A small part of each leaning corpus is withheld from parameter estimation and used to select the words to be lexicalized. After checking the learning corpus, the maximum number of partial parse trees is set to 10 ($y_{max} = 10$).

Table 1: Corpus.

	#sentences	#words	#chars
learning	1,072	30,292	46,212
test	119	3,268	4,909

To evaluate the predictive power of our model, we calculated their cross entropy on the test corpus. In this process, the annotated tree in the test corpus is used as the structure of the sentences. Therefore the probability of each sentence in the test corpus is not the summation over all its possible derivations. To compare the POS-based model and the lexicalized model, we constructed these models using the same learning corpus and calculated their cross entropy on the same test corpus. The POS-based model and the lexicalized model have the same unknown word model, thus its contribution to the cross entropy is constant.

We implemented a parser based on the dependency models. Since our models, including a character-based unknown word model, can return the best parse tree with its probability for any input, we can build a parser that receives a character sequence as input. It is not easy to evaluate, however, because errors may occur in segmentation of the sequence into words and in estimation of their POSs. For this reason, in the following description, we assume a word sequence as the input.

The criterion for a parser is the accuracy of its output dependency relations. This criterion is widely used to evaluate Japanese dependency parsers. The accuracy is the ratio of the number of the words annotated with the same dependency to the number of the words as in the corpus:

$$\textit{accuracy} = \frac{\#\text{words depending on the correct word}}{\#\text{words}}$$

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 depends always on the last word.

Table 2: Cross entorpy and accuracy of each model.

language model	cross entropy	accuracy
selectively lexicalized	6.927	89.9%
completely lexicalized	6.651	87.1%
POS-based	7.000	87.5%
linear structure*	—	78.7%

* Each word depends on the next word.

4.2 Evaluation

Table 2 shows the cross entropy and parsing accuracy of the baseline, where all words depend on the next word, the POS-based dependency model and two lexicalized dependency models. In the selectively lexicalized model, words to be lexicalized are selected by the algorithm described in section 2. In the completely lexicalized model, all words are lexicalized. This result attests experimentally that the parser based on the selectively lexicalized model is the best parser. As for predictive power, however, the completely lexicalized model has the lowest cross entropy. Thus this model is estimated to be the best language model for speech recognition. Although there is no direct relation between cross entropy of the language model and error rate of a speech recognizer, if we consider a spoken language parser, it may be better to select the words to be lexicalized using other criterion.

We calculated the cross entropy and the parsing accuracy of the model whose parameters are estimated from 1/4, 1/16, and 1/64 of the learning corpus. The relation between the learning corpus size and the cross entropy or the parsing accuracy is shown in Figure 4. The cross entropy has a stronger tendency to decrease as the corpus size increases. As for accuracy, there is also a tendency for parsers to become more accurate as the size of the learning increases. The size of the corpus we have at this stage is not at all large. However, its accuracy is at the top level of Japanese parsers, which use 50,000 - 190,000 sentences. Therefore, we conclude that our approach is quite promising.

5 Related Works

Historically, structures of natural languages have been described by a context-free grammar and ambiguities have been resolved by parsers based on a context-free grammar (Fujisaki et al., 1989). In recent years, some attempts have been made in the area of parsing by a finite state model (Oflazer, 1999) etc. Our parser is also based on a finite state model. Unlike these models, we focused on reports on a limit on language structure caused by the capacity our memory (Yngve, 1960) (Miller, 1956). Thus our

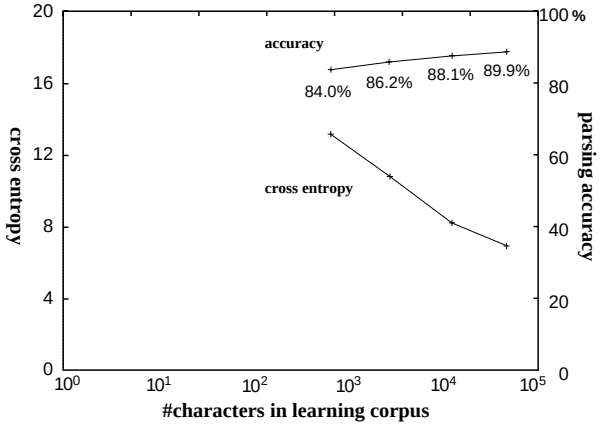

Figure 4: Relation between cross entropy and parsing accuracy.

model is psycholinguistically more appropriate.

Recently, in the area of parsers based on a stochastic context-free grammar (SCFG), some researchers have pointed out the importance of the lexicon and proposed lexicalized models (Charniak, 1997; Collins, 1997). In these papers, they reported significant improvement of parsing accuracy. Taking these reports into account, we introduced a method of partial lexicalization and reported significant improvement of parsing accuracy. Our lexicalization method is also applicable to a SCFG-based parser and improves its parsing accuracy.

The model we present in this paper is a generative stochastic language model. Chelba and Jelinek (1998) presented a similar model. In their model, each word is predicted from two right-most head words regardless of dependency relation between these head words and the word. Eisner (1996) also presented a stochastic structural language model, in which each word is predicted from its head word and the nearest one. This model is very similar to the parser presented by Collins (1996). The greatest difference between our model and these models is in that our model predicts the next word from the head words, or partial parse trees, depending on it. Clearly, it is not always two right-most head words that have dependency relation with the next word. It follows that our model is linguistically more appropirate.

There have been some attempts at stochastic Japanese parser (Haruno et al., 1998) (Fujio and Matsumoto, 1998) (Mori and Nagao, 1998). These Japanese parsers are based on a unit called *bunsetsu*, a sequence of one or more content words followed by zero or more function words. The parsers take a sequence of units and outputs dependency relations between them. Unlike these parsers, our model de-

scribes dependencies between words; thus our model can easily be extended to other languages. As for the accuracy, although a direct comparison is not easy between our parser (89.9%; 1,072 sentences) and these parsers (82% - 85%; 50,000 - 190,000 sentences) because of the difference of the units and the corpus, our parser is one of the state-of-the-art parsers for Japanese language. It should be noted that our model describes relations among three or more units (case frame, consecutive dependency relations, etc.); thus our model benefits a greater deal from increase of corpus size.

6 Conclusion

In this paper we have presented a stochastic language model based on dependency structure. This model treats a sentence as a word sequence and predicts each word from left to right. The history at each step of prediction is a sequence of partial parse trees covering the preceding words. To predict a word, our model first selects the partial parse trees that have a dependency relation with the word, and then predicts the next word from the selected partial parse trees. We also presented an algorithm for lexicalization. We built parsers based on the POS-based model and its lexicalized version, whose parameters are estimated from 1,072 sentences of a financial newspaper. We tested the parsers on 119 sentences from the same newspaper, which were excluded from the learning. The accuracy of the dependency relation of the lexicalized parser was 89.9%, the highest obtained by any Japanese stochastic parser.

References

L. E. Baum. 1972. An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov process. *Inequalities*, 3:1–8.

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.

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.

Kenneth Ward Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191.

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.

Doug Cutting, Julian Kupiec, Jan Pedersen, and Penelope Sibun. 1992. A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, pages 133–140.

Evangelos Dermatas and George Kokkinakis. 1995. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–163.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th International Conference on Computational Linguistics*, pages 340–345.

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.

Fredelick Jelinek, Robert L. Mercer, and Salim Roukos. 1991. Principles of lexical language modeling for speech recognition. In *Advances in Speech Signal Processing*, chapter 21, pages 651–699. Dekker.

Julian Kupiec. 1989. Augmenting a hidden Markov model for phrase-dependent word tagging. In *Proceedings of the DARPA Speech and Natural Language Workshop*, pages 92–98.

Bernard Merialdo. 1994. Tagging English text with a probabilistic model. *Computational Linguistics*, 20(2):155–171.

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 and Makoto Nagao. 1998. A stochastic language model using dependency and its improvement by word clustering. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 898–904.

Kemal Oflazer. 1999. Dependency parsing with an extended finite state approach. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 254–260.

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