Internal Use Only

002

TR-IT-0127

Stochastical Parsing of Ill-Formedness in Spoken Natural Language

Michael Paul Yasuharu Den

1995.9

Abstract

This report describes a statistical approach to parse spontaneous speech, taking into account the phenomena of natural conversations. The input sentences are taken from the ATR Dialogue Database. Using an adapted version of the stochastical language model BLI , we yields in the detection and recovery of ill-formedness occuring in spontaneous speech.

ATR Interpreting Telecommunications Research Laboratories

Contents

1 Introduction

In contrast to conventional grammatical analysis the parsing of spontaneous speech has to take into account the characteristics of natural conversations, which does not occur in more formal styles of speech. These ungrammatical structures cause problems to an automated syntactic analysis system.

We describe an approach to detect and recover from different kinds of ill-formedness, using input sentences of the ATR Dialogue Database. The algorithm is based on the stochastical language model BLI (Bayesian Language Inference). This model allows us to analyze ill-formed input with mathematically sound consideration of full syntactic context.

First we give a short overview over the different phenomena of spontaneous speech. Then we will describe the BLI model and the way how this approach can be adapted to our system. Preliminary experiments to test our algorithm on artificially changed data will be described and the results are used to improve our approach in order to be applied successfully on "real data". The report ends with the summary of the results and future aspects.

2 Ill-Formedness in Spontaneous Speech

Analysis of conversational data[1] collected by Laurel Fais [8] has shown that ill-formed input proves to be very frequent. As reported in [15] about 75% of utterances in natural conversations are well-formed by any criterion. The remaining 25% of utterances are ungrammatical and ill-formed, thus cannot be handled by conventional grammatical analysis.

In order to fulfill the goal of parsing ill-formed sentences we have to take into account the nature of the phenomena of spontaneous speech. There are several characteristics of spontaneous speech, which are not found in more formal styles of speech. These phenomena can be grouped in different categories [7],[8]:

syntactic violations

this category covers structural differences between speech and writing [19], e.g. the omission of particles, which are used to identify the grammatical nature of the preceding sequence. in spoken Japanese.

starts and stops of conversation

these phenomena introduce a certain number of structures, which make no significant contribution to the conversation:

- *false start*: the initial uttered material is "replaced" by the following utterance. In the case of a *repetition* the replacement is identical to the original; a *repair* corrects a lexical item, whereas a *fresh start* corrects a phrase.
- *filled pause*: non-word sounds, that a speaker typically makes to fill silence, when taking time to consider a structure, lexical item or conversational direction, e.g. "um" or "ah" in English and "えと" or "あの" in Japanese [24]

[1]The ATR environment for Multimodal Interaction [14]

- *break*: characterized by the lack of continuity between initial, discarded utterance and the following restart. Breaks are instances of omissions, in which material is deleted, that cannot be recovered.
- *knowable omission*: speaker deletes material from an utterances, that can be recovered in some way. Two categories can be distinguished: in the first syntactic material necessary to the well-formedness is deleted (→ break); in the second adjunctival material is omitted.
- *interjection*: break of utterance in order to change direction, but then return to the breaking point and resumes the thought. Thereby the return to the first construction is often accompanied by a repetition of the initial phrase.
- *correction*: switching from one syntactic direction to another ("self-repair"). In contrast to *breaks* the semantic direction remains across the syntactic shift, but corrections doesn't return to the original structure, as *interjections*.
- *repetition*: repeating material around the reconstructing phrase. Many repetitions only repeat a particular word or phrase. But they serve other functions as well, e.g. emphasizing an utterance or confirming understanding of a statement.

noun phrase phenomena

structures, such as *topicalization*, *left and right dislocation* and the use of *appositives* are all grammatical, but they are fairly common phenomena in natural conversations.

sentence level issues

fragmentary exclamatory phrases, which can be divided in different groups of expressions. *Idiomatic phrases and structures* (e.g. "hey", "oh look") are singular, i.e. they occur alone without a structural attachment to the sentence. *Yes/No-answers* (e.g. "OK", "はい") are used to signify dis-/agreement with or understanding of a previous utterance. *Discourse markers* (e.g. "well", "ね") are sequentially dependent elements, which brackets units of the talk [23].

Between human beings these phenomena do not weaken the understanding of natural conversations, because humans adapted some mechanism to overcome these problems. However, the special characteristics of spontaneous speech do cause significant problems to automated syntactic analysis systems.

Moreover, other errors introduced by speech recognizer failure or incorrect part-of-speech tagging are likely to come up and widen the gap between the utterance as intended by the speaker and the actual input given to the parser.

In the following, we will assume that this input is a string of symbols delivered by a part-of-speech tagger, allowing some symbols to be considered "unrecognized", the other symbols being taken from a list of grammatical categories (cf. Appendix A and B). Each string of symbols will be referred to as a *sentence*. The input will be considered *ill-formed* whenever the intended *sentence* (as defined by human analysis) differs from the actual input *sentence* yielded by the part-of-speech tagger. These differences between intended sentence and actual input sentence can be classified into three elementary categories:

- insertion of a symbol
- substitution of a symbol for another symbol
- deletion of a symbol

Insertion phenomena of *unrecognized* symbols are typically encountered in the case of "filled pauses", whereas insertion phenomena of *recognized* symbols occur in "false starts". Filled pauses and false starts, usually discussed in the literature as *disfluencies*, may of course cause the insertion of one or more symbols.

A *substitution of a symbol* for another symbol typically occurs in cases, where speech recognition either fails to identify a constituent or just mistakes a constituent for another one. Errors coming from the speaker can also be responsible for this type of ill-formedness, e.g. in the case of slips of the tongue, but these errors don't occur frequently [4].

Finally a *deletion of a symbol* will be encountered whenever a speaker deletes material from an utterance. This type of deletion is typical of the differences between spoken and written language.

The framework defined so far is purely syntactic. The input sentences given to our parser are a list of grammatical symbols, and only syntactic information is made available. Other works taking into account semantics and world knowledge have been conducted, e.g. using abduction-based inference schemes [5],[10].

But these approaches seems not be sufficient enough. So we have considered alternative solutions in order to parse ill-formed input [11]. Because the use of stochastic models in the field of natural language processing has recently led to dramatic improvements in the performance of parsing systems, a statistical approach seems to be promising [1],[18]. While allowing automatic training of stochastic grammars, these models also provide the quantitative analysis needed in the disambiguation process. This mathematically sound analysis makes the use of statistical models quite relevant to the task of parsing ill-formed input.

However simple local stochastic models like *n-gram* models [12], *probabilistic context-free grammars* [13] or *tree-adjoining grammar formalisms* [22] only give us general information about how likely a structure is to appear *anywhere* in a given sentence. Rule expansion at a given node only depends on the portion of input spanned by this node (inside context), and doesn't consider the remaining part of the input (outside context). Therefore these simple statistical models would not display the full consideration of context.

The target of our approach is to develop an algorithm, which allows the full use of structural information (combining detection and parsing process), which takes into account the whole input sentence (inside and outside context) and which uses powerful mathematical tools in the disambiguation process in order to select the best parse among a great number of possible parses. The approach is based on the stochastical language model BLI , which we will describe in the next section.

3 Bayesian Language Inference (BLI)

The *Bayesian Language Inference*[2] is a language model for speech recognition, which combines the theory of *Bayesian Networks* [9] with the concept of *Probabilistic Context Free Grammars* [2].

A Context Free Grammar (CFG) is used to describe the natural language. The words of the language can be clustered in categories, e.g. a noun or an adverb, which are referred as *terminals.* The fragments of sentences, e.g. a noun-phrase or a verb-phrase, are represented by *non-terminals.* Through the repeated application of *rewriting rules* for non-terminals, sentences can be generated. If these rules are restricted, so that a non-terminal symbol can either be rewritten as a string of two non-terminals or as a single terminal symbol, the CFG is said to be in *Chomsky normal form* [3].

In order to use context information during the selection of a rewriting rule, a *Stochastic Context Free Grammar* (SCFG/PCFG) assigns a certain probability p to each rule, which provides a measure for the strings which can be generated. A PCFG, which consist of N_{nt} non-terminals and N_t terminals, is defined as:

$$< W, G, s, R >, \qquad \begin{array}{ll} (W_i)_{1\leq i\leq N_t} & = \text{set of terminals} \\ (G_j)_{1\leq j\leq N_{nt}} & = \text{set of non-terminals} \\ s & = \text{starting symbol} \\ R & = \text{set of rewriting rules} \end{array}$$

The BLI uses a PCFG in Chomsky normal form, which can be described by the following quantities:

$\mathbf{A_{ijk}}$ a tensor denoting the probabilities for the rewriting rules $G_i \rightarrow G_j G_k \in R$.

$\mathbf{B_{im}}$ a matrix denoting the probabilities for the rewriting rules $G_i \rightarrow W_m \in R$.

$\mathbf{p_n}$ a vector describing the probability of G_n being the initial symbol.

Additional, these quantities must satisfy the following stochastic constraints:

$$\forall i : \quad \sum_{j=1}^{N_{nt}} \sum_{k=1}^{N_{nt}} A_{ijk} + \sum_{m=1}^{N_t} B_{im} = 1, \qquad \sum_{n=1}^{N_{nt}} p_n = 1$$

The *theory of belief propagation* in Bayesian networks [20] is concerned with the propagation of partial information between (possibly hidden) nodes in a network. Given the observable evidence, the probability distribution over the states of a node can be computed by passing certain messages on a local scale, i.e. between adjacent nodes.

In BLI , these techniques are used, to learn the PCFG rules of a given grammar from examples by processing unlabeled training text. The existing estimates of the grammar rule probabilities are used to construct parse trees over segments of utterances. Belief propagation

[2]BLI was developed by Helmut Lucke at ATR [16],[17].

is then applied to these tree in order to obtain the posterior probability distribution of the non-terminal symbols at each grammar node.

Before describing the single steps of the algorithm, we have to introduce the notations, that are used in the rest of this report (cf. figure 1).

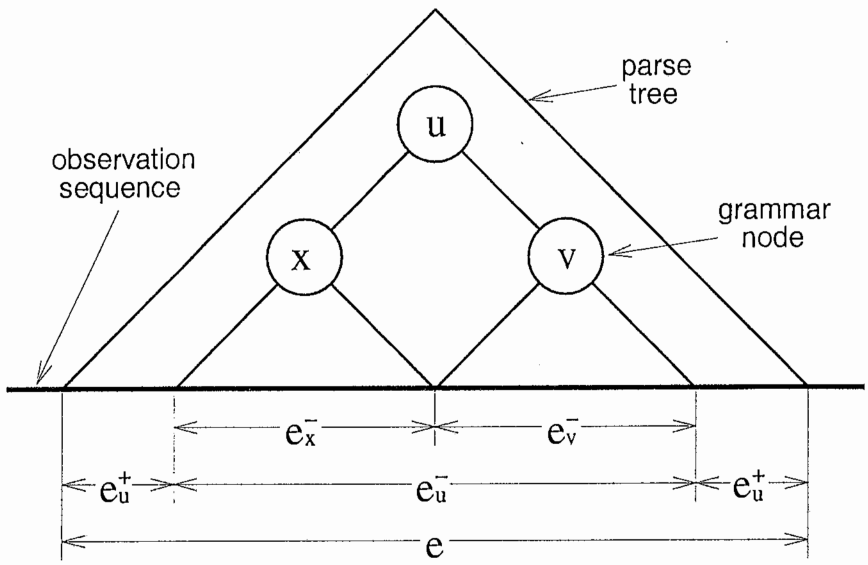

Figure 1: Definition of inner and outer evidence

With N_{nt} and N_t we denote the number of non-terminal and terminal symbols, respectively. We will use the letters u,v,x as variables describing the non-terminal symbols. The part of the observation sequence, which is produced by u, is called the *inner evidence* e_u^- and the remaining part of the sequence *outer evidence* e_u^+. e stands for the entire observation sequence, which is spanned by the parse tree. Further we denote the conditional probability $\mathrm{BEL}(u) = P(u|e)$ the *belief* of u. In order to calculate the belief-vector $\mathrm{BEL}(u)$ we define the auxiliary functions $\lambda(u) = P(e_u^-|u)$ and $\pi(u) = P(u|e_u^+)$. Using these notation we can define the *entropy* of a tree node u as:

$$E(u) = -\log_2(P(e_u^-, e_u^+)) = -\log_2(\lambda(u) \cdot \pi(u))$$

Thereby $a \cdot b$ denotes the familiar *dot product* $a \cdot b = \sum_{i=1}^{N_{nt}} a_i b_i$, whereas the *vector product* ab is the component-wise vector product $(ab)_i = a_i b_i$.

The BLI algorithm is divided in several steps. In the segmentation phase the observation sequence is divided into segments. For each segment the topology of the spanning tree has to be calculated, i.e. we have to decide on the structure of the tree without explicit knowledge of the identity of the non-terminal symbols at each node. The decision, which non-terminal should be assigned to the nodes of the structured tree, is carried out in the assignment phase. The last problem in BLI is concerned with the continuous training of the parameter, i.e. how can the A_{ijk} and B_{im} be re-estimated to reduce the overall entropy. But this phase isn't addressed here and the reader is referred to [16] and [17].

3.1 Segmentation Problem

The input of the BLI consist of unlabeled data, i.e. the algorithm uses no information about sentence boundaries as segmentation points. So even units, smaller than a sentence (e.g. a phrase), can be chosen as a segmentation unit. In order to find the segmentation points, the BLI algorithm uses a simple dynamic programming type approach.

Up to a specified maximal length $T_b - T_a$, the parts of the input data $(T_a\ T_b) = s_{T_a} \ldots s_{T_b}$ will be observed. Then for each pair $(t_a\ t_b)$ of the observation sequence, with $T_a \leq t_a < t_b \leq T_b$, a node $n(t_a\ t_b)$ will be conjectured, which spans the segment $s_{t_a} \ldots s_{t_b}$. For such a node x the probability of generating $e_x^- = s_{x_a} \ldots s_{x_b}$ is defined as:

$$\Lambda(x) = \Lambda(t_{x_a}, t_{x_b}) = P(e_x^- | x)$$

The following recursion allows us to calculate the Λ-values of the nodes of higher rank in a bottom-up fashion:

$$\Lambda(u)_i = \Lambda(t_x, t_v)_i = \sum_{t_{xv}} \sum_{jk} A_{ijk} \cdot \Lambda(t_x, t_{xv})_j \cdot \Lambda(t_{xv}, t_v)_k$$

For each segment $(t_a\ t_b)$ up to the maximum length these Λ-vectors and the entropy of the spanning root node of $(t_a\ t_b)$ are calculated. Thus gives us a lattice, through which the overall entropy minimizing path is selected to determine the segmentation points of the observed sequence (cf. figure 2).

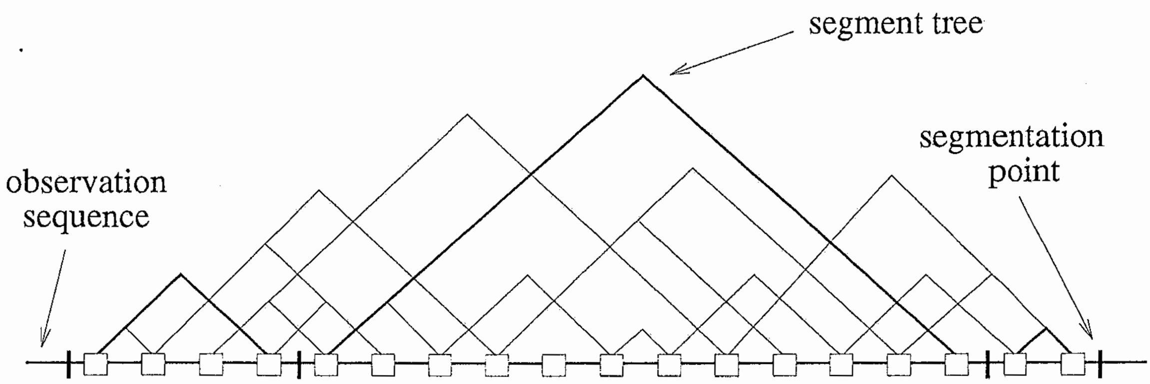

Figure 2: Determination of segmentation points

3.2 Structure Problem

These segmentation points represent the left and right boundaries of the trees. For each tree we have a root node r, which spans $e_r^- = (t_{r_1}\ t_{r_2})$ of the observation sequence. But until now we don't know anything about the structure of the tree, i.e. the identity of the descendants of r.

If r is no terminal node ($r_2 - r_1 > 1$), then it expands into two nodes $x = n(r_1, r_s)$ and $v = n(r_s, r_2)$, whereas $r_1 \leq r_s \leq r_2$. Using a prior distribution vector π for the root node r the outer evidence of the two daughter nodes are calculated (cf. section 3.3) and propagated top-down through the tree spanned by r. Using these π-vectors and the Λ-vectors used in the segmentation phase, the evidence of each node can be calculated as follows:

$$E(u) = -\log_2(\Lambda(u) \cdot \pi(u))$$

The division point is selected, by trying all possibilities of splitting the spanned observation sequence into two parts and minimizing the sum $E(x) + E(v)$ of evidences of each pair of daughter-nodes x and v. This approach is applied top-down, starting at the root node r, until the structure of the complete tree is found (cf. figure 3).

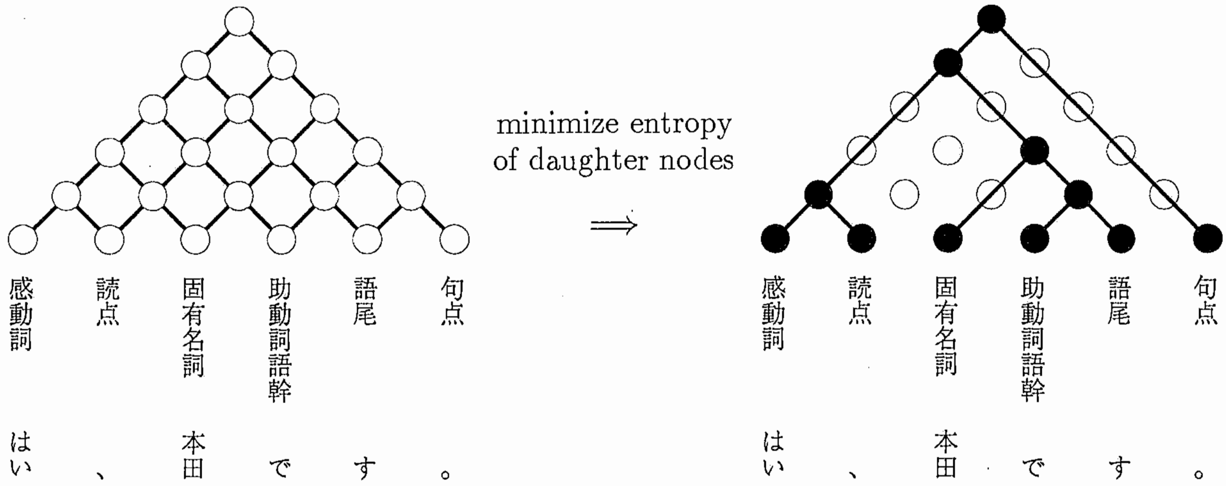

Figure 3: Calculation of the tree structure

3.3 Assignment Problem

In this probabilistic framework, the assignment task for each node u is simply that of determining the non-terminal symbol of highest probability given the global evidence e. This is achieved by finding the vector $\text{BEL}(u) = P(u|e)$, the i-th component of this vector being defined as the probability, that node u corresponds to non-terminal symbol G_i, given the entire input sequence. In order to calculate $\text{BEL}(u)$ we use the auxiliary vectors λ and π:

$$\lambda(u) = P(e_u^-|u) \qquad \lambda(s_t) = (0, \ldots, 0, 1, 0, \ldots, 0)$$
$$\pi(u) = P(e_u^+|u) \qquad \pi(s_t) = P(s_t|e_{s_t}^+)$$

The $\lambda(u)$-vector provides us with information about the nature of u based on *inner* evidence, whereas $\pi(u)$ provides us with the same type of information, but based on *outer* evidence.

Additional, the λ- and π-vectors are defined for a terminal s_t, whereby the 1 in $\lambda(s_t)$ appears in the s_t'th position and $\pi(s_t)$ describes the probability distribution of s_t, given all other symbols except s_t.

The difference between Λ- and λ-vectors comes from the fact, that in assignment phase the probabilities are calculated given the tree structure.

The relations between probabilities of adjacent nodes are provided by the equations:

$$\begin{aligned} \lambda(u)_i &= \sum_{jk} A_{ijk}\lambda(v)_j\lambda(x)_k \\ \pi(v)_j &= \alpha\sum_{ik} A_{ijk}\pi(u)_i\lambda(x)_k \\ \pi(x)_k &= \beta\sum_{ij} A_{ijk}\pi(u)_i\lambda(v)_j \end{aligned} \qquad \begin{aligned} \lambda(z)_i &= \sum_{m} B_{im}\lambda(s_t)_m \\ \pi(z)_m &= \alpha\sum_{i} B_{im}\pi(z)_i \end{aligned}$$

where α and β are normalization constants. All the λ's and π's can be determined recursively using only local calculations and the belief vector is then given by:

$$\mathrm{BEL}(u) = \frac{\lambda(u)\pi(u)}{\lambda(u)\cdot\pi(u)}$$

These equations can be understood in the following way (cf. figure 1): $\lambda(u)$ is determined using only λ-values of daughter nodes x and v, i.e. the inside evidence e_u^- is divided into e_x^- and e_v^- and the λ-vectors can be calculated bottom-up. On the other hand, calculating $\pi(x)$ is done using the π-values (outer evidence) of the mother node u and the λ-values (inner evidence) of the sister node v. So the outside evidence e_x^+ is being divided into e_u^+ and e_v^-.

Once all λ-values have been determined, the π-vectors can be calculated from top to bottom nodes. The final equation, yielding $\mathrm{BEL}(u)$, only means that the probability that node u stands for a given non-terminal symbol is obtained through the combination of two sources of information: inner evidence given by $\lambda(u)$ and outer evidence given by $\pi(u)$ (cf. figure 4). Full syntactic context, divided into inner and outer evidence, is therefore considered.

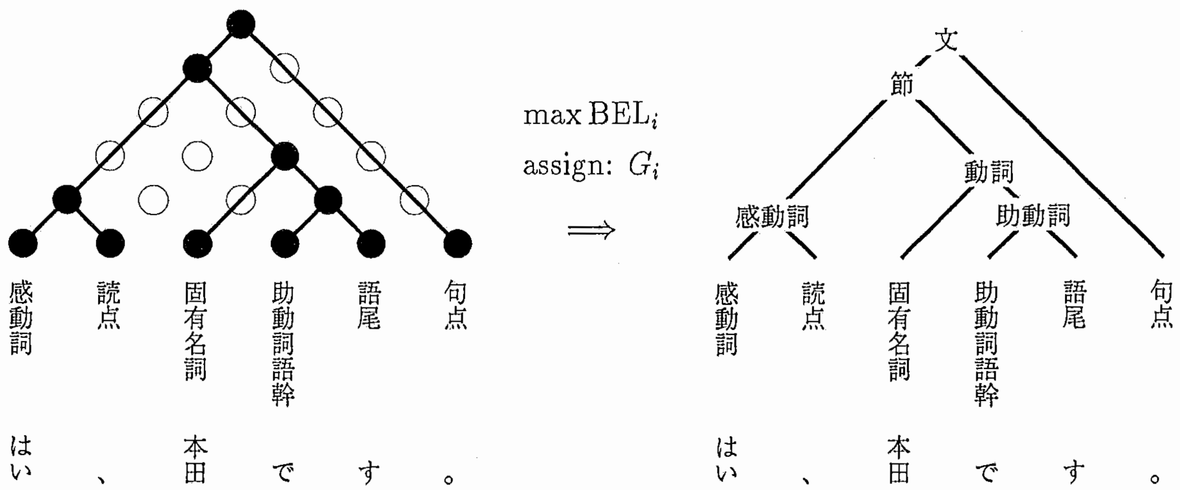

Figure 4: Assignment of grammatical categories

4 Parsing of Ill-Formedness

The BLI system forms the basis of our approach to parse ill-formedness in spoken natural language. We will describe, how the methods developed in BLI can be adapted in order to handle the kinds of ill-formedness described in section 2.

First experiments were carried out by Pierre Hudry [11]. He applied the ILLPARSE algorithm to data, whose ill-formedness was achieved by artificially changing (insertion and substitution) some parts of correct input sentences. In contrast to this hand-changed sentences, we are concerned in real data, which provides the special characteristics of spoken natural language.

But first we have to introduce some additional notations, which will be used in the rest of this report. S is denoted as the set of possible input sentences, i.e. the set of finite strings formed of terminal symbols. We call W_f the subset of sentences, for which the BLI method as we have described it in section 3 succeeds in yielding a parse; I_f is further defined as $S \setminus W_f$, the subset of sentences, which could not be parsed successfully. Let $s_o \in W_f$ be the sentence originally intended by the speaker and s the actual input sentence obtained after whatever deletions, insertions and substitutions occur.

4.1 Adaption of BLI

The BLI method uses the λ- and π-vectors only as auxiliary functions to calculate the belief vector BEL. However, combining the two sources of information provided by the inner evidence λ and the outer evidence π yields in an effective method to detect and recover the different types of ill-formedness. Before we can describe this algorithm (cf. section 4.2), we have to explain how the single steps of the BLI approach can be adapted in our system.

Whereas in BLI the segmentation points of the unlabeled data are calculated automaticly, the ILLPARSE system uses *sentence segmentation*, i.e. the segmentation points are given by the structure of the input sentences. For each input sentence one parse tree will be generated, whereby each tree is represented by its root-node.

The *structure phase* of ILLPARSE is similar to the one of BLI (cf. section 3.2). Traversing the parse tree top-downwards, for each lattice the possible splitting combinations are determined and the one, who minimizes the entropy of its daughter nodes, is selected as the division point.

In the case of $s \in I_f$, however, the system should choose a tree structure taking into account the existence of well-formed subtrees and their different probabilities. To achieve this, we introduced some noise in the probabilities attached to the different rewriting rules[3]. In order to deal with the different types of ill-formedness, the original probabilistic context-free grammar has to be altered, e.g. additional rules[4] like $G_i \rightarrow G_i\ G_\star$ or $G_i \rightarrow\ G_\star G_i$ are added with extremely low probabilities, whereby $G_\star$ can be any of the specified grammar categories. Because the assigned probabilities of these rules are extremely low, the ILLPARSE systems

[3]These modified rules are only to be used in the structure task, and *not* in the assignment task.

[4]These rules are represented by the vectors $A_{i\star i}$ and $A_{ii\star}$ of the tensor A.

obtains the same results as the BLI system, if $s \in W_f$. In the case of ill-formed input, however, a tree structure will be chosen using the greatest number of well-formed subtrees.

In the *assignment phase* of BLI the syntactic nature of the words is given by the inner evidences, i.e. the λ-vectors are propagated bottom-up. These values are then used to calculate the outer evidences π, which determines the expected non-terminals given the global sentence context.

In ILLPARSE , however, the ill-formedness can cause $P(e_u^-|u) = 0$ for a node u. Thus the bottom-up propagation of $\lambda(u)$ results in zero probability vectors for all superior nodes. This causes BEL$= 0$ and prevents us to assign categories to those nodes.

But, the partial information provided by the non-zero probability vectors, can be analyzed on a finer-grain level, in order to recover from the ill-formedness.

4.2 Detection and Repair of Ill-Formedness

Before we can describe the algorithm, we have to mention some limitations of the current ILLPARSE system. First we are restricted to the case of a *single error*. In theory multiple errors can be handled, but we haven't considered these problem yet. Furthermore we are only concerned with ill-formedness of the kind *insertion* and *substitution*, leaving aside the case of *deletions*.

Now, let us consider the tasks, which has to be performed in order to parse an ill-formed input successfully:

- detection of the region, where the ill-formedness is present.
- identification of the type of ill-formedness
- recovery from the ill-formedness identified in previous step

The region of the ill-formedness can be determined in a very straightforward way. The ill-formedness causes $\lambda(u) = 0$ for a node u and because of the bottom-up propagation all superior nodes will have zero λ-probabilities (cf. section 3.3). Thus our criteria for *identifying the region of ill-formedness* is to find a node u, with daughter nodes x and v, such that (cf. figures 5 and 6):

$$\lambda(u) = 0; \quad \lambda(x) \neq 0; \quad \lambda(v) \neq 0$$

Whereas the inner evidence of the ancestors of u gives us no information at all (zero λ-probabilities), the outer evidence of these nodes is still reliable. Using a prior distribution vector for the root-node the recursive formula for calculating the outer evidence takes into account the outer evidence of the mother node and the inner evidence of the sister node. But in the case of a single error the sister node has a non-zero λ-probability. So the π vectors of the ancestor nodes of u can be accurately determined, including $\pi(u)$.

The *identification of ill-formedness*, detected in the previous step, will then be performed by directly comparing the λ- and π-vectors for the group of the nodes u, x, v and analyzing coherence between their different values.

Insertion

In the case of an insertion at node v (cf. figure 5) the nodes u and x are actually the same node, i.e. if there would be no insertion, these nodes are identical. In order to hypothesize an insertion at node v, we have to check, whether the vectors $\pi(u)$ and $\lambda(x)$ are "sufficiently close" to each other. The values of $\pi(u)$ and $\lambda(x)$ should provide similar information, i.e. there should be no contradiction ($\pi(u) \cdot \lambda(x) \gg 0$). Thus we have to introduce a threshold value θ_i, which yields in the following *criteria for identifying an insertion* at node v:

$$\lambda(x) \cdot \pi(u) > \theta_i$$

The part of the input sentence, which is responsible for the ill-formedness, can then be eliminated and the parser proceeds with the calculation of the λ- and π-vectors.

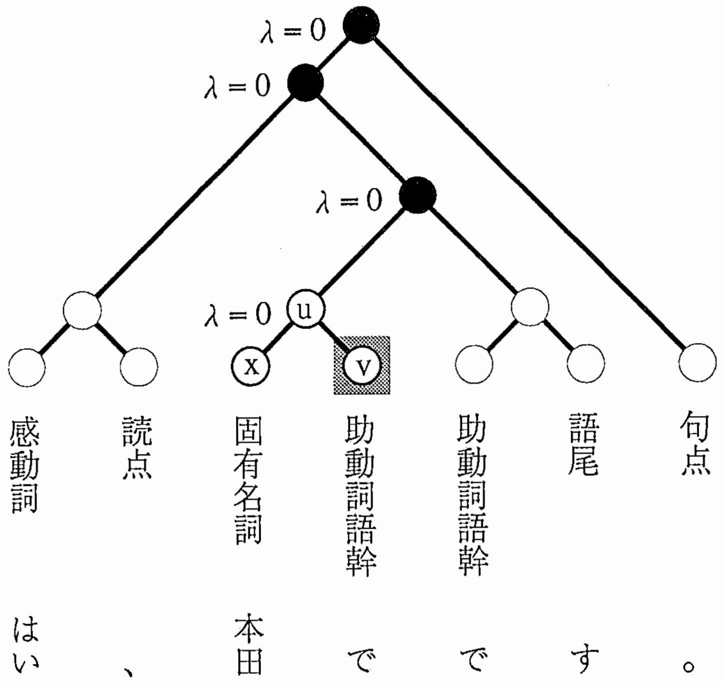

Figure 5: Insertion occuring at node v

Substitution

A substitution occuring at node x (cf. figure 6) brings false information about the inner evidence of this node. Therefore, the information provided by the inner evidence $\lambda(v)$ at node v and outside evidence $\pi(u)$ at node u are likely to contradict each other. Consequently $\pi(x) = \pi(u)\lambda(v)$ gives us no information at all ($\pi(x) \cdot \pi(x) \approx 0$). Again we introduce a threshold variable to define the following *criteria for identifying a substitution* at node x:

$$\pi(x) \cdot \pi(x) < \theta_s$$

In the assignment task we have to rely upon the outer evidence of node u, because the inner evidence fails to bring us any information. Thus we have to determine the most likely grammar category G_i, based on the information provided by the outer evidence probabilities $(\max_i \pi(u)_i)$ and assign it to node u.

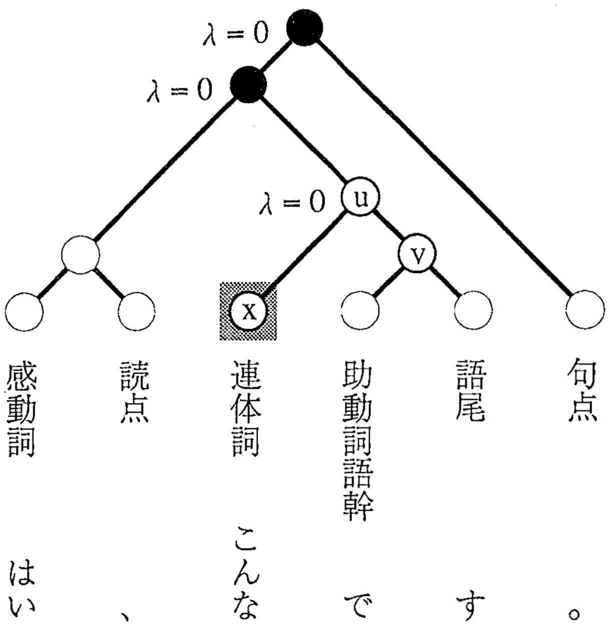

Figure 6: Substitution at node x

In order to check the validity of the recovery, the sentence has to be re-parsed, i.e. starting from node u the λ-vectors have to be propagated up to the root node of the parse tree. Using the prior distribution, as well as the new λ-values the outer evidence and the BEL-vector of each node are re-calculated. If the re-parse failed, we have to determine the next most likely grammar category, assign it to node u and re-parse it again, until the sentence is successful recovered or no valid assignment is found.

Computational Costs

It is important to note, that the computational costs of the operations to detect and recover from an ill-formedness are not any higher than the one involved in parsing well-formed input. This can be explained by the fact, that an ill-formed section of input will generate a great number of zero λ- and π-probability vectors, which all lead to trivial calculations.

This apparent reduction in costs will of course be compensated later on in the recalculation of probability vectors from the new non-zero λs- and π-vectors. But the important point here is, that dealing with ill-formed input using the described methods does not increase the computational costs tremendously.

4.3 Parsing of Artificially Changed Data

In order to prove the validity of the described approach, preliminary experiments were carried out using artificially altered, tagged data [11]. The input sentences were taken from the ATR Dialogue Database [6]. The set of input data, used in this experiments, consisted of 245 sentences, whereby its length ranged between 2 and 8 symbols.

Each of these well-formed sentences were artificially changed by randomly inserting and substituting one of the following categories:

感動詞	(kandōshi)	→	interjection
語尾	(gobi)	→	suffix
格助詞	(kakujoshi)	→	case particle

The categories 語尾 and 格助詞 were chosen, because they are strongly constrained grammatical categories, which should be easily detected as the source of the ill-formedness.

For each of the input sentences we made 7 iterations of the algorithm. First the correct, unchanged input data was parsed. Then, each of the above mentioned categories were randomly inserted (3 parses) and substituted (3 parses). Thus the altering of the the correct input data gives us a set of 735 input sentences for testing insertions and substitutions, respectively.

As described in section 4.2 the system had to detect the region of the ill-formedness, i.e. the category, we used to modify the input sentence, and to identify the type of the present ill-formedness. If an insertion was detected, we recover the original sentence by deleting the ill-formed branch of the parse tree. In the case of substitution we have to rely on the outer evidence of the detected node, i.e we assign the category with max. probability, given the outer evidence, and re-parse the sentence in order to update the assignment of the zero probability nodes in the parse tree.

The results of the preliminary experiments are summarized in Table 1.

	Changed to Well-Form (% of parsed)	Detected (% of failed to parse)	Recovered (% of failed to parse)
Insertion	24.19 %	87.75 %	59.46 %
Substitution	20.14 %	82.29 %	68.65 %
Total	22.16 %	85.01 %	64.05 %

Table 1: Results using artificially changed data

Out of the 735 input sentence for insertion 180 sentence (24.19%) were parsed correctly, i.e. the ill-formed inputs were changed to well-formedness. For the remaining 555 sentences the system failed to parse the input. For 527 of the failed sentences (87.75%) an insertion was identified as the type of ill-formedness and in 330 cases (59.46%) the system succeeded in recovering the exact parse tree, i.e. a sentence structure identical to the original input data was found.

In the case of substitution 148 sentences (20.14%) were changed to well-formedness. Out of the remaining 587 input sentences 468 (82.29%) were detected as a substitution and 403 (68.65%) were recovered identical to the unchanged data.

After describing the results of the experiments, we have to mention the limitations of our approach, too. First of all, only a single error is allowed, i.e. only one ill-formedness is introduced at one time for each input sentence. Another limitation is, that we are only concerned in the detection and recovery of insertions and substitutions, leaving out the case of deletion and unrecognized symbols.

4.4 Parsing of Spontaneous Speech

Starting with the results of the preliminary experiments we are now concerned in parsing spontaneous speech. The "real data", also taken from the ATR Dialogue Database, is provided by a part-of-speech tagger, whereby the ill-formed parts of the utterances are marked, using the following meta-characters:

[]	→	interjection	{}	→	overlap
()	→	repair	<>	→	comment

Our experiments are focused on the detection and recovery of the first two types of marked ill-formedness. These characters are eliminated from the input sequence during the reading of the data. Thus the input of our system consist of a sequence of known grammatical categories, just as in the experiments described above. But the information, provided by the meta-characters, can be used in order to analyze the results of our algorithm.

In contrast to the preliminary experiments, there are no substitutions marked in our corpus, only insertions. The kind of ill-formedness, which has to be detected, is therefore limited to *filled-pause*, *false start*, *interjection* and *correction*.

The corpus of our experiments consists of 2311 sentences. Because the input of our system requires a sequence of grammatical categories, we have to abstract from the symbolic level of the sentences. Thus the number of the associated category sequences decreases to 1960 unique input sequences. Out of them we uses 1399 utterances, its length ranging between 2 and 17, to extract single errors, yielding in 595 well-formed input sequences, 690 interjections and 114 repairs.

In order to elucidate the recovery mechanism, we illustrate our approach (cf. section 4.2) given the two examples in figure 8 and 7.

In the trivial case of an *interjection* the category 間投詞 (kantōshi) is detected as the source of the ill-formedness, occuring in the second position. Again, by eliminating the ill-formed branch of the parse tree and re-calculating the BEL-vector for each node, the correct sentence can be recovered.

In the case of the *repair* the ill-formedness, introduced by the category 連体詞 (rentaishi), is detected on the left side of the lowest node marked with "$\lambda = 0$". After eliminating the

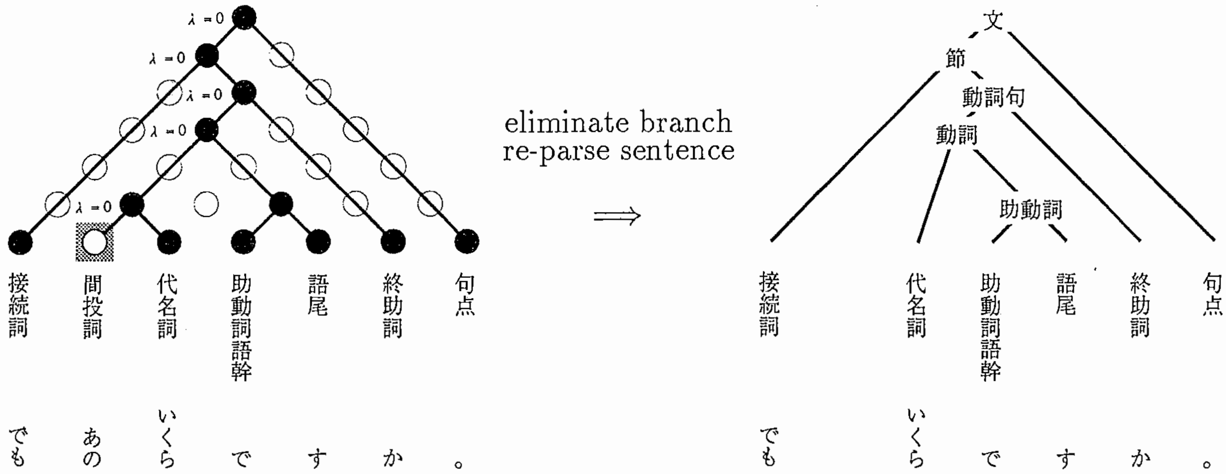

Figure 7: Recover from ill-formedness of type *interjection*

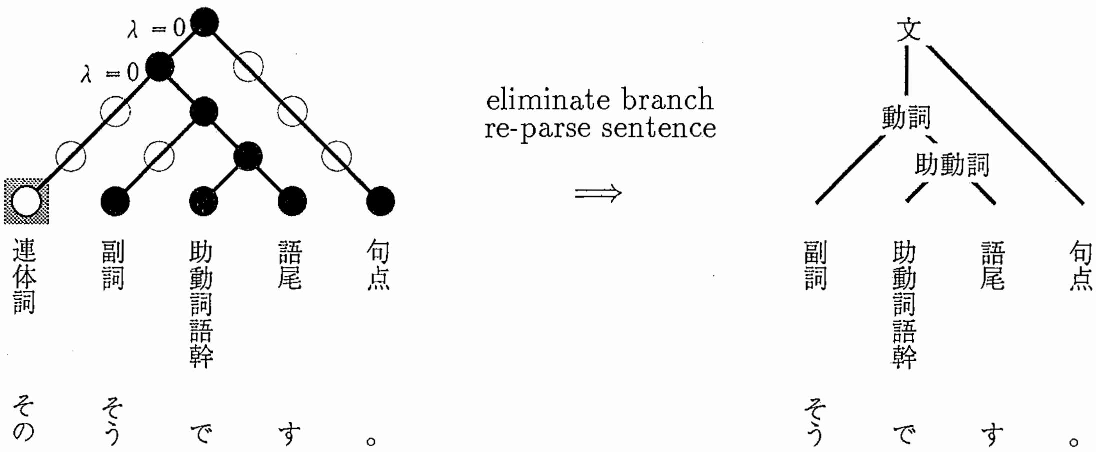

Figure 8: Recover from ill-formedness of type *repair*

left branch the re-propagation of the λ- and π-values yields in recovering the correct sentence structure.

The results of our experiments are summarized in Table 2. In the case of well-formed input sentences a success rate of parsing correctly of 61.34% were yielded. In the remaining 38.66% an ill-formedness was erroneously detected.

In the case of ill-formed input sentences we distinguish between trivial cases, i.e. the insertion of the category 間投詞, and non-trivial cases (e.g. repairs). Out of the 755 trivial cases none was changed to well-formedness. A correct detection was done in 53.38% of the ill-formed input and in 2.65% the detection was wrong. The non-trivial cases consist only of 49 examples, out of which only 11 sentences (22.45%) were detected correct. Besides 11 sentences (20.40%) were wrongly detected and 23 cases (46.94%) are changed to well-form.

Thus gives us a total rate of correct detected insertion of 53.00% and a failure of the identification of the ill-form type in 13.33%. But all detected insertions, even the wrong

Well – Formed Input

Good	Failed
61.34%	38.66%

Ill – Formed Input

	Trivial (間投詞)	Non-Trivial	Total
Changed to Well-Form	—	46.94%	2.86%
Correct Detected	53.38%	22.45%	53.00%
Wrong Detected	2.65%	20.40%	13.44%

Table 2: Results using spontaneous speech data

detected ones, could be recovered to well-form sentence structure.

The detection of 113 substitutions was wrong in each case, because there are no substitutions marked in our spontaneous speech data. But even if the detection was wrong, the system yielded to recover 75 sentences (66.37%).

5 Implementation Details

In this section we describe some characteristics of the implementation of our approach, which should be mentioned for those, who are willing to continue the work described in this report.

Data

The input data is part of the ATR Dialogue Database and is provided by a part-of-speech tagger. The following examples (cf. figure 7) shows the format of our data:

```
5|930|2050|11980|でも|デモ|でも|接続詞|||||
5|930|2060|11990|「||「||||||
5|930|2060|12000|あの|アノ|あの|間投詞|||||
5|930|2060|12010|」||」||||||
5|930|2070|12020|いくら|イクラ|いくら|代名詞|||||
5|930|2070|12030|で|デ|で|助動詞|特殊サ|語幹|||
5|930|2070|12040|す|ス|す|語尾|特殊サ|終止|||
5|930|2070|12050|か|カ|か|終助詞|||||
5|930|2070|12060|。||。|記号|||||
```

In order to handle Japanese characters correct, ILLPARSE requires the EUC-format for the data-files. We used the UNIX-command "nkf -e" for changing the original data[5] to the required file-format.

Grammar

The grammar used in our approach is a probabilistic context-free grammar (PCFG) in *Chomsky normal form*, which consists of 412 rewriting rules (cf. Appendix C). The original grammar was not in Chomsky normal form, so we had to transform the grammar to the required format by introducing additional "non-terminals" (symbol_G251, ..., symbol_G262).

The probabilities assigned to the rules of our PCFG are not optimal. They are directly estimated from the corpus of the ATR Dialogue Database. But , at least, these frequency counts provides us with likely values for the rewriting rules.

In contrast to the basic work, described in [11], the meaning of the categories used in the grammar differs slightly from those used in the spontaneous speech data. On the one side the differentiation between the categories 使役助動詞語幹 (shiekijodōshigokan) and 受身助動詞語幹 (ukemijodōshigokan) in the grammar, can not be found in the data. So we have to adapt the grammar by mapping both categories to the more general category 助動詞語幹 (jodōshigokan). On the other side the part-of-speech tagger assigned some categories to the corresponding part of the utterance, which are too general to be handled by the parser. Therefore we have to change the categories 助動詞 (jodōshi) and 補助動詞 (hojodōshi) to the more specific ones 助動詞語幹 (jodōshigokan) and 補助動詞語幹 (hojodōshigokan).

Another adaption is concerned with the category 記号 (kigō). During the tagging process this category is used as well for "。" and "、", as for other special characters, like "？"," %", etc. But in the grammar the symbols "。" and "、" are assigned to the categories 句点 and 読点, respectively.

The changes of the data takes place during the reading of the input data. The changes of the categories can be summarized as follows:

<u>*data*</u>				<u>*grammar*</u>	
助動詞	→	助動詞語幹			
補助動詞	→	補助動詞語幹	使役助動詞語幹	→	助動詞語幹
記号	→	句点 for "。"	受身助動詞語幹	→	助動詞語幹
記号	→	読点 for "、"			

As described earlier we are not working on the symbolic sentence level, but using a sequence of grammar categories as basic patterns. In the implementation described in [11] there is no differentiation between a non-terminal, used in the input sequence, and the one assigned to a non-leaf node in the parse tree. There the same symbol is used in both cases. In order to clear the notation we introduce an additional naming convention, by adding an asterix ("*") at the end of the categories used in the input sequences (cf. Appendix A and B). Thus we

[5] ~mizu/HUMAN_INTERPRETER/TAGGED_DATA

have to introduce explicit rules of the form "$G \longrightarrow G* : 1.0$" for all input categories and add them to the grammar.

Parsing

One limitation of our approach is the restriction to a single error. But due to the characteristics of spontaneous speech, there are multiple errors occuring in the data. Thus we have to eliminate all, but one ill-formedness in each input sentence.

In order to identify the type of the detected ill-formedness the threshold-values θ_i and θ_s are not used explicitly. Instead a multiplication factor is used for comparing the probabilities of the respective ill-type. Analyzing the results of the preliminary experiments we refined this factor, yielding better results for the parsing of natural dialogue utterances.

6 Discussion and Conclusion

Comparing the results of the preliminary experiments (cf. section 4.3) and the ones with spontaneous speech data (cf. section 4.4), there are some remarks, which has to be mentioned.

In [21] the average rate of "changes to well-formed" is reported as 10%. In our experiments only 2.86% of parsing an ill-formed input yields in a good parse. But because substitutions are not marked in our data, we only take into account the case of insertions. The high percentage (22.16%) found in the preliminary experiments is due to the randomly introducing of the ill-formedness.

Concerning the case of ill-formed input we had a high percentage of trivial insertions of the category 間投詞 (755 sentences). Only 49 examples of non-trivial cases of insertions could be extracted from the corpus. Out of them only 22.45% could be detected correct. This is due to high percentage of changes to well-formedness for these ill-formed input sentences (46.94%).

For 13.44% of the input sentences an ill-formedness of type insertion was erroneously detected. But all these cases were well-formed. Thus the wrong detection was due to the failure of the parser and not to identification method. Also most of the detections of a substitution are due to a failure of the parser, i.e. a well-formed input sentence failed to be parse. Only in 30 cases a substitution was identified for an ill-formed (insertion) input sequence, i.e. in 13.76% of the wrong detected sentences. But with increasing length of the input sentence the failing rate of the parser decreases (cf. figure 9).

Because there are no substitutions marked in the corpus, all detections of an ill-formedness of type *substitution* were wrong. But the recovery rate for these sentences (66.37%) corroborate the results, found in the previous experiments (68.65%).

One possible improvement of our approach is concerned with the rule probabilities of our PCFG. As mentioned before these are only frequency counts, estimated from the corpus and thus not optimal. In order to optimize these parameters we can adapt the BLI algorithm, to take into account the sentence boundaries of our input sequences. Using the training

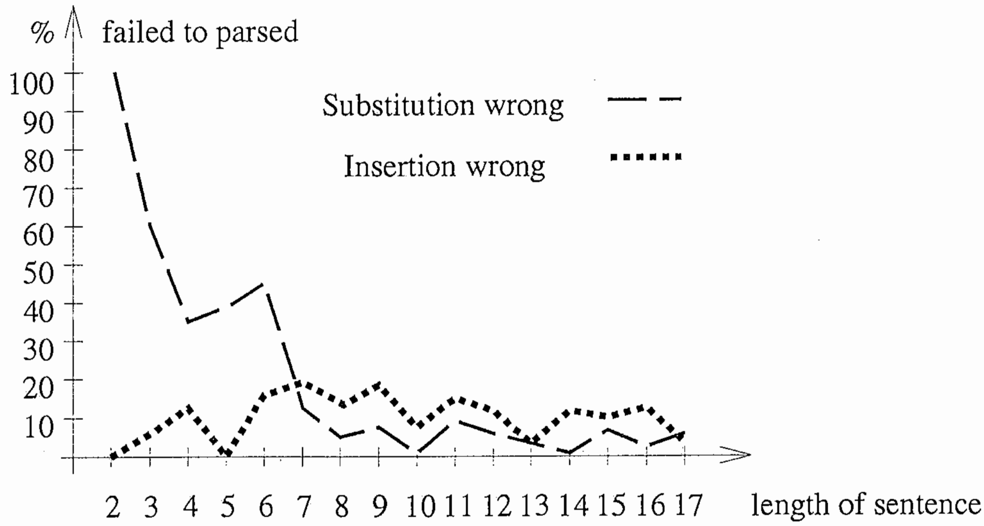

Figure 9: Wrong detection of substitution and insertion

feature of the original BLI approach, we should obtain optimal probability values for our rewriting rules and can thus reduce the percentage of erroneous parses, as well in the cases of ill-formed sentences (change to well-formedness and wrong detection of the ill-type), as in well-formed input (fail to parse).

Besides, the PCFG, we presently use, should be revised. In case of short input sequences the grammar is not specific enough, failing to find an applicable rule, especially in the case of short input sentence (cf. figure 9). Thus the system failed to parse the sentence.

Another task for the future is to get ride of the limitation of a single error. In order to parse spontaneous speech in a reasonable way, the handling of multiple errors is indispensable. The problem in the case of multiple error is, that the outer evidence of all nodes in the parse tree can't be calculated top-downwards (cf. section 3.3) any longer. The multiple error can cause two daughter nodes to have zero λ probabilities, resulting in zero outer evidences for these nodes, when propagating the π-vectors top-down in the structure task. One solution to this problem is to use prior distribution vectors not only for the root node of the parse tree, but also assigning such a vector to all tree nodes. These distribution vectors should, at least, depend upon the number of symbols it covers and the relative position in the tree. In the case of multiple errors, we can use these additional information to calculate the outer evidence of the respective nodes and proceed with the algorithm described above. The recursive application of this approach should yield in reasonable results.

Until now there is only a Japanese grammar available in our implementation. Thus we have to extract only the Japanese part of our natural conversation data, leaving the English part aside. Because of the more complex sentence structure in English utterance it would be an interesting enhancement of our approach to get hold of an English grammar and apply our algorithm to the English part of the ATR Dialogue Database.

References

[1] Black, E., Parsing English by Computer: The State of the Art, Proceedings of the ATR International Workshop on Speech Translation, 1993.

[2] Charniak, E., Statistical Language Learning, MIT Press, 1993

[3] Chomsky, N., On Certain Formal Properties of Grammars, *Information and Control*, No.2, pp. 137–167, 1959.

[4] Clark, H.H., Wasow, T., Repeated Words in Spontaneous Speech - A Preliminary Report, Center for the Study of Language and Information, Stanford University, 1994.

[5] Den, Y., Generalized Chart Algorithm: An Efficient Procedure for Cost-Based Abduction, *Proceedings of 32nd Annual Meeting of the ACL*, pp. 218–225, 1994.

[6] Ehara, E., et al. Contents of the ATR Dialogue Database, ATR Technical Report TR-I-0186. Kyoto, Japan: ATR Interpreting Telephony Laboratories, 1990.

[7] Fais, L., Non-grammatical Phenomena in Real English Conversation, ATR Technical Report TR-IT-0007. Kyoto, Japan: ATR Interpreting Telecommunications Laboratories, 1993.

[8] Fais, L., Structures in Spontaneous English Conversation, ATR Technical Report TR-IT-0040. Kyoto, Japan: ATR Interpreting Telecommunications Laboratories, 1994.

[9] Heckerman, D., and Wellman, M. Bayesian Networks, Communications of the ACM, Vol. 38, No. 3, 1995

[10] Hobbs, J.R., Stickel, M.E., Appelt, D.E., and Martin, P., Interpretation as Abduction, *Artificial Intelligence* 63, pp. 69–142, 1993.

[11] Hudry, P., and Den, Y., A Statistical Approach to Parsing Ill-Formed Input, ATR Technical Report TR-IT-0076. Kyoto, Japan: ATR Interpreting Telecommunications Laboratories, 1994.

[12] Jelinek, F., Up from trigrams!, Proceedings Eurospeech, pp. 1037–1040, 1991.

[13] Jelinek, F., Lafferty, J.D., and Mercer, R.L., Basic Methods of Probabilistic Context Free Grammars, Continuous Speech Recognition Group IBM T.J. Watson Research Center, 1991.

[14] Loken-Kim, K., Fumihiro, Y., Kazuhiko, K., Fais, L., and Ryo, F., EMMI-ATR environment for multi-modal interactions, ATR Technical Report TR-IT-0018. Kyoto, Japan: ATR Interpreting Telecommunications Laboratories, 1993.

[15] Labov, W., *Sociolinguistic patterns*, Oxford: Basil Blackwell, 1972.

[16] Lucke, H., A Method for Inferring Stochastic Context-free Grammars Using the Theory of Bayesian Causal Trees, Proccedings of the Institute of Electronics, Information and Communication Engineers, Technical Report of ASJ Speech Commitee, SP92–113, pp. 79–86, 1992.

[17] Lucke, H., On the Applicability of Bayesian Belief Network to Language Modeling in Speech Recognition, ATR Technical Report TR-IT-0028, Kyoto, Japan: ATR Interpreting Telecommunications Laboratories, 1993.

[18] Marcus, M., Statistical Natural Language Processing: Current Trends and Future Directions, Proceedings of the ATR International Workshop on Speech Translation, 1993.

[19] Niyi Akinnaso, F., On the Differences Between Spoken and Written Language, *Language and Speech*, Vol. 25, Part 2, pp. 97–125, 1982.

[20] Pearl, J., *Probabilistic Reasoning in Intelligent Systems - Networks of Plausible Inference*, Morgan and Kaufmann, 1987.

[21] Sagawa, Y., Ohnishi, N., Sugie, N., A Parser Coping with Self-Repaired Japanese Utterances and Large Corpus-Based Evaluation, pp. 593–597, 1993.

[22] Schabes, Y., Stochastic Lexicalized Tree-adjoining Grammars, Proceedings of the 16th Intrnational Conference on Computational Linguistics COLING'92, pp. 426–432, 1992.

[23] Schiffrin, D., *Discourse markers*, Cambridge: Cambridge University Press, 1987.

[24] Seligman, M., and Boitet, C., A Whiteboard Architecture for Automatic Speech Translation. *Proceedings of the International Symposium on Spoken Dialogue '93*, pp. 243-246, 1993.

A Terminal Symbols

サ変名詞 *
引用助詞 *
格助詞 *
感動詞 *
間投詞 *
記号 *
係助詞 *
形容詞 *
形容名詞 *
固有名詞 *
語尾 *
終助詞 *
住所名 *
準体助詞 *
助動詞語幹 *
人名 *
数詞 *
接続詞 *
接続助詞 *
接頭辞 *
接尾辞 *
代名詞 *
日時 *
普通名詞 *
副詞 *
副助詞 *
並立助詞 *
補助動詞語幹 *
本動詞 *
連体詞 *
連体助詞 *
読点 *
句点 *

B Non-Terminal Symbols

symbol_G251
symbol_G252
symbol_G253
symbol_G254
symbol_G255
symbol_G256
symbol_G257
symbol_G258
symbol_G259
symbol_G260
symbol_G261
symbol_G262
かぎ括弧
アンダーバー
サ変名詞
テ形補助動詞
テ形補助動詞語幹
パーセント記号
ピリオド
引用助詞
引用符
格助詞
感嘆符
感動詞
間投詞
丸括弧
記号
疑問符
句点
区画番地
係助詞
形容詞
形容名詞
固有名詞
後置詞句
語尾
使役助動詞語幹
受身助動詞語幹
終助詞
住所
住所名
住所要素
準体助詞
助動詞
助動詞語幹
人名
数詞
数詞連体詞句
数量詞
姓名
接続詞
接続助詞
接頭辞
接尾辞
節
態の助動詞
態の動詞
態の動詞句
代名詞
中黒
通貨記号
等号
動詞
動詞句
読点
日時
番地連体詞句
普通名詞
副詞
副詞句
副詞節
副詞的名詞
副助詞
複合区画番地
複合語
複合数詞
複合日時
複合番地要素
文
文副詞
並立助詞
補助動詞
補助動詞語幹
本動詞
名詞句
名詞節
連体詞
連体詞句
連体修飾節
連体助詞
連用修飾

C Grammar Rules

symbol_G251	→	数詞	丸括弧	:	1.00000000
symbol_G252	→	普通名詞	引用符	:	1.00000000
symbol_G253	→	複合語	引用符	:	1.00000000
symbol_G254	→	アンダーバー	人名	:	1.00000000
symbol_G255	→	中黒	人名	:	1.00000000
symbol_G256	→	等号	人名	:	1.00000000
symbol_G257	→	サ変名詞	読点	:	1.00000000
symbol_G258	→	普通名詞	格助詞	:	1.00000000
symbol_G259	→	ピリオド	人名	:	1.00000000
symbol_G260	→	symbol_G259	ピリオド	:	1.00000000
symbol_G261	→	symbol_G260	人名	:	1.00000000
symbol_G262	→	固有名詞	かぎ括弧	:	1.00000000
サ変名詞	→	サ変名詞	読点	:	0.01265823
サ変名詞	→	接頭辞	symbol_G257	:	0.00632911
サ変名詞	→	接頭辞	サ変名詞	:	0.98101266
テ形補助動詞	→	テ形補助動詞語幹	語尾	:	1.00000000
テ形補助動詞語幹	→	接続助詞	補助動詞語幹	:	1.00000000
引用助詞	→	引用助詞	引用助詞	:	0.06250000
引用助詞	→	引用助詞	読点	:	0.93750000
格助詞	→	格助詞	格助詞	:	0.11627907
格助詞	→	格助詞	読点	:	0.88372093
感動詞	→	サ変名詞	感動詞	:	0.03004292
感動詞	→	感動詞	読点	:	0.62017167
感動詞	→	後置詞句	感動詞	:	0.03648069
感動詞	→	副詞	感動詞	:	0.25536481
感動詞	→	副詞句	感動詞	:	0.04721030
感動詞	→	名詞句	感動詞	:	0.01072961
区画番地	→	数詞	接尾辞	:	1.00000000
係助詞	→	係助詞	読点	:	1.00000000
形容詞	→	接頭辞	形容詞	:	1.00000000
形容名詞	→	接頭辞	形容名詞	:	1.00000000
後置詞句	→	サ変名詞	格助詞	:	0.05732689
後置詞句	→	サ変名詞	係助詞	:	0.01417069
後置詞句	→	サ変名詞	副助詞	:	0.00128824
後置詞句	→	固有名詞	格助詞	:	0.03832528
後置詞句	→	固有名詞	係助詞	:	0.00161031
後置詞句	→	後置詞句	係助詞	:	0.05088567
後置詞句	→	後置詞句	副助詞	:	0.00032206
後置詞句	→	人名	格助詞	:	0.00032206
後置詞句	→	数量詞	格助詞	:	0.01449275
後置詞句	→	数量詞	係助詞	:	0.00064412
後置詞句	→	姓名	格助詞	:	0.00354267
後置詞句	→	節	引用助詞	:	0.01159420
後置詞句	→	態の動詞句	引用助詞	:	0.00096618
後置詞句	→	代名詞	格助詞	:	0.04573269
後置詞句	→	代名詞	係助詞	:	0.02673108
後置詞句	→	代名詞	副助詞	:	0.00418680
後置詞句	→	動詞	symbol_G258	:	0.00032206
後置詞句	→	動詞	引用助詞	:	0.01449275
後置詞句	→	動詞	格助詞	:	0.00096618
後置詞句	→	動詞句	引用助詞	:	0.02705314
後置詞句	→	動詞句	副助詞	:	0.00032206
後置詞句	→	日時	格助詞	:	0.00837359
後置詞句	→	日時	係助詞	:	0.00032206
後置詞句	→	普通名詞	格助詞	:	0.15169082
後置詞句	→	普通名詞	係助詞	:	0.04090177
後置詞句	→	普通名詞	副助詞	:	0.00193237
後置詞句	→	複合語	格助詞	:	0.06312399
後置詞句	→	複合語	係助詞	:	0.02769726
後置詞句	→	複合語	副助詞	:	0.00128824
後置詞句	→	複合数詞	格助詞	:	0.00032206
後置詞句	→	複合日時	格助詞	:	0.00740741
後置詞句	→	名詞句	格助詞	:	0.28276973
後置詞句	→	名詞句	係助詞	:	0.08566828
後置詞句	→	名詞句	副助詞	:	0.00515298
後置詞句	→	名詞節	格助詞	:	0.00450886
後置詞句	→	名詞節	係助詞	:	0.00354267
語尾	→	語尾	読点	:	1.00000000
終助詞	→	終助詞	読点	:	1.00000000
住所	→	住所	住所名	:	0.27272727
住所	→	住所	姓名	:	0.09090909
住所	→	住所	複合区画番地	:	0.03030303
住所	→	住所	複合番地要素	:	0.27272727
住所	→	住所名	住所名	:	0.03030303

住所	→	住所要素	住所名	:	0.03030303
住所	→	住所要素	住所要素	:	0.27272727
住所名	→	住所名	読点	:	1.00000000
住所要素	→	住所名	接尾辞	:	1.00000000
助動詞	→	助動詞	読点	:	0.00438917
助動詞	→	助動詞語幹	語尾	:	0.99561083
人名	→	人名	読点	:	1.00000000
数詞	→	丸括弧	symbol_G251	:	0.12500000
数詞	→	数詞	読点	:	0.37500000
数詞	→	接頭辞	数詞	:	0.50000000
数詞連体詞句	→	数詞	記号	:	0.54166667
数詞連体詞句	→	複合数詞	記号	:	0.45833333
数量詞	→	数詞	パーセント記号	:	0.00977199
数量詞	→	数詞	接尾辞	:	0.62540717
数量詞	→	数詞	通貨記号	:	0.00325733
数量詞	→	数詞	普通名詞	:	0.03583062
数量詞	→	数量詞	接尾辞	:	0.16286645
数量詞	→	数量詞	副助詞	:	0.08143322
数量詞	→	接頭辞	数量詞	:	0.04885993
数量詞	→	通貨記号	数詞	:	0.01302932
数量詞	→	普通名詞	数量詞	:	0.01628664
数量詞	→	副詞	数量詞	:	0.00325733
姓名	→	人名	symbol_G254	:	0.04166667
姓名	→	人名	symbol_G255	:	0.20833333
姓名	→	人名	symbol_G256	:	0.14583333
姓名	→	人名	symbol_G261	:	0.02083333
姓名	→	人名	人名	:	0.58333333
接続詞	→	接続詞	読点	:	1.00000000
接続助詞	→	接続助詞	読点	:	1.00000000
接尾辞	→	接尾辞	接尾辞	:	0.42307692
接尾辞	→	接尾辞	読点	:	0.57692308
節	→	感動詞	感動詞	:	0.05551020
節	→	感動詞	節	:	0.04244898
節	→	感動詞	動詞	:	0.04571429
節	→	感動詞	動詞句	:	0.11102041
節	→	接続詞	感動詞	:	0.02693878
節	→	接続詞	節	:	0.03346939
節	→	接続詞	動詞	:	0.01469388
節	→	接続詞	動詞句	:	0.16408163
節	→	動詞	節	:	0.00081633
節	→	動詞	動詞	:	0.00244898
節	→	動詞	動詞句	:	0.00897959
節	→	動詞句	感動詞	:	0.00489796
節	→	動詞句	節	:	0.00163265
節	→	動詞句	動詞	:	0.00081633
節	→	動詞句	動詞句	:	0.02204082
節	→	副詞節	感動詞	:	0.02775510
節	→	副詞節	節	:	0.04408163
節	→	副詞節	動詞	:	0.01877551
節	→	副詞節	動詞句	:	0.32489796
節	→	文副詞	感動詞	:	0.00081633
節	→	文副詞	節	:	0.00408163
節	→	文副詞	動詞	:	0.00244898
節	→	文副詞	動詞句	:	0.04163265
態の助動詞	→	助動詞語幹	語尾	:	1.00000000
態の動詞	→	サ変名詞	補助動詞	:	0.43478261
態の動詞	→	本動詞	語尾	:	0.56521739
態の動詞句	→	後置詞句	態の動詞句	:	0.53960396
態の動詞句	→	態の動詞	助動詞語幹	:	0.21287129
態の動詞句	→	態の動詞	態の助動詞	:	0.12871287
態の動詞句	→	副詞	態の動詞句	:	0.10396040
態の動詞句	→	本動詞	助動詞語幹	:	0.01485149
代名詞	→	接頭辞	代名詞	:	0.55555556
代名詞	→	代名詞	読点	:	0.44444444
動詞	→	サ変名詞	形容名詞	:	0.00017271
動詞	→	サ変名詞	助動詞	:	0.00086356
動詞	→	サ変名詞	動詞	:	0.00155440
動詞	→	形容詞	語尾	:	0.02797927
動詞	→	固有名詞	助動詞	:	0.00811744
動詞	→	後置詞句	サ変名詞	:	0.05250432
動詞	→	後置詞句	形容名詞	:	0.00777202
動詞	→	後置詞句	助動詞	:	0.00483592
動詞	→	後置詞句	動詞	:	0.36062176
動詞	→	後置詞句	本動詞	:	0.05647668

動詞	→	住所	助動詞	: 0.00189983
動詞	→	人名	助動詞	: 0.00120898
動詞	→	数詞	助動詞	: 0.00086356
動詞	→	数量詞	サ変名詞	: 0.00155440
動詞	→	数量詞	形容名詞	: 0.00034542
動詞	→	数量詞	助動詞	: 0.00656304
動詞	→	数量詞	動詞	: 0.00777202
動詞	→	数量詞	本動詞	: 0.00051813
動詞	→	姓名	助動詞	: 0.00138169
動詞	→	代名詞	助動詞	: 0.00777202
動詞	→	代名詞	動詞	: 0.00794473
動詞	→	日時	助動詞	: 0.00017271
動詞	→	普通名詞	助動詞	: 0.00725389
動詞	→	普通名詞	動詞	: 0.00241796
動詞	→	普通名詞	本動詞	: 0.00034542
動詞	→	副詞	サ変名詞	: 0.00708117
動詞	→	副詞	形容名詞	: 0.00207254
動詞	→	副詞	助動詞	: 0.01381693
動詞	→	副詞	動詞	: 0.06113990
動詞	→	副詞	本動詞	: 0.00846287
動詞	→	副詞句	サ変名詞	: 0.00431779
動詞	→	副詞句	形容名詞	: 0.00069085
動詞	→	副詞句	助動詞	: 0.00120898
動詞	→	副詞句	動詞	: 0.03419689
動詞	→	副詞句	本動詞	: 0.00189983
動詞	→	複合語	サ変名詞	: 0.00017271
動詞	→	複合語	助動詞	: 0.00310881
動詞	→	複合語	動詞	: 0.00034542
動詞	→	複合数詞	助動詞	: 0.00207254
動詞	→	複合日時	助動詞	: 0.00051813
動詞	→	本動詞	語尾	: 0.22987910
動詞	→	名詞句	助動詞	: 0.03609672
動詞	→	名詞句	動詞	: 0.00811744
動詞	→	名詞句	本動詞	: 0.00069085
動詞	→	連用修飾	サ変名詞	: 0.00207254
動詞	→	連用修飾	形容名詞	: 0.00017271
動詞	→	連用修飾	動詞	: 0.00863558
動詞	→	連用修飾	本動詞	: 0.00431779
動詞句	→	サ変名詞	補助動詞	: 0.01087866
動詞句	→	サ変名詞	補助動詞語幹	: 0.00125523
動詞句	→	形容詞	助動詞	: 0.00041841
動詞句	→	形容名詞	助動詞	: 0.01213389
動詞句	→	態の動詞句	助動詞	: 0.00711297
動詞句	→	態の動詞句	助動詞語幹	: 0.00104603
動詞句	→	態の動詞句	補助動詞	: 0.00146444
動詞句	→	態の動詞句	補助動詞語幹	: 0.00020921
動詞句	→	動詞	係助詞	: 0.00167364
動詞句	→	動詞	終助詞	: 0.03054393
動詞句	→	動詞	助動詞	: 0.20669456
動詞句	→	動詞	助動詞語幹	: 0.02029289
動詞句	→	動詞	補助動詞	: 0.16610879
動詞句	→	動詞	補助動詞語幹	: 0.02008368
動詞句	→	動詞句	テ形補助動詞	: 0.00020921
動詞句	→	動詞句	係助詞	: 0.00167364
動詞句	→	動詞句	終助詞	: 0.10941423
動詞句	→	動詞句	助動詞	: 0.30376569
動詞句	→	動詞句	助動詞語幹	: 0.01255230
動詞句	→	動詞句	副助詞	: 0.00020921
動詞句	→	動詞句	補助動詞	: 0.01589958
動詞句	→	動詞句	補助動詞語幹	: 0.00292887
動詞句	→	副詞節	助動詞	: 0.00041841
動詞句	→	本動詞	助動詞	: 0.00376569
動詞句	→	本動詞	助動詞語幹	: 0.00020921
動詞句	→	本動詞	補助動詞	: 0.00209205
動詞句	→	本動詞	補助動詞語幹	: 0.00083682
動詞句	→	名詞節	助動詞	: 0.06610879
日時	→	数詞	接尾辞	: 0.44444444
日時	→	日時	接尾辞	: 0.11111111
日時	→	日時	普通名詞	: 0.11111111
日時	→	普通名詞	日時	: 0.33333333
番地連体詞句	→	数詞	記号	: 0.35714286
番地連体詞句	→	数詞	連体助詞	: 0.28571429
番地連体詞句	→	複合番地要素	記号	: 0.28571429
番地連体詞句	→	複合番地要素	連体助詞	: 0.07142857
普通名詞	→	サ変名詞	接尾辞	: 0.00904977
普通名詞	→	数量詞	普通名詞	: 0.04524887
普通名詞	→	接頭辞	サ変名詞	: 0.00452489
普通名詞	→	接頭辞	普通名詞	: 0.59728507
普通名詞	→	代名詞	接尾辞	: 0.15837104
普通名詞	→	普通名詞	接尾辞	: 0.16289593
普通名詞	→	普通名詞	読点	: 0.02262443
副詞	→	副詞	読点	: 1.00000000
副詞句	→	サ変名詞	接尾辞	: 0.00983607
副詞句	→	節	接続助詞	: 0.00655738
副詞句	→	代名詞	接尾辞	: 0.01967213
副詞句	→	代名詞	副助詞	: 0.20655738
副詞句	→	動詞	格助詞	: 0.04590164
副詞句	→	動詞	接続助詞	: 0.08524590
副詞句	→	動詞	接尾辞	: 0.00655738
副詞句	→	動詞	副詞的名詞	: 0.05245902
副詞句	→	動詞	並立助詞	: 0.00327869
副詞句	→	動詞句	格助詞	: 0.01311475
副詞句	→	動詞句	接続助詞	: 0.04918033
副詞句	→	動詞句	副詞的名詞	: 0.06557377
副詞句	→	動詞句	並立助詞	: 0.00327869
副詞句	→	普通名詞	接尾辞	: 0.02622951
副詞句	→	普通名詞	副詞的名詞	: 0.00327869
副詞句	→	副詞	格助詞	: 0.04918033
副詞句	→	副詞	係助詞	: 0.03934426
副詞句	→	副詞	数量詞	: 0.04918033
副詞句	→	副詞	副詞	: 0.00655738
副詞句	→	副詞	副詞句	: 0.00983607
副詞句	→	副詞	副助詞	: 0.04262295
副詞句	→	副詞句	格助詞	: 0.04262295
副詞句	→	副詞句	係助詞	: 0.05573770
副詞句	→	副詞句	副助詞	: 0.00327869
副詞句	→	複合語	接尾辞	: 0.00655738
副詞句	→	本動詞	接続助詞	: 0.00983607
副詞句	→	本動詞	接尾辞	: 0.00327869
副詞句	→	連体詞	副詞的名詞	: 0.00655738
副詞句	→	連体詞句	副詞的名詞	: 0.07540984
副詞句	→	連体修飾節	副詞的名詞	: 0.00327869
副詞節	→	形容詞	接続助詞	: 0.00178571
副詞節	→	節	接続助詞	: 0.08035714
副詞節	→	動詞	接続助詞	: 0.25714286
副詞節	→	動詞句	接続助詞	: 0.64642857
副詞節	→	副詞節	格助詞	: 0.00535714
副詞節	→	副詞節	係助詞	: 0.00714286
副詞節	→	本動詞	接続助詞	: 0.00178571
副詞的名詞	→	副詞的名詞	読点	: 1.00000000
副助詞	→	副助詞	読点	: 1.00000000
複合区画番地	→	区画番地	区画番地	: 0.50000000
複合区画番地	→	複合区画番地	区画番地	: 0.50000000
複合語	→	接頭辞	複合語	: 0.50000000
複合語	→	複合語	数量詞	: 0.31250000
複合語	→	複合語	接尾辞	: 0.12500000
複合語	→	複合語	読点	: 0.06250000
複合数詞	→	数詞	数詞	: 0.04000000
複合数詞	→	数詞連体詞句	数詞	: 0.96000000
複合日時	→	日時	日時	: 0.85365854
複合日時	→	普通名詞	複合日時	: 0.02439024
複合日時	→	複合日時	日時	: 0.12195122
複合番地要素	→	番地連体詞句	数詞	: 1.00000000
文	→	感動詞	句点	: 0.14502822
文	→	節	疑問符	: 0.04385584
文	→	節	句点	: 0.38428137
文	→	動詞	句点	: 0.06990881
文	→	動詞句	感嘆符	: 0.00043422
文	→	動詞句	疑問符	: 0.08165871
文	→	動詞句	句点	: 0.29483283
文副詞	→	副詞	文副詞	: 0.06818182
文副詞	→	文副詞	読点	: 0.93181818
並立助詞	→	並立助詞	読点	: 1.00000000
補助動詞	→	補助動詞	読点	: 0.00681818
補助動詞	→	補助動詞語幹	語尾	: 0.99318182
本動詞	→	形容詞	接尾辞	: 0.00261097
本動詞	→	接頭辞	本動詞	: 0.96344648
本動詞	→	動詞	接尾辞	: 0.00261097
本動詞	→	動詞	本動詞	: 0.01044386
本動詞	→	動詞句	本動詞	: 0.01827676
本動詞	→	本動詞	読点	: 0.00261097
名詞句	→	かぎ括弧	symbol_G262	: 0.00057904
名詞句	→	サ変名詞	読点	: 0.00057904
名詞句	→	サ変名詞	副助詞	: 0.00231616

名詞句	→	引用符	symbol_G252	: 0.00173712
名詞句	→	引用符	symbol_G253	: 0.00057904
名詞句	→	形容名詞	普通名詞	: 0.00057904
名詞句	→	固有名詞	接尾辞	: 0.00057904
名詞句	→	固有名詞	副助詞	: 0.00115808
名詞句	→	人名	接尾辞	: 0.00463231
名詞句	→	人名	普通名詞	: 0.00752750
名詞句	→	数詞	副助詞	: 0.00057904
名詞句	→	数量詞	副助詞	: 0.00057904
名詞句	→	姓名	接尾辞	: 0.00289519
名詞句	→	姓名	普通名詞	: 0.00231616
名詞句	→	節	副助詞	: 0.00057904
名詞句	→	態の動詞句	サ変名詞	: 0.00115808
名詞句	→	態の動詞句	固有名詞	: 0.00289519
名詞句	→	態の動詞句	普通名詞	: 0.00231616
名詞句	→	態の動詞句	名詞句	: 0.00057904
名詞句	→	代名詞	読点	: 0.00405327
名詞句	→	代名詞	副助詞	: 0.00231616
名詞句	→	動詞	サ変名詞	: 0.00579039
名詞句	→	動詞	固有名詞	: 0.00057904
名詞句	→	動詞	数量詞	: 0.00173712
名詞句	→	動詞	普通名詞	: 0.05616676
名詞句	→	動詞	副助詞	: 0.00057904
名詞句	→	動詞	複合語	: 0.00231616
名詞句	→	動詞	名詞句	: 0.00231616
名詞句	→	動詞句	サ変名詞	: 0.00984366
名詞句	→	動詞句	固有名詞	: 0.00405327
名詞句	→	動詞句	普通名詞	: 0.10364794
名詞句	→	動詞句	副助詞	: 0.00057904
名詞句	→	動詞句	複合語	: 0.00521135
名詞句	→	動詞句	名詞句	: 0.00579039
名詞句	→	普通名詞	読点	: 0.00636943
名詞句	→	普通名詞	副助詞	: 0.00810654
名詞句	→	副詞	普通名詞	: 0.00289519
名詞句	→	複合語	読点	: 0.00231616
名詞句	→	複合語	副助詞	: 0.00463231
名詞句	→	名詞句	副助詞	: 0.01215981
名詞句	→	連体詞	サ変名詞	: 0.02663578
名詞句	→	連体詞	固有名詞	: 0.00057904
名詞句	→	連体詞	数量詞	: 0.00115808
名詞句	→	連体詞	普通名詞	: 0.06195715
名詞句	→	連体詞	複合語	: 0.00579039
名詞句	→	連体詞	複合日時	: 0.00057904
名詞句	→	連体詞	名詞句	: 0.00289519
名詞句	→	連体詞句	サ変名詞	: 0.07759120
名詞句	→	連体詞句	固有名詞	: 0.01968732
名詞句	→	連体詞句	準体助詞	: 0.00405327
名詞句	→	連体詞句	人名	: 0.00868558
名詞句	→	連体詞句	数詞	: 0.00057904
名詞句	→	連体詞句	数量詞	: 0.00463231
名詞句	→	連体詞句	姓名	: 0.00694847
名詞句	→	連体詞句	代名詞	: 0.00231616
名詞句	→	連体詞句	日時	: 0.00752750
名詞句	→	連体詞句	普通名詞	: 0.35726694
名詞句	→	連体詞句	複合語	: 0.07527504
名詞句	→	連体詞句	複合数詞	: 0.00057904
名詞句	→	連体詞句	複合日時	: 0.00173712
名詞句	→	連体詞句	名詞句	: 0.04574406
名詞句	→	連体修飾節	サ変名詞	: 0.00289519
名詞句	→	連体修飾節	普通名詞	: 0.00926462
名詞節	→	態の動詞句	準体助詞	: 0.02346041
名詞節	→	動詞	準体助詞	: 0.31085044
名詞節	→	動詞句	準体助詞	: 0.57184751
名詞節	→	連体修飾節	準体助詞	: 0.09384164
連体詞句	→	サ変名詞	並立助詞	: 0.00367309
連体詞句	→	サ変名詞	連体助詞	: 0.09366391
連体詞句	→	固有名詞	並立助詞	: 0.00642792
連体詞句	→	固有名詞	連体助詞	: 0.09366391
連体詞句	→	後置詞句	連体助詞	: 0.02571166
連体詞句	→	人名	連体助詞	: 0.00091827
連体詞句	→	数詞	連体助詞	: 0.00183655
連体詞句	→	数量詞	並立助詞	: 0.00183655
連体詞句	→	数量詞	連体助詞	: 0.01928375
連体詞句	→	姓名	並立助詞	: 0.00091827
連体詞句	→	姓名	連体助詞	: 0.00367309
連体詞句	→	態の動詞句	連体助詞	: 0.00183655
連体詞句	→	代名詞	並立助詞	: 0.00091827

連体詞句	→	代名詞	連体助詞	: 0.06427916
連体詞句	→	動詞	連体助詞	: 0.01836547
連体詞句	→	動詞句	連体助詞	: 0.01469238
連体詞句	→	日時	連体助詞	: 0.00918274
連体詞句	→	普通名詞	並立助詞	: 0.03581267
連体詞句	→	普通名詞	連体助詞	: 0.29752066
連体詞句	→	副詞	連体助詞	: 0.00459137
連体詞句	→	副詞句	連体助詞	: 0.00367309
連体詞句	→	複合語	並立助詞	: 0.01377410
連体詞句	→	複合語	連体助詞	: 0.11202938
連体詞句	→	複合日時	並立助詞	: 0.00091827
連体詞句	→	複合日時	連体助詞	: 0.00367309
連体詞句	→	名詞句	並立助詞	: 0.01285583
連体詞句	→	名詞句	連体助詞	: 0.15426997
連体修飾節	→	動詞	動詞	: 0.01754386
連体修飾節	→	動詞	動詞句	: 0.03508772
連体修飾節	→	動詞	連体修飾節	: 0.01754386
連体修飾節	→	動詞句	動詞	: 0.01754386
連体修飾節	→	動詞句	動詞句	: 0.07017544
連体修飾節	→	動詞句	連体修飾節	: 0.01754386
連体修飾節	→	副詞節	動詞	: 0.31578947
連体修飾節	→	副詞節	動詞句	: 0.40350877
連体修飾節	→	副詞節	連体修飾節	: 0.01754386
連体修飾節	→	本動詞	動詞	: 0.01754386
連体修飾節	→	連体詞句	助動詞	: 0.07017544
連体助詞	→	連体助詞	読点	: 1.00000000
連用修飾	→	サ変名詞	連用修飾	: 0.03846154
連用修飾	→	形容詞	語尾	: 0.25000000
連用修飾	→	形容名詞	助動詞	: 0.55769231
連用修飾	→	普通名詞	連用修飾	: 0.01923077
連用修飾	→	副詞	連用修飾	: 0.06730769
連用修飾	→	連体詞句	助動詞	: 0.03846154
連用修飾	→	連用修飾	係助詞	: 0.02884615
サ変名詞	→	サ変名詞 *		: 1.00000000
引用助詞	→	引用助詞 *		: 1.00000000
格助詞	→	格助詞 *		: 1.00000000
感動詞	→	感動詞 *		: 1.00000000
間投詞	→	間投詞 *		: 1.00000000
記号	→	記号 *		: 1.00000000
係助詞	→	係助詞 *		: 1.00000000
形容詞	→	形容詞 *		: 1.00000000
形容名詞	→	形容名詞 *		: 1.00000000
固有名詞	→	固有名詞 *		: 1.00000000
語尾	→	語尾 *		: 1.00000000
終助詞	→	終助詞 *		: 1.00000000
住所名	→	住所名 *		: 1.00000000
準体助詞	→	準体助詞 *		: 1.00000000
助動詞語幹	→	助動詞語幹 *		: 1.00000000
人名	→	人名 *		: 1.00000000
数詞	→	数詞 *		: 1.00000000
接続詞	→	接続詞 *		: 1.00000000
接続助詞	→	接続助詞 *		: 1.00000000
接頭辞	→	接頭辞 *		: 1.00000000
接尾辞	→	接尾辞 *		: 1.00000000
代名詞	→	代名詞 *		: 1.00000000
日時	→	日時 *		: 1.00000000
普通名詞	→	普通名詞 *		: 1.00000000
副詞	→	副詞 *		: 1.00000000
副助詞	→	副助詞 *		: 1.00000000
並立助詞	→	並立助詞 *		: 1.00000000
補助動詞語幹	→	補助動詞語幹 *		: 1.00000000
本動詞	→	本動詞 *		: 1.00000000
連体詞	→	連体詞 *		: 1.00000000
連体助詞	→	連体助詞 *		: 1.00000000
読点	→	読点 *		: 1.00000000
句点	→	句点 *		: 1.00000000