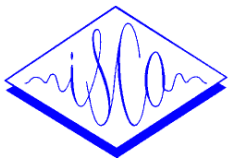

3rd International Conference on
Spoken Language Processing (ICSLP 94)
Yokohama, Japan
September 18-22, 1994

SIMILARITY-BASED IDENTIFICATION OF REPAIRS IN JAPANESE SPOKEN LANGUAGE

Gen-ichiro KIKUI* and Tsuyoshi MORIMOTO
ATR Interpreting Telecommunications Research Laboratories
2-2, Hikari-dai, Seika-cho, Soraku-gun, Kyoto, 619-02, Japan

Abstract

The paper presents an algorithm for locating parts of an utterance that is canceled by the speaker using repairs, or self-corrections. Especially, we concentrate on identification of the on-set of a canceled part. The underlying idea is that the repair and reparendum intervals are as similar to each other as are two coordinated phrases. We adopted the similarity-based algorithm for analyzing Japanese coordination structures proposed by Kurohashi and Nagao. The success rate of the algorithm is 92% for tagged utterances in ATR dialogue database.

1 Introduction

Currently there is much interest in spontaneous speech understanding systems. Such systems should be able to extract correct meaning from utterances including repairs, or self-corrections, because about 10 % of spontaneous utterances in Japanese contain repairs [4] like in English[5].

Following [5], a repair is divided into three parts: 1) the morpheme sequence to be repaired, called the *reparendum interval*, 2) the *disfluency interval* made up of silence and/or a cue word (e.g., "oops"), and 3) the *repair interval* that correspond to the utterance of the correcting material[1]. In the following example, the reparendum and the repair intervals are the word sequences marked by normal brackets and square brackets respectively [2].

至急	p[3]	(領収書の	コピー)
immediately		receipt-gen	copy
失礼	p	[登録書の	コピー] を ..
sorry		registration_form-gen	copy-acc

It is crucial to identify the positions of reparendum and disfluency intervals because most repairs are corrected simply by removing these intervals.

Recent studies have shown that the off-set (i.e., the end point) of a reparendum and the following difluency interval can be identified by using acoustic-prosodic cues (e.g., fricatives, stops) and/or morphological idiosyncrasy [5].

The remaining problem is how to identify the on-set (i.e., the starting point) of a reparendum. Previous studies are based on the observation that a part of the reparendum interval will be syntactically or semantically repeated in the repair interval.

In [1], repairs, including reparendum interval, are located by pattern matching the morpheme sequence of the input utterance. The patterns represent which morphemes are repeated in repair phenomena. The plausibility of the identified reparendum intervals are verified by syntactic and semantic analysis. One problem is that these patterns are not flexible enough to represent various sorts of repetition.

In [6], the algorithm first searches for a pair of similar words or phrases across (a candidate of) the off-set of a reparendum. It then creates a syntactic sub-tree that dominates both the preceding word of the pair and the off-set word. The reparendum interval corresponds to the leaves of the created sub-tree. The algorithm also relies on a full-sentence grammar to verify the plausibility of the reparendum interval. The algorithm achieved a high success rate, but no criterion is described for selecting one sub-tree when several sub-trees are possible.

This paper concentrates on locating the on-set of a reparendum. In particular, we focus on the algorithm for finding the most similar pair of word-sequences that is necessary for choosing the most plausible location out of possible on-set locations. The basic idea is that the repair and reparendum intervals are as similar to each other as are two coordinated phrases. We adopted the similarity-based analysis algorithm for Japanese coordination structure [2] to search for similar word-sequences on a morphologically analyzed input.

In the following, an outline of our entire algorithm is described in Section 2. Section 3 introduces the DP-based algorithm for analyzing coordination structures. Section 4 gives our algorithm in detail. Section 5 introduces experimental results and discussions.

2 Outline of Our Algorithm

First, we summarize the two characteristics of reparendum intervals which our algorithm relies on.

1. If we remove the reparendum interval of a repair together with the disfluency interval, the ill-formedness caused by the repair is resolved.
2. The reparendum and repair intervals are similar to each other in some sense (e.g., syntactic or semantic

*Currently at NTT Network Information Systems Laboratories.
[1]This definition is slightly different from that in [3].
[2]A literal translation of the example is "immediately, (a copy of the receipt), sorry , [a copy of the registration form] ..."
[3]'p' represents a pause.

or pragmatic sense).

Our algorithm consists of two separate search routines corresponding to the above two characteristics of reparendum intervals. Each routine is responsible for finding candidates of the reparendum interval based on the corresponding characteristic. Results of the two routines are finally merged.

2.1 Search with well-formedness Constraints

This part of the algorithm corresponds to the first characteristic of repairs. Among several aspects of well-formedness, our algorithm deals with only *syntactic* well-formedness (i.e., grammaticality).

The system first makes a set of all sub-sequences that ends with the given off-set. Each sub-sequences is then tested to determine whether it has the first characteristic or not. Sub-sequences that have the characteristic are chosen as candidates.

A given sub-sequence may by tested by parsing the corrected utterance (i.e., the utterance without the sub-sequence) with a reliable grammar. However, this is difficult because the well-formedness of the entire sentence is not guaranteed even if reparendum intervals are retracted and because developing a reliable global grammar is still an open-ended task. Therefore, our algorithm checks local grammaticality with an *adjacency matrix* that constrains what category of word can follow a given category Although checking well-formedness with the adjacency matrix is weaker than checking by full-syntactic analysis, it has been proved to be effective in a practical morphological analyzer for Japanese. The algorithm judges that the given sub-sequence satisfies the condition if the word immediately left of the sub-sequence can be adjacent to the starting word of the repair interval.

Note that there is no priority among the chosen candidates.

2.2 Search with Similarity Condition

This part of the algorithm corresponds to the second characteristic of repairs. It searches for the most similar pair of word-sequences brackets the given off-set. The similarity score of the pair should exceed a predetermined threshold. If such pair exists, then the prior part of the pair is the candidate of a reparendum, else the search process fails. Note that this routine outputs only one or zero candidates. To realize this routine, the similarity score between two sequences of words and its calculation algorithm should be defined. We adopted the similarity calculation developed for analyzing coordination structures in Japanese sentence. This routine is explained later.

2.3 Merging Results

System determines one estimated reparendum interval out of the result of the first routine in a simple way. Basically, the system chooses the reparendum candidate whose on-set is nearest to the on-set of the result of the second routine. If the ending word of the reparendum and the starting word of the repair interval are both postpositions that are not allowed to be adjacent [4], then the system selects the shortest reparendum candidate. If the second routine fails, the system also chooses the shortest candidate as a default.

3 Similarity-based Analyzer for Japanese Coordination

This section briefly introduces the accurate analysis algorithm for Japanese coordination structures proposed by Kurohashi and Nagao[2] that captures similarity of two coordinated expressions.

First, the input sentence is morphologically analyzed and segmented into a sequence of *bunsetsu*[5], denoted as $b_1, b_2, \ldots, b_n$ (or as $b_{1,n}$ in abbreviation).

Next, a coordination marker[6] is searched for. Let b_l be the bunsetsu containing the coordination marker.

Finally the analyzer tries to find the most similar pair of sequences bracketing the coordination marker. More formally, the final step searches for $k(1 \leq k \leq l)$ and $m(l+1 \leq m \leq n)$ that maximize the *similarity score* between two bunsetsu sequences, $b_{k,l}$ and $b_{l+1,m}$. This step is carried out in two sub-steps. First, for every m, k is determined that maximizes the similarity score between $b_{k,l}$ and $b_{l+1,m}$. Then, among the resulting combinations of k and m, the combination of the highest similarity score is selected.

The next sub-section explains the first half of the final step, namely, how to determine k that maximizes the similarity score between $b_{k,l}$ and $b_{l+1,m}$ for given l and m.

3.1 Finding Similar Pair of Sequences

The algorithm presupposes that the similarity score between two bunsetsu phrases is defined according to their syntactic and/or lexical properties. For example, if two bunsetsu share the same noun, the similarity score is high. Let A = (a(i,j)) be a matrix where a(i,j) contains the similarity score between b_i and b_j.

The analyzer searches for b_k and b_m on partial matrix $A' = (a(i,j)(1 \leq i \leq l, l+1 \leq j \leq m))$.

We consider a *path* from the bottom right element (a(l,m)) to an element in the leftmost column of the partial matrix A'.

$$path \quad ::= \quad (a(s_1,m), a(s_2,m-1), \ldots, a(s_{m-l}, l+1))$$
$$where\,, s_1 = l, s_{i+1} \leq s_i (1 \leq i \leq m-l+1).$$

A path represents which element in $b_{j+1,k}$ corresponds to which element in $b_{s_{k-j},j}$. For example, if a path con-

[4] In this case, the former postposition is the reparendum

[5] A bunsetsu is a kind of phrase used in Japanese school grammar. A bunsetsu consists of a content word and some functional words.

[6] A coordination conjunction or a verb in adverbial form.

tains $a(p,q)$ then b_p corresponds to b_q. Figure1 shows an example of a path. Note that several paths are possible between two particular elements on the matrix.

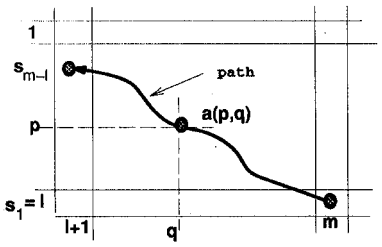

Figure 1: An example of a path

A path has a score that represents the 'goodness' of the correspondence between the two sequences. The similarity between two bunsetsu-sequences $b_{k,l}$ and $b_{l+1,m}$, is the highest score of all the possible paths connecting (l,m) to $(k,l+1)$.

The score of a path, for coordination analysis, is basically the sum of similarities on the path. If an element does not correspond to exactly one element on the path (i.e., $s_{i+1} = s_i$ or $s_{i+1} > s_i + 1$), a certain value, called the *penalty*, is subtracted from the score. The penalty is determined by the difference of the numbers of corresponding elements. Other rules for calculating the score are described in [2].

Using the above definition of similarity score, the left boundary (k) that maximizes the similarity score between prior sequence $(b_{k,l})$ and given posterior sequence $(b_{l+1,m})$ is the row number at the end of the path with the highest score (s_{m-l}).

The proposed algorithm efficiently locates the best path using the dynamic programming technique.

4 The Similarity-based Search for Repairs

This section describes how the algorithm in Section 3 is adopted to our task presented in Section 2.2.

4.1 Main Search Algorithm

First of all, the basic unit of an input utterance is changed from a bunsetsu phrase to a word. This is because reparendum and repair intervals often contain ill-formed sequences of words that do not form complete bunsetsu phrases, thus they should regarded as word sequences.

Secondly, the role of the coordination marker in the original algorithm is played by the off-set of the reparendum in our algorithm.

The third point is that the maximal length of the posterior sequence is bound to a fixed natural number u, whereas, in the original algorithm, the posterior sequence can reach to the end of the sentence.

The main part of the adopted algorithm is as follows:

1. Let w_l be the off-set word for the reparendum interval.
2. For every candidate of reparendum interval $(w_{l+1,m}$ where[3.] $l+1 \leq m \leq l+u)$, calculate the estimated reparendum interval and its score.

   The estimated reparendum interval is the word sequence that ends with the given off-set (w_l) and that is most similar to the candidate of repair interval. The word sequence most similar to a given sequence is located by the algorithm described below.
4. Return the estimated reparendum interval with the best score.

4.2 Similarity Calculation in Reparendum Identification

The second step of the algorithm searches for the sequence of words that is most similar to a given sequence of word. The search is carried out on the matrix introduced in Section 3, where each element contains a similarity score between two words, instead of the similarity score between bunsetsu phrases.

4.2.1 Similarity between two words

The similarity score between two words is heuristically defined depending on 1) the combination of their parts-of-speech (abbreviated as POS) and 2) the similarity of their lexical strings.

We classify combinations of two POS into the following 6 types.

type-1: The two POS are the same and are either postpositions or auxiliaries.

type-2: The two POS are the same. Both of them are functional word categories excluding postpositions and auxiliaries.

type-3: The two POS are the same and are content word categories.

type-4: The following combinations of parts-of-speech.

- a verb and a sahen-noun [7]
- a verb and a common-noun

type-5: The POS of the word in the reparendum candidate is unknown, and the POS in the corresponding repair is a content category.

type-6: All the combinations that are not covered by the above combinations above.

The table 1 defines the similarity score between two words for each type.

For every combination type except type-6, the score is positive when the lexical strings of the two words are exactly matched (the column of "Exact match"). For type-3, 4 and 5, scores proportional to the number characters shared by the two lexical strings are given unless they are not exactly matched (the column of "Partial match").

[7] A sahen-noun functions as a verb if it co-occurs with a special 'delexical verb' (e.g., "suru" or "dekiru").

Table 1: Similarity score between two words

comibination of parts-of-speech	Exact match	Partial match
type-1	2	0
type-2	1	0
type-3	10	2 * (the number of shared characters)
type-4	10	2 * (the number of shared Kanji characters)
type-5	0	2 * (the length of the common syllable sequence)
type-6	0	0

4.2.2 Similarity between word sequences

The similarity between two word sequences is calculated based on paths.

In the original algorithm, every path should end at the leftmost column of the matrix. This is because the whole posterior part is assumed to correspond to the whole prior part whose on-set is fixed to the bunsetsu phrase following the coordination marker.

However, in this case, the entire reparendum interval often corresponds to a part of the repair interval. Therefore, paths need not reach the leftmost column. The new definition of a path is given as follows.

$path \quad ::= \quad (a(s_1, m), a(s_2, m-1), \ldots, a(s_{m-o}, o))$
$where, s_1 = l, s_{i+1} \leq s_i \ (1 \leq i \leq m-o+1, l+1 \leq o \leq m).$

The score of a path is defined, again, in the same way as the original algorithm.

- The basic score is the sum of similarities on the path.
- If $s_i = s_{i-1}$ then $a(s_i, x)$ on the path is not included in the sum.
- If $s_i > s_{i-1} + 1$ then $2 * (s_i - s_{i-1} - 1)$ is subtracted from the score.
- If $o \neq l + 1$, then (o-l+1) is subtracted from the score.

The last item is newly introduced to prevent the algorithm from choosing similar but distant pairs of sequences.

4.3 Search for the best path

Although the definition of a path is altered, the algorithm to search for the best path remains the same because all paths possible with the new definition are equivalent to the intermediate data calculated during the original search process.

5 Experiment and Discussion

The method was tested using utterances from the ATR spoken language database. Utterances in the database are segmented and tagged. Moreover, reparendum intervals are manually marked with special brackets. In the experiment, we regarded these bracketed intervals as 'correct' reparenda.

From the database, we randomly chose 400 utterances that contained repairs. The first 100 utterances were used to adjust system parameters. The next 300 utterances were used to test the performance of the algorithm.

The success rate for the 300 utterances was 92%.

Most of the failures occurred when the beginning part of a reparendum was only semantically similar to the corresponding part of the repair interval. Repairs because the corresponding parts have rather direct semantic similarity will be correctly handled if we use a reliable thesaurus to assess word similarity. However, other type of repairs require deeper analysis where the semantic similarity is subtle.

The next major type of failures was due to the lack of precise syntactic and semantic constraints for well-formed structures. In particular, when the correct on-set of the reparendum is in a noun sequence, the first routine can not distinguish plausible nouns that might be the correct on-set of reparendum candidates; it selects all the nouns in the sequence as on-set candidates. However, some of nouns are not semantically plausible. This problem may be resolved if we incorporate 1) precise syntactic analysis as in [6] but just around the estimated reparendum interval and/or 2) adjacency analysis with preference metrics (e.g., with a part-of-speech bigram)

6 Concluding Remarks

We proposed an algorithm that identifies the on-set of a reparendum interval based on similarity calculations. The algorithm searches for the most similar pair of word-sequences across the given off-set of the reparendum interval. Experimental results for tagged utterances show that 92% of reparenda on-sets can be correctly identified.

References

[1] John Bear, John Dowding, and Elizabeth Shriberg. Integrating multiple knowledge sources for detection and correction of repairs in human-computer dialog. In *Proceedings of ACL-92*, 1992.

[2] Sadao Kurohashi and Makoto Nagao. Dynamic programming method for analyzing conjunctive structures in Japanese. In *Proceedings of Coling-92*, 1992.

[3] William Levelt. Monitoring and self-repair in speech. *Cognition*, 14, 1983.

[4] Jin'ichi Murakami and Shigeki Sagayama. A discussion of acoustic and linguistic problems in spontaneous speech recognition(in Japanese). In *Proc of SIG-NLC 91-57,*, 1991.

[5] Christine Nakatani and Julia Hirschberg. Speech-first model for repair detection and correction. In *Proceedings of ACL-93*, 1993.

[6] Yuji Sagawa, Noboru Ohnishi, and Noboru Sugie. On analysis of Japanese ill-formed sentences with self-repairs and their understanding by computer(in Japanese). *Journal of IPSJ*, 35(1), 1994.