A Method for Correcting Errors in Speech Recognition Using the Statistical Features of Character Co-occurrence

Satoshi Kaki, Eiichiro Sumita, and Hitoshi Iida
ATR Interpreting Telecommunications Research Labs,
Hikaridai 2-2 Seika-cho, Soraku-gun, Kyoto 619-0288, Japan
{skaki, sumita, iida}@itl.atr.co.jp

Abstract

It is important to correct the errors in the results of speech recognition to increase the performance of a speech translation system. This paper proposes a method for correcting errors using the statistical features of character co-occurrence, and evaluates the method.

The proposed method comprises two successive correcting processes. The first process uses pairs of strings: the first string is an erroneous substring of the utterance predicted by speech recognition, the second string is the corresponding section of the actual utterance. Errors are detected and corrected according to the database learned from erroneous-correct utterance pairs. The remaining errors are passed to the posterior process which uses a string in the corpus that is similar to the string including recognition errors.

The results of our evaluation show that the use of our proposed method as a post-processor for speech recognition is likely to make a significant contribution to the performance of speech translation systems.

1 Introduction

In spite of the increased performance of speech recognition systems, the output still contains many errors. For language processing such as a machine translation, it is extremely difficult to deal with such errors.

In integrating recognition and translation into a speech translation system, the development of the following processes is therefore important: (1) detection of errors in speech recognition results; (2) sorting of speech recognition results by means of error detection; (3) providing feedback to the recognition process and/or making the user speak again; (4) correct errors, etc.

For this purpose, a number of methods have been proposed. One method is to translate correct parts extracted from speech recognition results by using the semantic distance between words calculated with an example-based approach (Wakita *et al.*, 97). Another method also obtains reliably recognized partial segments of an utterance by cooperatively using both grammatical and n-gram based statistical language constraints, and uses a robust parsing technique to apply the grammatical constraints described by context-free grammar (Tsukada *et al.*, 97). However, these methods do not carry out any error correction on a recognition result, but only specify correct parts in it.

In this paper we therefore propose a method for correcting errors, which is characterized by learning the trend of errors and expressions, and by processing in an arbitrary length string.

Similar work on English was presented by (E.K. Ringger et al., 96). Using a noisy-channel model, they implemented a post-processor to correct word-level errors committed by a speech recognizer.

2 Method for Correcting Errors

We refer to two compositions of the proposal as Error-Pattern-Correction (EPC) and Similar-String-Correction (SSC) respectively. The correction using EPC and SSC together in this order is abbreviated to EPC+SSC.

2.1 Error-Pattern-Correction (EPC)

When examining errors in speech recognition, errors are found to occur in regular patterns rather than at random. EPC uses such error patterns for correction. We refer to this pattern as an Error-Pattern.

An Error-Pattern is made up of two strings. One is the

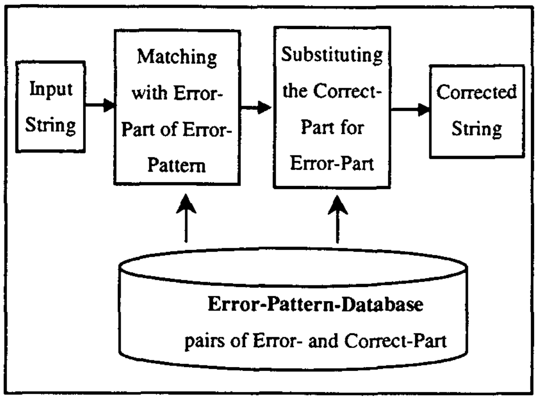
Input String
Matching with Error-Part of Error-Pattern
Substituting the Correct-Part for Error-Part
Corrected String
Error-Pattern-Database
pairs of Error- and Correct-Part

Figure 2-1 The block diagram for EPC

string including errors, and the other is the corresponding correct string (the former string is referred to as the Error-Part, and the latter as the Correct-Part respectively). These parts are extracted from the speech recognition results and the corresponding actual utterances, then they are stored in a database (referred to as an Error-Pattern-Database). In EPC, the correction is made by substituting a Correct-Part for an Error-Part when the Error-Part is detected in a recognition result (see Figure 2-1). Table 2-1 shows some Error-Pattern examples.

Table 2-1 Examples of Error-Patterns

Correct-Part	Error-Part
は何名様	はな名様
して頂きますので	しててきますので
失礼いたします	していたします
ご希望	ご気後
支払い方法	支払いを方法

2.1.1 Extraction of Error-Patterns

The Error-Pattern-Database is mechanically prepared using a pair of parts from the speech recognition results and the corresponding actual utterance. The examples below show candidates grouped according to the correct part '<何>' and the erroneous part '<な>'.

Error-Pattern Candidates	Frq.
<何>：<な>	3
は<何>：は<な>	3
数は<何>：数は<な>	3
数は<何>名様で：数は<な>名様で	2
数は<何>名様ぐ：数は<な>名様ぐ	1

EPC is a simple and effective method because it detects and corrects errors only by pattern-matching. The unrestricted use of Error-Patterns, however, may produce the wrong correction. Therefore a careful selection of Error-Patterns is necessary. In this method, several selection conditions are applied in order, as described below. Candidates passing all of the conditions are employed as Error-Patterns.

Condition of High Frequency: Candidates of not less than a given threshold value (2 in the experiment) in frequency are selected to collect errors which have a high frequency of occurrence in recognition results.

Condition of Non-Side Effect:, This step excludes the candidate whose Error-Part is included in actual utterances to prevent the Error-Part from matching with a section of actual utterances.

Condition of Inclusion-1: Because a long Error-Part is more accurate for matching, this step selects an Error-Pattern whose Error-Part is as long as possible. For two arbitrary candidates, when one of their Error-Parts includes the other, and their frequencies are the same value, the candidate whose Error-Part includes the other is accepted.

Condition of Inclusion-2: If some Error-Parts are derived from different utterances and have a common part in them, this common part is suitable for an Error-Pattern. Therefore in this step, an Error-Pattern with its Error-Part as short as possible is selected. For two arbitrary candidates, when one of their Error-Parts includes the other, and their frequencies have different values, the included candidate is accepted.

2.2 Similar-String-Correction (SSC)

In an erroneous Japanese sentence, the correct expressions can be estimated frequently by the row of characters before and after the erroneous sections of the sentence. This means that we are involuntarily applying a portion of a regular expression to an erroneous section.

Instead of this portion of the regular expression, SSC uses a collection of strings, the members of which are in the corpus (this collection we refer to as the String-Database). As shown in the block diagram in figure 2-2, the correction is performed through the following steps. the first step is error detection. The next step is the retrieval of the string that is most

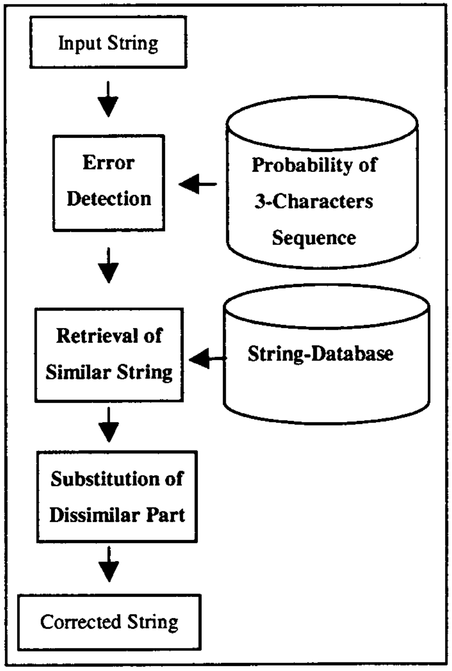
Input String
Error Detection
Probability of 3-Characters Sequence
Retrieval of Similar String
String-Database
Substitution of Dissimilar Part
Corrected String

Figure 2-2 The block diagram of SSC

similar to the string including errors from the String-Database (the former string is referred to as the Similar-String, and the latter as the Error-String). Finally, the correction is made using the difference between these two strings.

2.2.1 Procedure for Correction

The procedure for correction varies slightly, depending on the position of the detected error: a top, a middle, or a tail, in an utterance. Here we will explain the case of a middle.

Step 1: Estimate an erroneous section (referred to as an error-block) with error detection method[1]. If there is no error-block, the procedure is terminated.

Depending on the position of the error-block, the procedure branches in the following way.

If P1 is less than T (T=4), then go to the step for a top.

If a value L - P2 + T is less than T, then go to the step for a tail.

In all other cases, go to the step for a middle.

Here, P1 and P2 denote the start and end positions of an error-block, and L denotes the length of the input string.

Step 2: Take the string (Error-String) that comprises an error-block and each M (5 in the experiment) character before and after the error-block out of the input string, and using this string (Error-String) as a query key, retrieve a string (Similar-String) from the String-Database to satisfy the following condition. It must be located in a middle of an utterance, it must have the highest value (S), and S must be not less than a given threshold value (0.6 in the experiment). Here, S is defined as:

$$S = (L - N) / L$$

where L is the length of the Similar String, and N is the minimum number of character insertions, deletions, or substitutions necessary to transform the Error-String to the Similar-String.

If there is no Similar-String, then go to step 1 leaving this error-block undone.

Step 3: If the two strings (denoted A and B), that are each K (2 in the experiment) characters before and after an error-block in the Error-String, are found in the Similar-String, take out the string (denoted C) between A and B in the Similar-String. If it is not found, then go to Step 1 leaving this error-block undone.

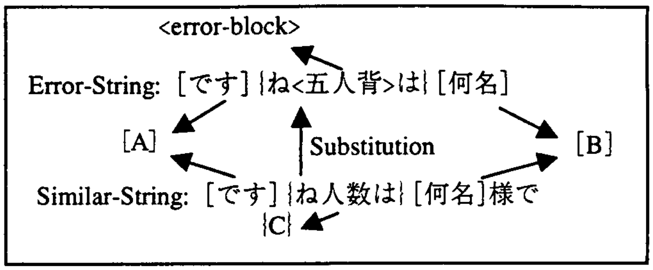
<error-block>
Error-String: [です] ¦ね<五人背>は¦ [何名]
[A]
Substitution
[B]
Similar-String: [です] ¦ね人数は¦ [何名]様で
¦C¦

Figure 2-3 The procedure of SSC

Substitute string C as the correct string for the string between A and B in the Error-String (see figure 2-3).

3. Evaluation

3.1 Data Condition for Experiments

Results of Speech Recognition: We used 4806 recognition results including errors, from the output of speech recognition (Masataki *et al.*, 96; Shimizu *et al.*, 96) experiment using an ATR spoken language database (Morimoto *et al.*, 94) on travel arrangements. The characteristics of those results are shown in table 3-1.

The breakdown of these 4806 results is as follows: 4321 results were used for the preparation of Error-Patterns and the other 495 results were used for the evaluation.

Table 3-1 The recognition characteristics

Recognition accuracy(%) (in character)	Insertion	Deletion	Substitution	Sum
74.73	2642	1702	8087	12431

Preparation of Error-Patterns: As the threshold value for the frequency of the occurrence, we employed a value of not less than 2, therefore we obtained 629 Error-Patterns using the 4321 results of speech recognition.

Preparation of the String-Database: Using the different data-sets of the ATR spoken language database from the above-mentioned 4806 results, we prepared the String-Database.

We employed 3 as the threshold value for the frequency of the occurrence, and 10 as the length of a string, therefore obtaining 16655 strings.

3.2 Two Factors for Evaluation

We evaluated the following two factors before and after correction: (1) the counting of errors, and (2) the effectiveness of the method in understanding the recognized results.

[1] For detecting errors in Japanese sentences, the method using the probability of character sequence was reported to be fairly effective (Araki *et al.*, 93). The result of a preliminary experiment was that the precision and recall rates were over 80% and over 70% respectively.

To confirm the effectiveness, the recognition results were evaluated by two native Japanese. They assigned one of five levels, A-E, to each recognition result before and after correction, by comparing it with the corresponding actual utterance. Finally, we employed the overall results of the stricter of two evaluators.

(A) No lacking in the meaning of the actual utterance, and with perfect expression.
(B) No lacking in meaning, but with slightly awkward expression.
(C) Slightly lacking in meaning.
(D) Considerably lacking in meaning.
(E) Unable to understand, and unable to imagine the actual utterance.

4. Results and Discussions

4.1 Decrease in the Number of Errors

Table 4-1 shows the number of errors before and after correction. These results show the following.

Table 4-1 The number of errors before and after correction

	Insertion	Deletion	Substitution	Sum
Before	264	206	891	1361
EPC	226(-14.4)	190(-7.8)	853(-4.3)	1269(-6.8)
SSC	251(-4.9)	214(+3.9)	870(-2.4)	1335(-1.9)
EPC+SSC	216(-18.2)	198(-3.9)	831(-7.9)	1245(-8.5)

The values inside brackets () are the rate of decrease

In EPC+SSC, the rate of decrease was 8.5%, and the decrease was obtained in all type of errors.

In SSC, the number of deletion errors increased by 3.9%. The reason for this is that in SSC, correction by deleting the part of a substitution error frequently caused new deletion errors as shown in the example below. From the standpoint of the correction it might be a mistaken correction, but it increases understanding of the results by deleting a noise and makes the results viable for machine translation. It therefore practically refines the speech recognition results.

Correct String:
'はいありがとうございます京都観光ホテル**予約係で**ございます'
"*Hai arigatou gozaimasu Kyoto Kanko Hoteru yoyaku gakari de gozaimasu*", (Thank you for calling Kyoto Kanko Hotel reservations.)
Input String:
'あはいありがとうございますえ京都観光ホテル**や日間**ございます'
"*A hai arigatou gozaimasu e Kyoto Kanko Hoteru yanichikan gozaimasu*", (Thank you for calling Kyoto Kanko Hotel ....)
Corrected String:
'あはいありがとうございますえ京都観光ホテルでございます'
"*A hai arigatou gozaimasu e Kyoto Kanko Hoteru de gozaimasu*", (Thank you for calling Kyoto Kanko Hotel.)

4.2 Improvement of Understandability

Table 4-2 shows the number of change in the evaluated level.

The rate of improvement after correction was 7%. There were also a lot of cases that improved their level by recovering content words. For example, the word "cash" was recovered in '返金で→現金で' (before→after), "guide" in '五内→ご案内', etc.

These results confirm that our method is effective in improving the understanding of the recognition results.

On the other hand, there were four level-down cases. Three of these cases were caused by the misdetection of errors in the SSC procedure. The remaining case occurred in the EPC procedure. The Error-Pattern used in this case could not be excluded by the condition of non-side effects because its Error-Part was not included in the corpus of the actual utterance.

Table 4-2 The number of changes in the evaluated level before and after correction.

	EPC	SSC	EPC+SSC
Improve	18(3.7)	15(3.1)	34(7.0)
No Change	466(96.1)	467(96.3)	447(92.2)
Down	1(0.2)	3(0.6)	4(0.8)

The values inside brackets () are the rate (%) of the number to total number of evaluated results.

4.3 More Applicable for a Result Having a Few Errors

Table 4-3 shows the rate of change in the evaluated level by the original number of erroneous characters[2]

Table 4-3 The rate of change in the evaluated level by the original number of erroneous characters involved in the recognition results (EPC+SSC).

Num. of erroneous characters	Num. of results	Rate(%) of change		
		Improve	No Change	Down
0	102	0.0	98.0	2.0
1	30	16.7	80.0	3.3
2	21	28.6	66.7	4.8
3	26	19.2	80.8	0.0
4	40	12.5	87.5	0.0
5	27	14.8	85.2	0.0
6	24	12.5	87.5	0.0
7	21	9.5	90.5	0.0
8	17	0.0	100.0	0.0
9	20	5.0	95.0	0.0
10	29	0.0	100.0	0.0
11	22	0.0	100.0	0.0
12≧	106	2.8	97.2	0.0
Total	485	7.0	92.2	0.8

[2] This number is the minimum number of character insertions, deletions or substitutions necessary to transform the result of recognition into a corresponding actual utterance.

included in the recognition results.

The recognition results improving their level after correction mostly fell in the range of erroneous numbers by not more than 7. The reasons for this are that with there being many errors, the failure of the corrections increases because the corrections are prevented by other surrounding errors. In addition, when only a few successful corrections have been made, they have little influence on the overall understanding.

These results show that the proposed method is more applicable for a recognition result having a few errors, as compared with one having many errors.

5 Conclusion

As described above, our proposed method has the following features:

(1) Since the proposed method is designed with a arbitrary length string as a unit, it is capable of correcting errors which are hard to deal with by methods designed to treat words as units.

For example, the insertion error 'を' ("*wo*") in the string '支払いを方法' ("*shiharai wo houhou*") shown in table 2-1 cannot be corrected by a method designed to treat words as units, because of the existence of the particle 'を' ("*wo*") as a correct word. However with the proposed method, it is possible to correct this kind of error by using the row of characters before and after 'を' ("*wo*").

(2) In the proposed method of learning the trend of errors and expressions with long strings, it is possible to correct errors where it is difficult to narrow the candidates down to the correct character with the probability of the character sequence alone.

When considering the candidate for 'て' ("*te*") in 'してできますので' ("*shitetekimasunode*") shown in table 2-1 to satisfy the probability of the character sequence, its candidates, 'い' ("*i*"), 'お' ("*o*"), '頂' ("*itada*") are arranged in order of increasing probability. It is therefore difficult to narrow the candidates into the correct character '頂' ("*itada*") by the probability of character sequence alone. But with the proposed method it is possible to correct this kind of error by using the row of the characters before and after 'て' ("*te*").

(3) Both the Error-Pattern-Database and String-Database can be mechanically prepared, which reduces the effort required to prepare the databases and makes it possible to apply this method to a new recognition system in a short time.

From the evaluation, it became clear that the proposed method has the following effects:

(1) It reduces over 8% of the errors.
(2) It improves the understanding of the recognition results by 7%.
(3) It has very little influence on correct recognition results.
(4) It is more applicable for a recognition result with a few errors than one with many errors.

Judging from these results and features, the use of the proposed method as a post-processor for speech recognition is likely to make a significant contribution to the performance of speech translation systems.

In the future, we will try to improve the correcting accuracy by changing algorithms and will also try to improve translation performance by combining our method with Wakita's method.

References

T. Araki et al., 93. *A Method for Detecting and Correcting of Characters Wrongly Substituted, Deleted or Inserted in Japanese Strings Using 2nd-Order Markov Model.* IPSJ, Report of SIG-NL, 97-5, pp. 29-35 (1993)

T. Morimoto et al., 94: *A Speech and language database for speech translation research.* Proc. of ICSLP '94, pp. 1791-1794, 1994.

H. Masataki et al., 96. *Variable-order n-gram generation by word-class splitting and consecutive word grouping.* In Proc. of ICASSP, 1996.

T. Shimizu et al., 96. *Spontaneous Dialogue Speech Recognition using Cross-word Context Constrained Word Graphs.* ICASSP '96, pp. 145-148, 1996.

Y. Wakita et al., 97. *Correct parts extraction from speech recognition results using semantic distance calculation, and its application to speech translation.* ACL/EACL Workshop Spoken Language Translation, pp. 24-31, 1997-7.

H. Tsukada et al., 97. *Integration of grammar and statistical language constraints for partial word-sequence recognition.* In Proc. of 5th European Conference on Speech Communication and Technology (EuroSpeech '97), 1997.

E.K.Ringger et al., 96. *A Fertility Channel Model for Post-Correction of Continuous Speech Recognition.* ICSLP'96, pp. 897-900, 1996.