ERROR CORRECTION OF SPEECH RECOGNITION OUTPUTS USING GENERALIZED LR PARSING AND CONFUSION MATRIX

Tatsuya Iwasa and Kenji Kita

Faculty of Engineering
Tokushima University
Minami-josanjima, Tokushima 770, JAPAN

e-mail. {iwasa and kita}@is.tokushima-u.ac.jp

Abstract

In this paper, we describe a method for correcting misrecognition results derived from a speech recognizer. Our method is based on generalized LR parsing and uses a confusion matrix in order to select the best sentence out of multiple candidates. The proposed method was applied to the actual speech recognition system developed in our laboratory.

1 Introduction

Recently, remarkable progress has been made in acoustic modeling research, with hidden Markov models (HMMs) and a neural network-based approach. However, speech recognition based on acoustic information alone does not result in good performance for large vocabulary tasks. Successful speech recognition/understanding requires the use of linguistic information. There are two approaches using linguistic information:

(1) To directly use linguistic information at recognition time.

(2) To correct recognition outputs using linguistic information.

This paper is concerned with the second approach. In this paper, we describe a method for correcting misrecognized utterances using lexical and syntactic information. More precisely, we

describe an error-correcting generalized LR parser using candidate scoring based on a confusion matrix. The use of the confusion matrix provides a good cost calculation scheme and contributes to select the most likely sentence out of multiple candidates. Saito et al. [1] already proposed an error-correcting method using generalized LR parsing and a confusion matrix. Their method, however, assumes that consecutive errors do not occur in the misrecognized utterances. Our method is based on error-correcting LR parsing introduced in [2], which can handle consecutive errors, and is enhanced with scoring by the confusion matrix.

2 Speech Recognition System: An Overview

This section describes a speech recognition system developed in our laboratory.

2.1 Acoustic Models

As acoustic models, we adopt *hidden Markov models* (HMMs for short) [3], which have been successfully used in recent state-of-the-art speech recognition systems. HMMs are stochastic models suitable for handling the uncertainty that arises in speech, such as contextual effects and speaker variabilities.

In our speech recognition system, Japanese syllables are used as the basic HMM unit because the whole word-based approch is difficult to meet the real-time requirements in case of the large vocabulary size. There are about 100 phonetically different spoken syllables in all. Each syllable is represented by a continuous mixture HMM, in which an output probability density function is characterized by a 39-component diagonal covariance Gaussian mixture. See Table 1 for speech analysis conditions.

2.2 Recognition Method

As stated above, our system uses hidden Markov models of syllables as the basis for speech modeling. Word models are built by concatenating syllable models. The speech recognition module performs a time-synchronous Viterbi beam search, matching syllable models against the speech input. That is, it maintains a beam of the best scoring candidates and extends these one frame at a time. Recognition candidates with a low likelihood score are pruned.

Table 1: Speech analysis conditions

Sampling frequency and precision	16 kHz, 16 bit
Pre-emphasis	$1 - 0.97z^{-1}$
Hamming window	25 ms
Frame period	10 ms
Acoustic parameters	12 MFCC (mel-frequency cepstral coefficients) + 12 Δ MFCC + 12 $\Delta\Delta$ MFCC + power + Δ power + $\Delta\Delta$ power (39 dimensions in all)

All candidates cover the utterance from the beginning to the most recently processed frame. As a recognition path reaches the end of a syllable model, the search transits to the beginning of syllable models that can follow the current syllable which ends the path. Currently, our system uses no restrictions concerning syllable connections (i.e. any syllable can follow any other syllables). In the speaker-dependent condition, the syllable recognition rate is around 90%.

2.3 Examples of Recognition Outputs

Since our speech recognition system does not use any syntactic or semantic knowledge, it sometimes produces a syllable sequence which include misrecognized syllables. Table 2 shows some examples of recognition outputs.

Table 2: Speech recognition examples

	bf Real sequence	bf Recogniged sequence
1	ha chi ga tsu yo Q ka no yo ru ka ra de su	ha chi ga tsu yo Q ka no yo [ro] ka ra de su
2	i tsu ka ra o to ma ri ni na ri ma su ka	i tsu ka ra [ ] to ma ri ni na ri ma su ka
3	wa ka ri ma shi ta	wa ka ri [i] ma shi ta

There are three types of errors as follows:

1. Substitution error

   A syllable is incorrectly recognized as another syllable. For example, the tenth syllable /ru/ in example (1) is recognized as /ro/.

2. Deletion error

   A syllable which is actually spoken is not recognized. For example, the fifth syllable /o/ in example (2) is a deleted syllable.

3. Insertion error

   A syllable which is not actually spoken is recognized. For example, the fourth syllable /i/ in example (3) is an inserted syllable.

3 Error-Correcting Method: An Example

S	→	NP	V	
S	→	V		
NP	→	N		
NP	→	N	V	
N	→	ko	re	
P	→	o		
V	→	ku	re	
V	→	o	ku	re

Figure 1: Example of Grammar.

	action					goto				
	o	ko	ku	re	$	S	N	V	P	NP
0	sh 5	sh 1	sh 4			7	2	6		3
1				re 8						
2	sh 9, re 3		re 3						10	
3	sh 5		sh 4					11		
4				sh 12						
5			sh 13							
6					re 2					
7					acc					
8	re 5		re 5							
9	re 6		re 6							
10	re 4		re 4							
11					re 1					
12					re 7					
13				sh 14						
14					re 8					

Figure 2: LR parsing table for Figure 1

In this section, we present a sample trace of the error-correcting parser. An error-correcting method is essentially based on [2]. Nodes indicated by ●, ◎ and ○ are state vertexes. State vertex ◎ shows a Vca(current active vertex). Parsing actions are performed against all the current vertexes. State vertex ○ shows a Vna(next active vertex), that is, it will become an active vertex in the next step. Our error correcting parser has four kinds of parses. The first one is a deletion-correcting parse, whose results are included in Vca. The second one is a matching-parse, whose results are included in Vna. The third one is a substitution-correcting parse, whose results are included in Vna. The fourth one is an insertion-correcting parse, whose results are included in Vna.

In our example below, syllable sequence "o re ku re" will be corrected using the grammar in Figure 1 and the LR parsing table in Figure 2. For simplicity, the following assumptions are made:

(1) Syllable /a/ may be substituted by /o/.

(2) Syllable /o/ may be deleted.

(3) Syllable /re/ may be inserted.

(4) Two cosecutive syllables cannot be deleted.

Initially, state 0 is created.

Parsing the first syllable /o/

The assumption (2) creates a deletion-correcting parse in Figure 3. The parser looks for deleted syllable candidates before the first syllable /o/ in "o re ku re". By referring to the LR table, at state 0, /o/ expects the action "shift 5". The parser creates a new grammar vertex labelled by "o" containing "1, del". Here, "1, del" means that we correct the first syllable as a deleted syllable, and creates a new state vertex 5 which is included in Vca.

For the matching-parse in Figure 4, the parser considers all the state vertexes which is included in Vca. At state vertex 0, by referring to the LR table, "shift 5" is indicated. The parser creates a new grammar vertex labelled by "o" and a new state vertex 5 which is included in Vna. At state vertex 5, no action is attached to syllable /o/.

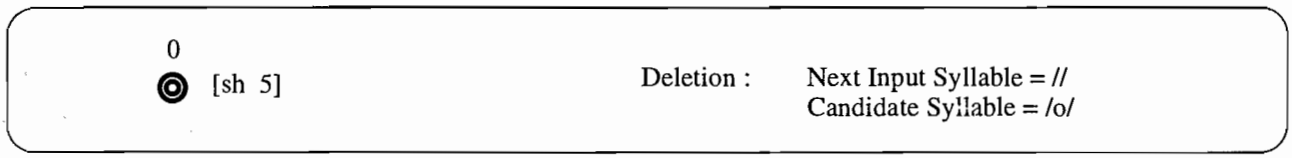

Figure 3: Trace of Error Correcting (1)

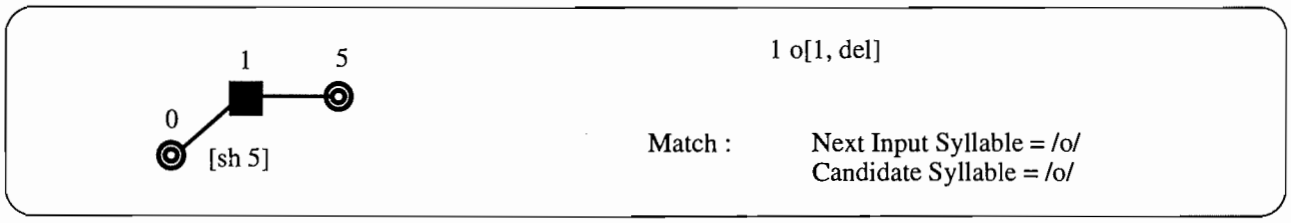

Figure 4: Trace of Error Correcting (2)

The assumption (1) creates the substitution-correcting parse in Figure 5, At state vertex 0, by referring to the LR table, "shift 1" is indicated. The parser creates a new grammar vertex labelled by "a" and a new state vertex 1 which is included in Vna. At sate vertex 5, no action is attached to syllable /a/.

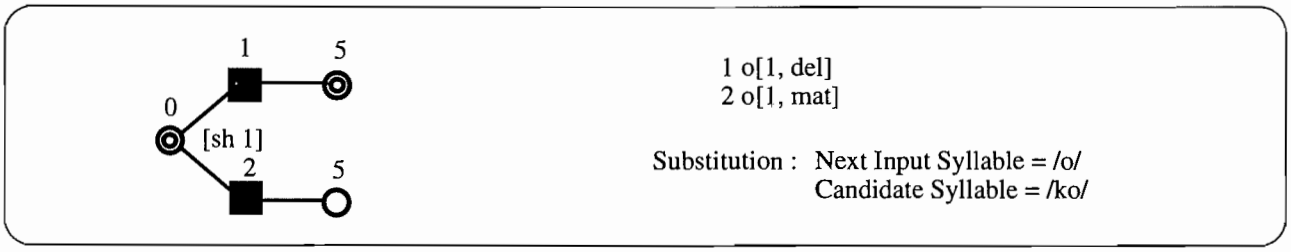

Figure 5: Trace of Error Correcting (3)

Our assumption tells that /o/ is not inserted.

Parsing the second syllable /re/

The matching-parse in Figure 6 has a current active vertex 1, which expects "shift 8" by syllable /re/. The parser creates a new grammar vertex labelled by "re" and a new state vertex which is included in Vna.

The assumption (3) tells that /re/ is possibly inserted. Thus, we obtain the insertion-correcting parse in Figure 7. The parser creates a new grammar vertex from each Vca and a new state vertex which is included in Vna.

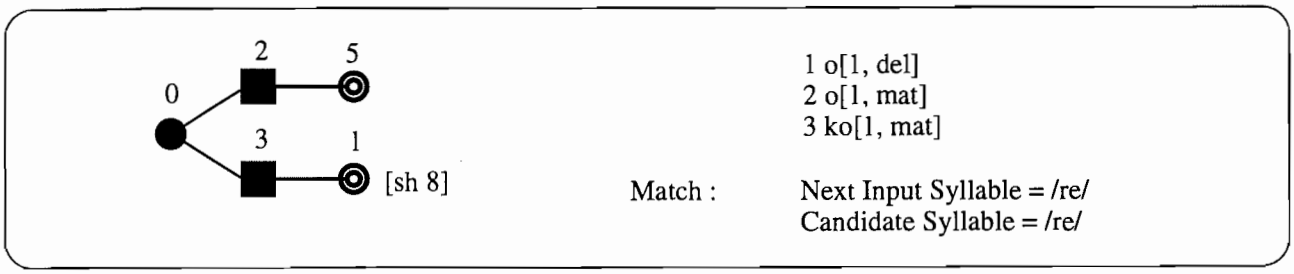

Figure 6: Trace of Error Correcting (4)

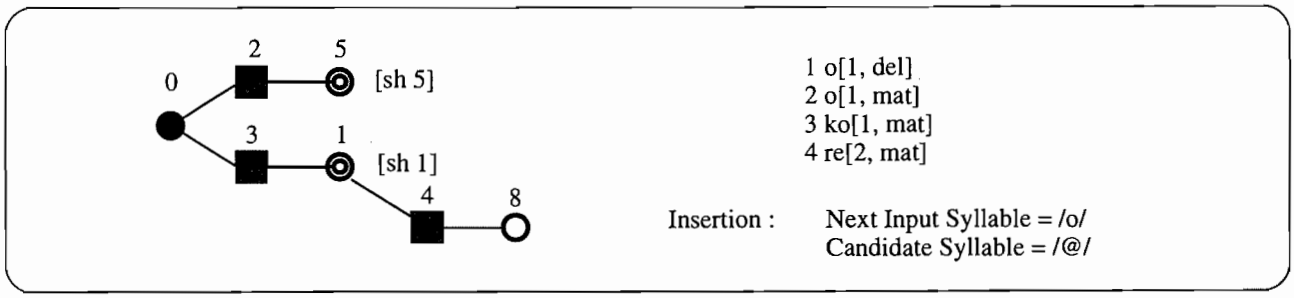

Figure 7: Trace of Error Correcting (5)

Parsing the third syllable /o/

The assumption (2) tells that /o/ is possibly deleted. Thus, we obtain the deletion-correcting parse in Figure ??. By referring to the LR table, state 8 has action "reduce 5". A reduce action will be performed in the same manner as in generalized LR parsing. The parser creates a new grammar vertex labelled by "N" containg "3 4". The parser creates a new state vertex 2.

Parsing the end-symbol /$/

State vertex 14 expects "reduce 8" in Figure 8. In reduce action, if the path includes the insertion-symbol("@"), our algorithm executes one more pop.

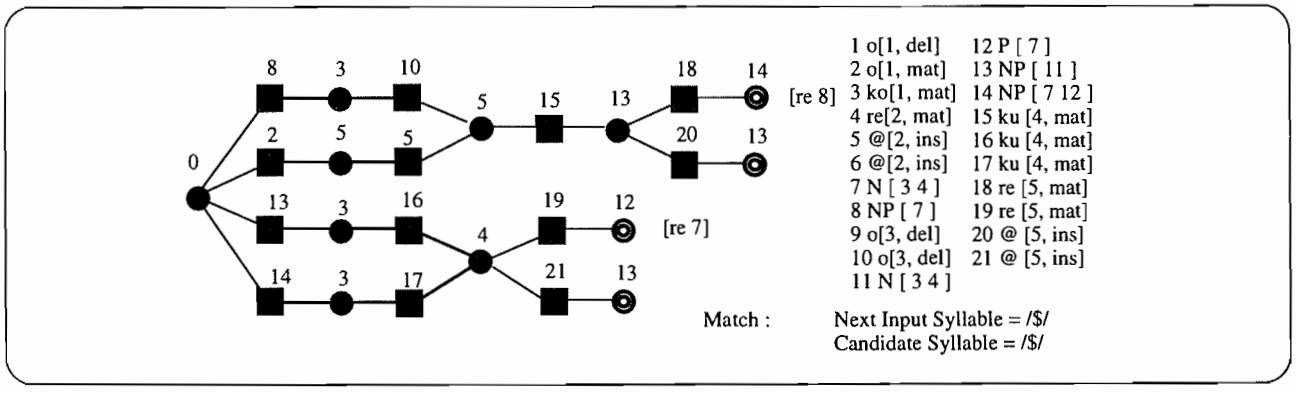

Figure 8: Trace of Error Correcting (6)

After this action, the stack is as in Figure 9.

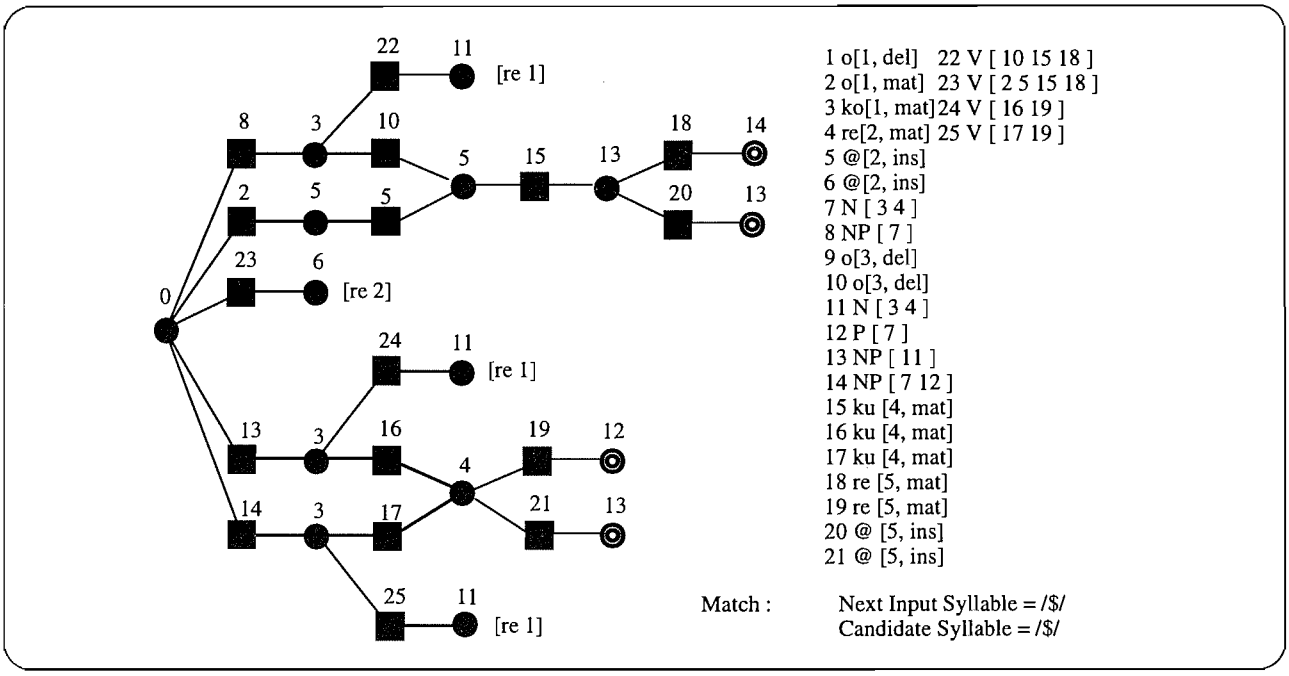

Figure 9: Trace of Error Correcting (7)

4 Confusion Matrix

A confusion matrix shows the probability of recognition (syllable by syllable). Figure 10 shows an example of a confision matrix. The first line shows that if the syllable /a/ is spoken, then it is recognized correctly 69.5% of the time; misrecognizes it as /i/ 2.4% of the time, as /ka/ 2.4% of the time, and so on. The last column shows the deletion probability. Therefore the probability that syllable /a/ was a deleted syllable is 8.5%. The last line shows insertion probability. Therefore the probability that syllable /a/ was a inserted syllable is 3.8%.

4.1 Cost Calculation Using Confusion Matrix

We calcurate the cost for the incorrect syllables as following,

1. Deleted syllables

   According to a confision matrix, the syllable /o/ is deleted more frequently than any other syllables. Thus, we parse the syllable /o/ with low cost than any other syllables as deleted syllable.

	a	i	u	e	o	ka	...	@
a	69.5	2.4	0.0	0.0	0.0	2.4		8.5
i	0.2	86.6	0.2	0.7	0.2	0.0		4.5
u	0.0	0.9	82.1	0.9	0.0	0.0		7.1
e	0.0	1.9	0.0	84.0	0.0	0.0		2.8
o	0.0	0.0	0.0	0.0	69.3	0.0		11.6
ka	0.0	0.0	0.0	0.0	0.0	97.7		0.3
⋮								⋮
@	3.8	26.5	3.8	1.7	3.8	0.4	...	

Figure 10: An Example of Confusion Matrix

2. Substituted syllables

   According to the confusion matrix, not all syllables can be substituted to any other syllable. If the syllable /a/ is generated from the device, there is a slight chance that original input was /i/ or /ka/,respectively, but no chance that the original input was /u/,/e/ or /o/. Thus,we pase the substituted syllable /i/ and /ka/ with lower cost than any other syllables.

3. Inserted syllables

   According to the confision matrix, the probability of the syllable /i/ being extra is higer than the probability of any other syllables. Thus, we paese the syllable /i/ with lower cost than any other syllables as inserted syllable.

4.2 An Error-Correcting Example

We include an error-correcting example in Table 3. The correct sentence is "ka i gi ni sa N ka shi ta i no de su ga", which means "I would like to register for the conference" in English. This sentence was misrecognized as "i Q ka i gi ni sa N ka shi ta i no de su ga", but successfully corrected.

5 Conclusion

This paper described a method for correcting misrecognition results derived from a speech recognizer. Our method is based on generalized LR parsing and uses a confusion matrix in order to select the best sentence out of multiple candidates.

Table 3: An Error-Correcting Example

sentence		cost
correct	ka i gi ni sa N ka shi ta i no de su ga	
misrecognized	i Q ka i gi ni sa N ka shi ta i no de su ga	
1	@ @ ka i gi ni sa N ka shi ta i no de su ga	0.24
2	@ @ ka i gi ni sa N ka o shi ta i no de su ga	0.29
3	o o i @ ka i gi ni sa N ka shi ta i no de su ga	0.33
4	@ @ ka i gi ni sa N ka o shi Q ta @ no de su ga	0.36
5	o o i @ ka i gi ni sa N ka o shi ta i no de su ga	0.38
6	@ @ ka i gi ni o o sa @ ka o shi ta i no de su ga	0.40
7	@ @ ka i gi ni sa N ka o shi ta o o o i no de su ga	0.42
8	o o i @ ka i gi ni sa N ka o shi Q ta @ no de su ga	0.45
9	@ @ ka i gi ni o o sa @ ka o shi Q ta @ no de su ga	0.47
10	@ @ ka i gi ni sa N ka o a shi ta o o i no de su ga	0.49

References

[1] H. Saito and M. Tomita: "GLR Parsing for Noisy Input", *Generalized LR Parsing*, Kluwer Academic Publishers (1991).

[2] T. Nagao and E. Tanaka: "An Error-correcting Algorithm for Context-free Languages Based on a Generalized LR Parser", IEICE Technical Report, COMP94-66 (1994).

[3] X. D. Huang, Y. Ariki and M. A. Jack, *Hidden Markov Models for Speech Recognition*, Edinburgh University Press (1990).

[4] A. V. Aho, R. Sethi and J. D. Ullman: "Compilers, Principles, Techniques, and Tools", Addison-Wesley (1986).

[5] M. Tomita (Ed.): *Generalized LR Parsing*, Kluwer Academic Publishers (1991).