HMM CONTINUOUS SPEECH RECOGNITION USING PREDICTIVE LR PARSING

Kenji KITA, Takeshi KAWABATA, Hiroaki SAITO

ATR Interpreting Telephony Research Laboratories
Seika-chou, Souraku-gun, Kyoto 619-02, JAPAN

ABSTRACT

This paper proposes a new continuous speech recognition method using an efficient parsing mechanism, an LR parser, driving HMM modules directly without any intervening structures such as a phoneme lattice. Accurate and efficient speech parsing is achieved by combining HMM and LR parsing. This method is tested in Japanese phrase recognition experiments. Two grammars are prepared, a general Japanese grammar and a task-specific grammar. The phrase recognition rate with the general grammar is 72% for top candidates and 95% for the five best candidates. With the task-specific grammar, recognition rate is 80% and 99%, respectively.

1 INTRODUCTION

There have been many speech recognition systems which use syntactic information to improve recognition accuracy. For example, statistical language modeling such as a bigram or a trigram [1, 2, 3], finite state grammars [4, 5] and context-free grammars [6, 7].

This paper proposes a new method for parsing speech data directly without any intervening structure such as a phoneme lattice. This method uses generalized LR parsing [8] and HMM phone verifiers. Generalized LR parsing is a kind of LR parsing [9], originally developed for programming languages and has been extended to handle arbitrary context-free grammars. An LR parser is guided by an LR parsing table automatically created from context-free grammar rules, and proceeds left-to-right without backtracking. Compared with other parsing algorithms such as the CYK(Cocke-Younger-Kasami) algorithm [10] or Earley's algorithm [11], a generalized LR parsing algorithm is the most efficient algorithm for natural language grammars.

There has been some applications of generalized LR parsing to speech recognition. Tomita [12] proposes a word lattice parsing algorithm. Saito [13] proposes a method of parsing phoneme sequences that include altered, missing and/or extra phonemes. However, these methods were inadequate because of the information loss due to signal-symbol conversion. Our approach does not use any intervening structures. The system drives HMM phone verifiers (HMM phone models) directly for detecting/verifying phones predicted using an LR parsing table.

The HMM has the ability to cope with the acoustical variation of speech by means of stochastic modeling. Moreover, any word models can be composed of phone models, so it is easy to construct a large vocabulary speech recognition system.

This paper is organized as follows. Section 2 describes the LR parsing mechanism. Section 3 describes our new approach, called HMM-LR. Section 4 describes recognition experiments using HMM-LR. Section 5 presents our conclusions.

2 LR PARSING

2.1 LR Parsing

LR parsing was originally developed for programming languages. It is applicable for a large class of context-free grammars.

The LR parser is deterministically guided by an LR parsing table with two subtables (action table and goto table). The action table determines the next parser action $ACTION[s, a]$ from the state s currently on top of the stack and the current input symbol a. There are four kinds of actions, *shift*, *reduce*, *accept* and *error*. *Shift* means shift one word from input buffer onto the stack, *reduce* means reduce constituents on the stack using the grammar rule, *accept* means input is accepted by the grammar, and *error* means input is not accepted by the grammar. The goto table determines the next parser state $GOTO[s, A]$ from the state s and the grammar symbol A.

The LR parsing algorithm is summarized below.

1. *Initialization.* Set p to point to the first symbol of the input. Push the initial state 0 on top of the stack.
2. Consult $ACTION[s, a]$ where s is the state on top of the stack and a is the symbol pointed to by p.
3. If $ACTION[s, a]$="*shift* s'", push s' on top of the stack and advance p to the next input symbol.
4. If $ACTION[s, a]$="*reduce* $A \rightarrow \beta$", pop $|\beta|$ symbols off the stack and push $GOTO[s', A]$ where s' is the state now on top of the stack.

5. If $ACTION[s,a]$="*accept*", parsing is completed.

6. If $ACTION[s,a]$="*error*", parsing fails.

7. Return to 2.

2.2 Generalized LR Parsing

Standard LR parsing cannot handle ambiguous grammars. For an ambiguous grammar, the LR parsing table will have *multiple entries* (conflicts). As a general method, a stack-splitting mechanism can be used to cope with multiple entries. Whenever a multiple entry is encountered, the stack is divided into two stacks, and each stack is processed in parallel. Thus, it is possible to use LR parsing to handle an ambiguous grammar which describes natural language.

A simple example grammar is shown in Table 1, and the LR parsing table, compiled from the grammar automatically, is shown in Table 2. The left part is the action table and the right part is the goto table. The entry "acc" stands for the action "accept", and blank spaces represent "error". The terminal symbol "$" represents the end of the input.

Table 1: Example grammar

1.	S	→	NP V
2.	S	→	V
3.	NP	→	N
4.	NP	→	N P
5.	N	→	k o r e
6.	P	→	o
7.	V	→	k u r e
8.	V	→	o k u r e

Table 2: LR parsing table

	e	o	u	k	r	$	S	N	V	P	NP
0		s5		s2			6	1	3		4
1		s7,r3		r3						8	
2		s9	s10								
3						r2					
4		s5		s11					12		
5				s13							
6						acc					
7				r6							
8				r4							
9					s14						
10					s15						
11			s10								
12						r1					
13			s16								
14	s17										
15	s18										
16					s19						
17		r5		r5							
18						r7					
19	s20										
20						r8					

3 HMM-LR METHOD

3.1 Basic Mechanism

In standard LR parsing, the next parser action (shift, reduce, accept or error) is determined using the current parser state and next input symbol to check the LR parsing table. This parsing mechanism is valid only for symbolic data and cannot be applied simply to continuous data such as speech.

In our approach, the LR parsing table is used to predict the next phone in the speech. For the phone prediction, the grammar terminal symbols are phones instead of the grammatical category names generally used in natural language processing. Consequently, a lexicon for the specified task is embedded in the grammar.

The following describes the system operation. First, the parser picks up all phones which the initial state of the LR table predicts, and invokes the HMM phone models to verify the existence of these predicted phones. During this process, all possible parsing trees are constructed in parallel. The phone verifier (HMM phone model) receives a probability array which includes end point candidates and their probabilities, and updates it using an HMM phone probability calculation process (Viterbi or trellis algorithm). This probability array is attached to each partial parsing tree. When the highest probability in the array is lower than a threshold level, the partial parsing tree is pruned by threshold level, and also by beam-search technique. The parsing process proceeds in this way, and stops if the parser detects an accept action in the LR parsing table. In this case, if the best probability point reaches the end of speech data, parsing ends successfully.

A very accurate, efficient parsing method is achieved through the integrated processes of speech recognition and language analysis.

3.2 Algorithm

To describe an algorithm for HMM-LR method, we first introduce a data structure named *cell*. A cell is a structure with information about one possible parsing. The following are kept in the cell:

- LR stack, with information for parsing control.
- Probability array, which includes end point candidates and their probabilities.
- Accept mark, which indicates whether or not parsing ended successfully.

[Definition of symbols]

N: Length of the code sequence for input speech.

M: Number of Markov states for the phone model that is verified.

O_i: i-th code in the code sequence.

$a(i,j)$: Transition probability from state i to state j in the phone model.

$b(i,j,k)$: Output probability for code k when taking transition from state i to state j.

[Algorithm]

1. *Initialization.* Create a new cell C. Push the LR initial state 0 on top of the LR stack of C. Initialize the probability array Q of C;

$$\begin{aligned} Q(0) &= 1.0 \\ Q(i) &= 0.0 \ \ (i = 1...N) \end{aligned}$$

2. *Ramification of cells.* Construct a set

$$\begin{aligned} S = \{(C,a,x) | \ \exists C,s,a,x(\ & \text{C is a cell \&} \\ & \text{C is not accepted \&} \\ & \text{s is a state of C \&} \\ & ACTION(s,a) = x \ \& \\ & x \neq \text{“}error\text{”})\}. \end{aligned}$$

   For each element $(C,a,x) \in S$, do operations below. If a set S is empty, parsing is completed.

3. If $x =$“*shift* s'”, verify the existence of phone a. In this case, update the probability array Q of the cell C by following computation $(i = 1...N, j = 1...M)$.

$$\begin{aligned} P(0,j) &= 0.0 \\ P(i,0) &= Q(i) \\ P(i,j) &= \max(P(i-1,j) \times a(j,j) \times b(j,j,O_i), \\ & \quad P(i-1,j-1) \times a(j-1,j) \times b(j-1,j,O_i)) \\ Q(i) &= P(i,M) \end{aligned}$$

   If $\max Q(i)$ $(i = 1...N)$ is below a threshold level set in advance, the cell C is abandoned. Else push s' on top of the LR stack of the cell C.

4. If $x =$“*reduce* $A \rightarrow \beta$”, do same as in standard LR parsing.

5. If $x =$“*accept*”, check whether $\max Q(i)$ $(i = 1...N)$ reaches the end of speech data. If so, the accept mark of the cell C is switched on.

6. Return to 2.

Generally, a set S constructed in step 2 above is quite large. It is possible to set an upper limit on the number of elements in S by beam-search technique. It is also possible to use the trellis algorithm in computaion 3.

Local ambiguity packing [8] can be used to represent cells efficiently.

4 EXPERIMENTS

HMM-LR method is applied to a Japanese phrase recognition task “*Secretary service of the international conference*”.

4.1 Speech Materials

The recognition experiments are carried out using 25 sentences including 279 phrases uttered by one male speaker.

The speech data is sampled at 12kHz, pre-emphasized by $(1 - 0.97z^{-1})$ and windowed using a 256-point Hamming window every 3 msec. Then a 12-order LPC analysis is carried out. A codebook of 256 LPC spectrum envelopes is generated from 216 phonetically balanced words. The Weighed Likelihood Ratio augmented with power values (PWLR) is used as LPC distance measure for vector quantization.

4.2 Phone Models

Japanese phones are classified into two categories, transient phones and stationary phones. The transient phone model has 4 states and 3 loops to represent its time structure. The stationary phone model has 2 states and 1 loop.

The phone models are trained using an isolated word database [14] (5,240 words). The duration control parameters are modified according to the utterance speed ratio of word utterances to phrase utterances. A Viterbi algorithm is used in the calculation of phone verification.

4.3 Grammars

Two grammars are prepared, a general grammar and a task-specific grammar. The general grammar is designed to cover many linguistic phenomena in Japanese, while the task-specific grammar is designed to cover only linguistic phenomena in our task. The task-specific grammar is created from the statistical data of the ATR linguistic database [15]. Each grammar describes Japanese phrase structures, and is written in the form of context-free grammar. Lexical entries are also written in the form of context-free grammar. Size of grammars is shown in Table 3.

4.4 Experiment Results

Results of phrase recognition experiments are shown in Tables 4 and 5.

In these recognition experiments, beam-search technique is used. The number of elements in S is limited. This upper limit is called *beam*. Recognition accuracy of the system depends on the beam. A larger beam

Table 3: Grammars

	General	*Task-specific*
Rules	1,461	582
Words	1,035	275
LR states	4,359	1,207
Perplexity/phone	5.87	3.86

number makes the system more accurate. As shown in Tables 4 and 5, the recognition rate is saturated as the beam becomes larger. The saturation point is about 100 for the general grammar, and about 30 for the task-specific grammar. Therefore, the beam must be chosen depending on the task difficulty.

The phrase recognition rate with the general grammar is 72% for top candidates and 95% for the five best candidates. With the task-specific grammar, recognition rate is 80% and 99%, respectively.

Table 4: Recognition rate with general grammar

Beam	$rank = 1$	≤ 2	≤ 3	≤ 4	≤ 5
5	38.4	44.8	44.8	44.8	44.8
10	56.6	67.0	69.9	70.6	70.6
20	66.0	78.9	82.4	84.2	85.0
50	71.7	86.0	90.3	92.8	93.4
100~	72.0	85.7	92.1	94.3	95.3

Table 5: Recognition rate with task-specific grammar

Beam	$rank = 1$	≤ 2	≤ 3	≤ 4	≤ 5
5	72.4	82.1	83.9	84.2	84.2
10	77.4	88.2	91.4	92.1	92.8
20	80.3	91.8	94.9	96.8	97.5
30~	79.9	92.8	96.1	97.5	98.6

5 CONCLUSION

In this paper, we proposed a new continuous speech recognition method (HMM-LR) using generalized LR parsing and HMM phone verifiers. The experiment results show that an HMM-LR method is very effective in continuous speech recognition.

We are currently working on enhancing the HMM phone modeling, and using multiple codebooks. Interim experiment results show that the recognition rate with the general grammar is 83% for top candidates and 97% for the three best candidates. With the task-specific grammar, the recognition rate is 90% and 99%, respectively.

ACKNOWLEDGMENTS

The authors are deeply grateful to Dr. Kurematsu, the president of ATR Interpreting Telephony Research Laboratories, all the members of the Speech Processing Department and the Knowledge & Data Base Department for their constant help and encouragement.

References

[1] K.Shikano: "Improvement of Word Recognition Results by Trigram Model", ICASSP 87 (April 1987)

[2] K.F.Lee, H.W.Hon: "Large-Vocabulary Speaker-Independent Continuous Speech Recognition Using HMM", ICASSP 88 (April 1988)

[3] J.R.Rohlicek, Y.L.Chow, S.Roucos: "Statistical Language Modeling Using a Small Corpus from an Application Domain", ICASSP 88 (April 1988)

[4] B.Lowerre, R.Reddy: "The HARPY Speech Understanding System", in *Trends in Speech Recognition* W.A.Lea (ed.), Prentice-Hall (1980)

[5] Y.L.Chow, M.O.Dunham, O.A.Kimball, M.A.Krasner, G.F.Kubala, J.Makhoul, P.J.Price, S.Roucos, R.M.Schwartz: "BYBLOS: The BBN Continuous Speech Recognition System", ICASSP 87 (April 1987)

[6] S.Nakagawa: "Spoken Sentence Recognition by Time-Synchronous Parsing Algorithm of Context-Free Grammar", ICASSP 87 (April 1987)

[7] H.Ney: "Dynamic Programming Speech Recognition Using a Context-Free Grammar", ICASSP 87 (April 1987)

[8] M.Tomita: "Efficient Parsing for Natural Language – A Fast Algorithm for Practical Systems", Kluwer Academic Publishers (1986)

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

[10] D.H.Younger: "Context-Free Languages in Time n^3", Information and Control 10(2), (1967)

[11] J.Earley: "An Efficient Context-Free Parsing Algorithm", CACM 13(2), (1970)

[12] M.Tomita: "An Efficient Word Lattice Parsing Algorithm for Continuous Speech Recognition", ICASSP 86 (April 1986)

[13] H.Saito, M.Tomita: "Parsing Noisy Sentences", COLING 88 (August 1988)

[14] K.Takeda, Y.Sagisaka, S.Katagiri: "Acoustic-Phonetic Labels in a Japanese Speech Database", Proc. of Euro. Conf. on Sp. Tech. Vol.2. (1987)

[15] K.Ogura, K.Hashimoto, T.Morimoto: "An Integrated Linguistic Database Management System", ATR Technical Report TR-I-0036 (August 1988)