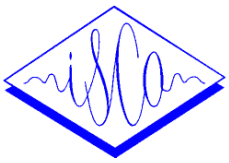

3rd European Conference on
Speech Communication and Technology
EUROSPEECH'93
Berlin, Germany, September 19-23, 1993

VERY-LARGE-VOCABULARY CONTINUOUS SPEECH RECOGNITION ALGORITHM FOR TELEPHONE DIRECTORY ASSISTANCE

Yasuhiro MINAMI, Kiyohiro SHIKANO, Tomokazu YAMADA, and Tatsuo MATSUOKA

NTT Human Interface Laboratories
Musashino-shi, Tokyo 180, Japan

ABSTRACT

This paper describes an algorithm for very large vocabulary (about 80,000 words) continuous speech recognition. To improve recognition accuracy and reduce the amount of computation, we implement an accurate and efficient algorithm based on a two-stage LR parser with phoneme HMMs. In this algorithm, we adopt the forward and backward trellis likelihoods for accurate scoring. We also adopt methods of adjusting windows and merging in phoneme sequences as well as grammatical states for efficient searching. This algorithm is applied to a telephone directory assistance system containing the names of more than 70,000 subscribers. The system recognizes the continuously uttered names and addresses. The results show that the system has good performance in spite of the large perplexity.

Keywords: *Continuous Speech Recognition, HMM, Search Algorithm.*

10.21437/Eurospeech.1993-477

1. INTRODUCTION

The aim of this paper is to present an algorithm for speaker-independent continuous recognition dealing with a vocabulary of about 80,000 words. The main problem in very large vocabulary continuous speech recognition is to reduce the search space without pruning the correct candidate, which requires an accurate and efficient search algorithm.

The HMM-LR system[1] utilizes a generalized LR parser[2] as a language model and the hidden Markov models (HMMs) as phoneme models. When the HMM-LR is applied to large-vocabulary continuous speech recognition, there are three major requirements: (1) accurate scoring for phoneme sequences, (2) reduction of trellis calculation, and (3) efficient pruning of phoneme sequence candidates.

For the accurate scoring, several speech recognition algorithms, which calculate the trellis from the end to the beginning of the utterance (backward trellis likelihood) as well as the forward trellis likelihood, have been proposed[3][4]. We also use forward and backward trellis likelihoods to get accurate scores. For the second requirement, the adjusting window, which uses only the probable part of the trellis according to the predicted phoneme, is used. For the third requirement, an algorithm for merging candidates which have the same allophonic phoneme sequences and the same grammatical states in context-free grammar is used.

2. SPEECH RECOGNITION ALGORITHM

2.1 TWO-STAGE LR PARSER

Our algorithm is based on the HMM-LR[1] algorithm. The LR parser is used as a language model for phoneme prediction. The parser uses an LR table, which is pre-compiled from context-free grammar rules, to predict phonemes. The flow of the recognition process is as follows:

(1) The HMM-LR predicts all possible phonemes using the LR table.

(2) It calculates the trellis likelihood scores using the HMMs of these predicted phonemes to verify them.

(3) In the case of the reduce operation, the state of the LR table moves to the next state.

(4) During the procedure (1)-(3), all possible phoneme sequence candidates are constructed in parallel. These phoneme sequence candidates are pruned using their HMM likelihood. The likelihood calculation is based on the trellis algorithm (not the Viterbi one). The log probabilities are used as the likelihood scores.

(5) The procedure (1)-(4) continues until the HMM-LR does not predict any more phonemes. Finally it outputs the recognition candidate which is accepted by the grammar with the highest likelihood score.

As shown in Figure 1, we introduce a two-stage LR parser that uses two classes of LR tables (a main grammar LR table and several sub-grammar LR tables). These grammar tables are separately compiled from context-free grammar rules. The sub-grammar LR tables deal with semantically classified items like city names, town names, block numbers, and subscriber names. The main grammar LR table controls the relationships between

these semantic items. Dividing the grammar into two classes has two advantages: Each grammar can be compiled separately, so the time needed for compiling the LR table is reduced; and it can easily be adapted to many types of utterances by changing the main grammar rules.

2.2 ACCURATE AND EFFICIENT SEARCH ALGORITHM

2.2.1 ACCURATE SCORING

Our algorithm uses not only a forward trellis but also a backward trellis to calculate the accurate score of a phoneme sequence candidate. The backward trellis likelihood is used as an estimation likelihood of succeeding phoneme sequences of the phoneme sequence candidate.

The phoneme verification process is as follows: The backward trellis is first calculated without any grammar constraints on the HMM phoneme models and stored. The LR parser predicts the next phonemes according to the LR tables. The forward trellis likelihood is calculated by verifying the predicted phoneme HMMs against the input speech. The score of the phoneme sequence is calculated by concatenating the forward trellis and the backward trellis. After all the predicted phonemes have been verified, the phoneme sequences are sorted by their scores and pruned. This beam search procedure is iterated from the beginning to the end of the input speech according to the LR tables.

2.2.2 ADJUSTING WINDOW

As the conventional HMM-LR uses all the input frames, it takes a lot of calculation to calculate the trellis likelihood scores of the predicted phonemes. If the probable part where the predicted phoneme exists is known, the algorithm can only use this part to calculate the trellis likelihood scores. Therefore, we propose an algorithm for determining an adjusting window which restricts the probable part of the trellis for each predicted phoneme (Figure 2).

The adjusting window (shaded rectangle in Figure 2) has a length of 50-frames (400 ms). The score of the adjusting window is calculated by concatenating the forward trellis and backward trellis within the window. In this procedure, the likelihood in the backward trellis (from the end of the utterance to the adjusting window) are multiplied by (1-ε), where ε is a small positive value. The window position is determined by choosing the position on the trellis where the maximum score is achieved. Due to ε, the adjusting window is usually correctly assigned to the phoneme boundary between the predicted phoneme and the next phoneme. After determining the adjusting window position, the forward trellis likelihood within the ad-

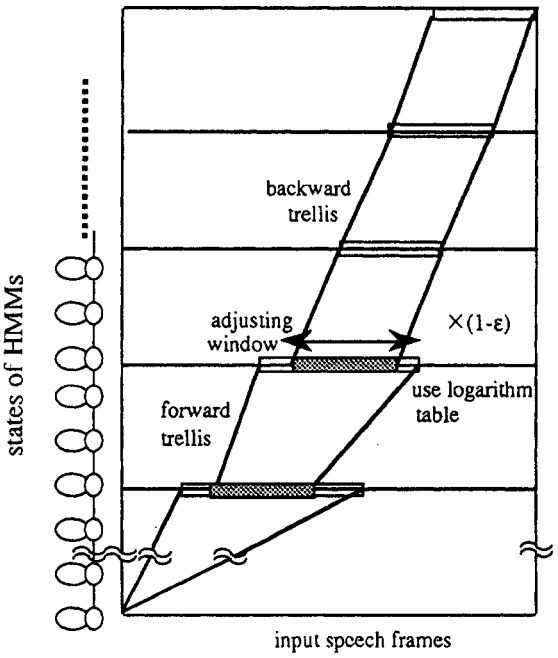

Figure 2. Adjusting window algorithm using forward and backward likelihoods

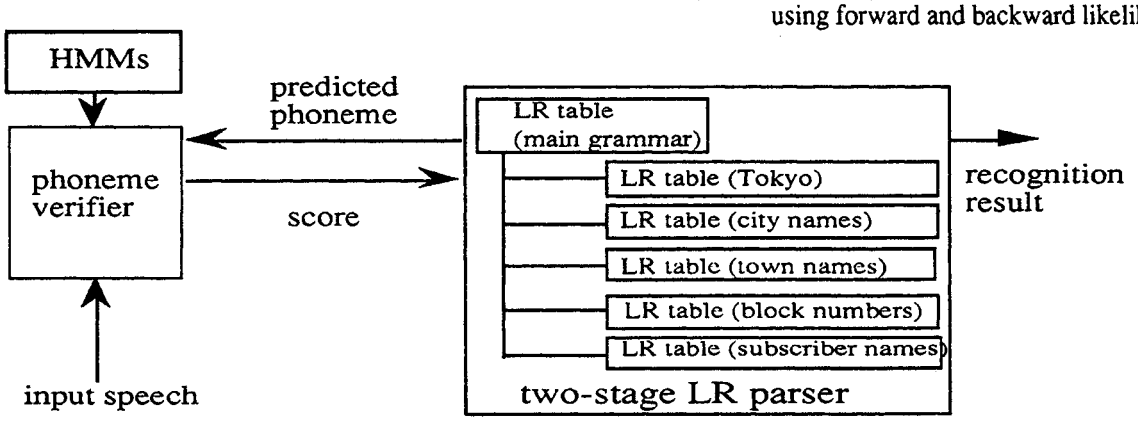

Figure 1. Continuous speech recognition system for telephone directory assistance

justing window is stored as the initial value of the next trellis calculation.

Determining the position of the adjusting window, however, requires an increase in calculation. Therefore, the adjusting window algorithm uses a logarithm table to reduce the number of computations needed to calculate the sum-of-products in the trellis as well as the trellis calculation.

2.2.3 MERGING PHONEME SEQUENCES AND GRAMMATICAL STATES

The LR tables need multiple pronunciation rules to cover allophonic phonemes, such as devoicing and long vowels in Japanese pronunciation. However, these multiple rules cause an explosion of the search space. To make the search space smaller, we merge phoneme sequence candidates as well as grammatical states when the following conditions hold:

(1) the grammar stacks of the candidates are the same,

(2) the candidates are in the same state in the LR tables, and

(3) the phoneme sequences of the candidates are the same (disregarding devoicing and the long vowels).

To further reduce the amount of calculation, we do not recalculate the forward trellis likelihood of a phoneme sequence candidate when the same phoneme sequences already exist in the candidates, even if the grammatical states of the candidates are different.

3. EXPERIMENTS

3.1 RECOGNITION TASK

Our task is a telephone directory assistance task covering two cities (around our laboratories). It contains more than 70,000 subscriber names. In this task, only the subscriber name is indispensable in each utterance. There are no constraints on the semantic items coming from the directory database. For example, the city name or the town name does not constrain the names of subscribers in the city or the town. Table 1 lists the size of the vocabulary for each semantic item.

3.2 PHONEME HMM

All HMMs are continuous Gaussian-mixture-density models with diagonal covariance, and the number of distributions is 4 for each state. These phoneme HMMs have four states and three loops. Their feature vectors consist of 16 cepstrum coefficients, 16 delta cepstral coefficients, and delta power (33 dimensions in total). Phoneme HMMs were trained by concatenated training. We used 56 HMMs as context-independent phoneme-like units (15 vowel models, 28 consonant models, 10 diphthong models, and 3 silence/pause models). We used the continuous speech database of the Acoustical Society of Japan (ASJ) to train the speaker-independent phoneme HMMs. This database was collected from 64 speakers, each of which uttered 150 phonetically balanced sentences.

3.3 GRAMMAR

We prepared two grammar rules as the main grammar.

(1) Grammar for continuous word utterance mode

This is based on a simple syntax that treats several simple sequences of semantic items. In this grammar, each sentence is constructed from an address and a subscriber name.

(2) Grammar for spontaneous utterance mode

This grammar has various rules for interjections, verb phrases and post positional particles etc. This grammar was made by investigating 300 sentences in a simulated telephone directory assistance dialog. This grammar accepted about 70% of the sentences. The rest of the sentences include difficult ones, such that some people explained the Chinese characters of the subscriber's name.

4. EXPERIMENTAL RESULTS

We evaluated the proposed algorithm with speaker-independent HMMs by using 51 sentences including 184 keywords. These utterances were continuous word utterances, simply connecting keywords. These sentences had no interjections or verb phrases. The beam width, number of phoneme sequence candidates, was fixed as 2500. The weighting value (1-ε) for the backward trellis was 0.99. The two recognition experiments, the recognition experiment using the continuous word utterance grammar and the recognition experiment using the spontaneous utterance grammar, were carried out. Eight speakers uttered 51 sentences, which included 184 keywords.

4.1 RESULT FOR CONTINUOUS WORD UTTERANCE GRAMMAR

In the case of using the continuous word grammar as the main grammar in the two-stage LR, the test set phoneme per-

Table 1. Vocabulary sizes for telephone directory assistance task.

semantic item	city names	town names	block numbers	subscriber names
vocabulary size	2	27	620	71,251

plexity was 3.3. Table 2 lists sentence and key-word recognition rates for each speaker. The sentence recognition for the top candidate, the top six candidates and the top 50 candidates are listed as the sentence recognition rates. The average sentence recognition rate was 64% and the average keyword recognition rate was 88%. These results show that the proposed algorithm is very accurate in spite of the large perplexity due to 70,000 subscriber names.

4.2 RESULT FOR SPONTANEOUS UTTERANCE GRAMMAR

In the case of using the spontaneous grammar as the main grammar in two-stage LR, the test set phoneme perplexity was 6.2. Table 3 lists the sentence and key-word recognition rates for each speaker. The average sentence recognition rate for top choice was 50% and the average keyword recognition rate was 81%.

Even using the spontaneous utterance grammar, our algorithm shows good performance, in spite of the fact that its phoneme perplexity is about twice as large as that of the continuous word utterance grammar.

5. CONCLUSION

We proposed an accurate and efficient continuous speech recognition algorithm for a very-large-vocabulary task. The algorithm was implemented in a telephone directory assistance system using a two-stage LR parser with phoneme HMMs. We evaluated its performance for speaker-independent continuous speech recognition using a vocabulary of roughly 80,000 words, including 71,251 subscriber names. The keyword recognition rates were 88% using a continuous word utterance grammar and 81% using a spontaneous speech utterance grammar. These results show that our algorithm for large-vocabulary continuous speech recognition works well.

6. ACKNOWLEDGMENTS

We thank Dr. Sadaoki Furui, the director of the Furui Research Laboratory of the NTT Human Interface Laboratories, and members of the laboratory for useful discussions. We used the ASJ speaker independent continuous speech database and ATR speech database.

REFERENCES

[1] K. Kita, K. Kawabata, and H. Saito, "HMM Continuous Speech Recognition Using Predictive LR Parsing," ICASSP 89, pp. 703-706 (May 1989).

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

[3] S. Austin, R. Schwartz, and P. Placeway, "The Forward-Backward Search Algorithm," ICASSP 91, pp. 697-700 (May 1991).

[4] R. Schwartz and S. Austin, "A Tree-Trellis Based Fast Search for Finding N Best Sentence Hypotheses in Continuous Speech Recognition," ICASSP 91, pp. 705-708 (May 1991).

Table 2. Speaker-independent continuous word utterance recognition results for continuous word utterance grammar.

speaker	M02	M06	M11	M12	M13	M14	F11	F12	average
sentence recognition rate(%) (=1)	75	76	65	69	61	43	59	67	64
sentence recognition rate (%)(<= 6)	90	94	80	92	78	57	71	80	80
sentence recognition rate (%)(<= 50)	94	100	96	98	92	78	92	88	92
keyword recognition rate (%)	93	93	90	89	88	80	85	89	88

Table 3. Speaker-independent continuous word utterance recognition results for spontaneous utterance grammar.

speaker	M02	M06	M11	M12	M13	M14	F11	F12	average
sentence recognition rate(%) (=1)	67	59	67	49	41	22	41	57	50
sentence recognition rate(%) (<=6)	80	67	71	57	45	27	47	63	57
sentence recognition rate(%)(<= 50)	86	73	75	63	57	45	63	69	66
keyword recognition rate(%)	90	85	90	76	78	68	74	83	81