TR-IT-0271

Acoustic and Pronunciation Modeling in Automatic Speech Recognition

深田 俊明
Toshiaki Fukada

1998年9月4日

In conventional speech recognition systems, after input speech is pre-processed by speech analysis, recognition results are obtained by finding (or searching) for probable hypotheses in terms of maximum likelihood criterion by using three knowledge resources, that is, acoustic models, language models and pronunciation models (i.e., dictionary). To develop sophisticated speech analysis, acoustic modeling, language modeling, pronunciation modeling and search are indispensable for improving performance of speech recognition systems. This report presents novel algorithms to improve acoustic models and pronunciation models. As for improved acoustic modeling, three kinds of approaches can be considered; (1) improvement of the current modeling, (2) incorporation of new acoustic features for acoustic modeling, and (3) development of a new acoustic modeling paradigm. In this report, (1) acoustic modeling using speaker normalization technique, (2) speech recognition using segment boundary information, and (3) model parameter estimation for mixture density segment models, are proposed for the above three approaches, respectively. In spontaneous speech recognition, as word pronunciation varies more than in read speech, actual pronunciation variations have to be incorporated into the pronunciation dictionary. This report presents a method for automatically generating multiple pronunciation dictionaries based on a pronunciation neural network that can predict plausible pronunciations from the standard pronunciation. Experimental results on spontaneous speech recognition show that automatically-derived pronunciation dictionaries give higher recognition rates than the conventional dictionary.

Contents

Chapter 1

Introduction

1.1 Background

Speech is the most natural form of human communication. Therefore, speech recognition devices find widespread use in many applications as man-machine interface for aiding productivity and convenience (e.g., manufacturing control application, voice-command in a car environment, dictation system), information retrieval over the telephone, and speech-to-speech translation system. The goal of automatic speech recognition (ASR) is to develop devices that transcribe human speech correctly into written text. Research in ASR originated as early as 1950's and the performance of ASR systems has improved dramatically in the last few years. Recently, speech recognition systems have started to appear in homes and offices by way of dictation software (e.g. IBM's Via Voice [1][2], Microsoft's Whisper [3]), running on regular mid-range PCs. People are increasingly aware of the availability of such systems and their widespread use is set to explode. Although ASR technology is now readily available, the performance of the resulting systems is still not comparable to that achieved by human beings and a perfect solution has not been found yet.

Most speech recognition systems make use of two major knowledge sources: a family of acoustic models and a language model. The language model dramatically improves the system performance in continuous speech recognition. However, the performance is highly dependent on the ability of the system to discriminate between the different sounds of speech. Therefore, the development of good acoustic models is indispensable for high performance speech recognition.

In addition, it is widely acknowledged that recognition performance decreases severely for conversational speech compared to the performance of read speech even when the same sequence of words is spoken. One of the reasons is that word pronunciation in spontaneous speech varies more than in read speech. This indicates that actual pronunciation variations have to be incorporated into the pronunciation dictionary.

The focus of this report, as suggested by its title, is on enhancing the abilities of acoustic models and pronunciation models with the aim of improving the overall recognition performance of the system.

1.2 This report

The remainder of this chapter introduces the major original contributions of this report.

The main research done for this report, improvements to state-of-the-art acoustic modeling, is described in Chapter 2, 3, and 4. Chapter 2 describes a method for speaker normalization to improve conventional acoustic models. Chapter 3 explains the importance of segment boundary information as a new feature for boosting recognition performance. In Chapter 4 model parameter estimation for mixture density segment models is exploited.

Chapter 5 describes a method for automatically generating a multiple pronunciation dictionary based on a pronunciation neural network to cope with pronunciation variations in spontaneous, conversational speech.

The final chapter brings together the conclusions drawn from Chapter 2 to Chapter 5 and summarizes the achievements of this work. Scope for further work is also discussed.

1.3 Original contribution

This report describes new approaches to generate sophisticated acoustic and pronunciation modeling within automatic speech recognition systems. The major original contributions are as follows:

- To reduce speaker variability in acoustic models, a novel speaker normalization scheme based on three dimensional (3-D) Viterbi decoding (i.e. time, state and warping factor) is developed. (Chapter 2)
- The 3-D Viterbi based normalized models are shown to improve recognition accuracy over speaker independent models, gender dependent models and conventional speaker normalized models in a recognition system with a one-pass search. (Chapter 2)
- Bi-directional recurrent neural network (BRNN) is successfully applied to segment boundary estimation. (Chapter 3)
- Segment boundary information is probabilistically incorporated into hidden Markov models (HMMs). (Chapter 3)

- Segment boundary information obtained from a BRNN is shown to be effective for the reduction of computation complexity and the improvement of recognition performance. (Chapter 3)
- A model parameter estimation method for polynomial segment models, where their mean and variance trajectories are specified with an arbitrary regression order, is developed. (Chapter 4)
- It is shown that modeling both the mean and variance trajectories is consistently superior to modeling only the mean trajectory in a vowel classification task. (Chapter 4)
- A method of neural network based automatic pronunciation modeling is developed and shown to be effective for improving recognition performances in spontaneous speech recognition. (Chapter 5)
- Language statistics (i.e., word bigram) are successfully used to improve pronunciation modeling for word boundary phonemes. (Chapter 5)

Chapter 2

Speaker normalized acoustic modeling

This chapter describes a novel method for speaker normalization based on a frequency warping approach to reduce variations due to speaker-induced factors such as the vocal tract length. In our approach, a speaker normalized acoustic model is trained using time-varying (i.e., state, phoneme or word dependent) warping factors, while in the conventional approaches, the frequency warping factor is fixed for each speaker. These time-varying frequency warping factors are determined by a 3-dimensional (i.e., input frames, HMM states and warping factors) Viterbi decoding procedure.

2.1 Speaker normalization

Robust and precise acoustic modeling is an indispensable technique for achieving high recognition performance. In most current speaker-independent speech recognition systems, acoustic models are trained using a large amount of speech uttered by a wide variety of speakers. The spectral distributions often exhibit high variance and hence high overlap among different phonemes. Therefore, recognition performance saturates even if a number of mixtures and states are used or the context is increased. Consequently, research efforts have been conducted to reduce variations due to speaker-induced factors based on speaker normalization [4][5][6][7][8][9], speaker clustering[10] or hybrid methods [11][12][13]. In recent years, many researchers have been working on speaker normalization, since one of the major sources of interspeaker variance is the vocal tract length. Acoustic modeling based on speaker normalization techniques can be roughly divided into two approaches:

1. frequency warping (FWP) [6][7][9]

2. maximum likelihood linear regression (MLLR) [8].

In the conventional FWP-based approaches, the frequency warping factor is fixed for each speaker. That is, these approaches do not have a framework of phoneme or HMM state dependent frequency warping, while in the MLLR-based approach, it is possible to define regression classes and associate a regression matrix with each class. Also, a phoneme or allophone dependent warping procedure would be reasonable, when training speech samples are biased to a certain speaker or gender for some phoneme contexts or allophones.

In the following section, we present FWP-based acoustic modeling in which warping factors are dynamically changed during an utterance. These frequency warping factors are determined by a 3-dimensional (i.e., input frames, HMM states and warping factors) Viterbi decoding procedure. In the proposed method, the recognition procedure can be performed with a one-pass search, while in most current FWP-based approaches, a multiple-pass search is required at the recognition stage.

2.2 Normalization procedures

2.2.1 3-D Viterbi decoding

The key point of the proposed normalization procedures is to perform a Viterbi search on a 3-D trellis space composed of input frames, HMM states and warping factors (see Fig 2.1). Note that the conventional frequency warping based normalization is done by finding a warping factor for each speaker which yields the highest likelihood among all possible warping factors by a 2-D (i.e., input frames and HMM states) trellis search (see Fig 2.2).

2.2.2 Training procedure

In the training stage, as transcriptions of speech are known, a transition of warping factors can be obtained by aligning the HMM states with the maximum likelihood criteria. The following procedure is used for acoustic model training:

1. Set the initial warping factor to the standard value (e.g., $\alpha = 0.46$ in the following experiment), for all speakers and generate the initial HMM.
2. Align the training utterances based on 3-D Viterbi decoding using the current HMM and find the optimal warping factor for each HMM state.
3. Train an HMM using the feature vector sequence of the optimal warping factors
4. Go to step 2 until there is no significant change between consecutive training iterations.

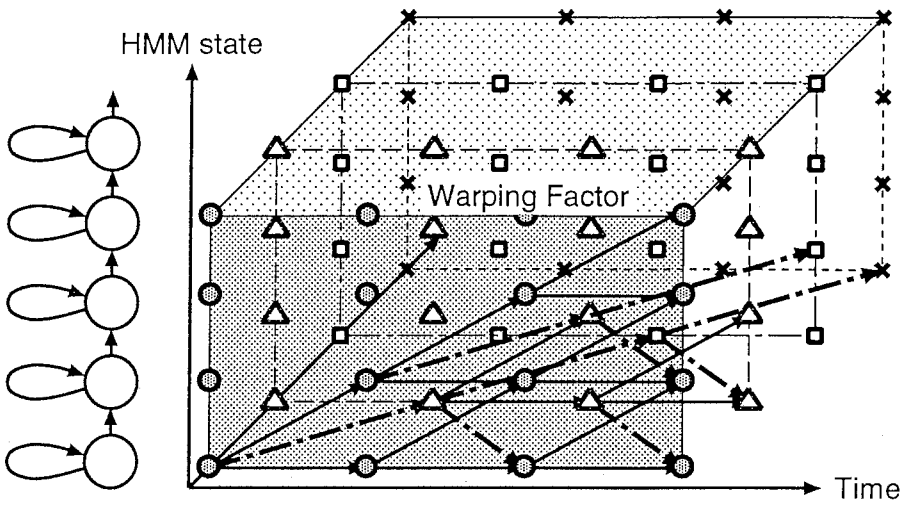


Figure 2.1: The proposed FWP scheme.

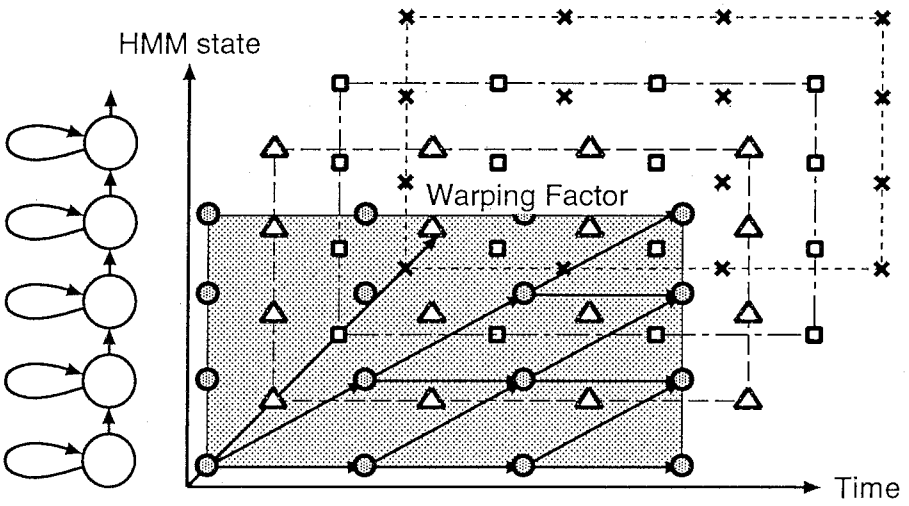


Figure 2.2: The conventional FWP scheme.

In this procedure, we apply constraints to the 3-D Viterbi decoding procedure so as not to change the warping factor too rapidly (see Section 2.2.3).

2.2.3 Recognition procedure

In the recognition stage, a transition matrix of warping factors and a phoneme (or word) sequence of speech are obtained by finding an optimal path with the highest likelihood. Figure 2.3 shows the recognition algorithm. In this figure, S, Q, D and N are the initial state sets, number of states, number of warping factors and number of frames, respectively.

Initialization:
for $q = 1$ to Q
  for $d = 1$ to D
    if $(q, d) \in \mathrm{S}$ then
      $P(q, d, 0) = \log \pi(q, d)$, where $\sum_{(q,d)\in \mathrm{S}} \pi(q, d) = 1$
    else
      $P(q, d, 0) = -\infty$

Recognition:
for $n = 1$ to N
  for $q = 1$ to Q
    for $d = 1$ to D
      $P(q, d, n) = \max_{q', d'} \{P(q', d', n-1) + \log a(q', q) + \log f(d', d)\} + \log b(q, \mathbf{x}(d, n))$

Figure 2.3: Recognition algorithm.

π, P, $a(q', q)$, $f(d', d)$, b and $\mathbf{x}$ are the initial state probability, accumulated probability, transition probability from state q' to q, transition probability from warping factor d' to d, output probability and feature vector, respectively.

In this chapter, the transition probability of warping factor $f(d', d)$ is given as:

$$f(d', d) = \begin{cases} 1.0, & |d' - d| \leq w \\ 0.0, & |d' - d| > w. \end{cases} \tag{2.1}$$

Here, we set to $w = 1$ for inter-phoneme state transitions and $w = 0$ for intra-phoneme state transitions (here denoted as FWP1). These constraints can be considered reasonable because the warping factor is not expected to change too rapidly. Note that the proposed speaker normalization procedure is equivalent to the conventional method (e.g. [6][7][9]) if $w = 0$ for any state transition (here denoted as FWP0).

2.3 Experiments

To investigate the relative effectiveness of the proposed method, we conducted continuous speech recognition experiments on a Japanese spontaneous speech database[14].

2.3.1 The front-end

As for the front-end, we use mel-cepstral analysis [15], though the proposed method is applicable to any kinds of spectral analysis methods.

Table 2.1: Recognition performance. (Phone accuracy %)

feature parameter	without LM	with LM
LPC-cepstrum	44.0	52.2
MFCC	48.2	55.5
Mel-cepstrum	**49.6**	**56.4**

We represent the model spectrum $H(e^{j\omega})$ by the M-th order mel-cepstral coefficients $\tilde{c}(m)$ as follows:

$$H(z) = \exp \sum_{m=0}^{M} \tilde{c}(m)\, \tilde{z}^{-m} \tag{2.2}$$

where

$$\tilde{z}^{-1} = \frac{z^{-1} - \alpha}{1 - \alpha z^{-1}}, \qquad |\alpha| < 1. \tag{2.3}$$

The phase characteristic of the all-pass transfer function $\tilde{z}^{-1} = e^{-j\tilde{\omega}}$ is given by

$$\tilde{\omega} = \tan^{-1} \frac{(1-\alpha^2)\sin\omega}{(1+\alpha^2)\cos\omega - 2\alpha}. \tag{2.4}$$

For example, for a sampling frequency of 16kHz, $\tilde{\omega}$ is a good approximation to the mel scale based on subjective pitch evaluations when $\alpha = 0.42$. If we choose $\alpha = 0.46$, the mel scale is quite similar to that used in mel-frequency cepstral coefficient (MFCC) analysis.

To obtain an unbiased estimate, we use the following criterion and minimize it with respect to $\{\tilde{c}(m)\}_{m=0}^{M}$.

$$E = \frac{1}{2\pi} \int_{-\pi}^{\pi} \{\exp R(\omega) - R(\omega) - 1\}\ d\omega \tag{2.5}$$

where

$$R(\omega) = \log I_N(\omega) - \log \left| H(e^{j\omega}) \right|^2 \tag{2.6}$$

and $I_N(\omega)$ is the modified periodogram of a weakly stationary process $x(n)$ with a time window of length N. Since E is convex with respect to $\tilde{c}(m)$, $\{\tilde{c}(m)\}_{m=0}^{M}$ can be obtained by the Newton-Raphson method[15].

To show the effectiveness of mel-cepstral analysis, we performed simple recognition experiments using the TIMIT database. 3 state and 5 mixture context-independent HMMs (61 phone sets) were trained for LPC-cepstrum, MFCC and mel-cepstrum. A phoneme bigram was used for the language model (LM). Table 2.1 shows the differences in recognition performance for these three types of feature parameters. We can see from these results that mel-cepstral analysis is one of the most useful pre-processing methods for speech recognition.

Frequency warping can be done by changing α in Eq. (2.3). Figure 2.4 shows examples of frequency warping for several α values. Figure 2.5 shows the warped spectra by these values.

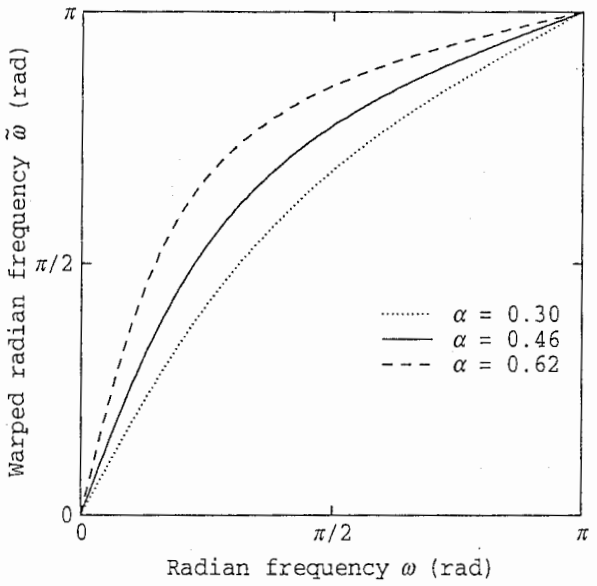
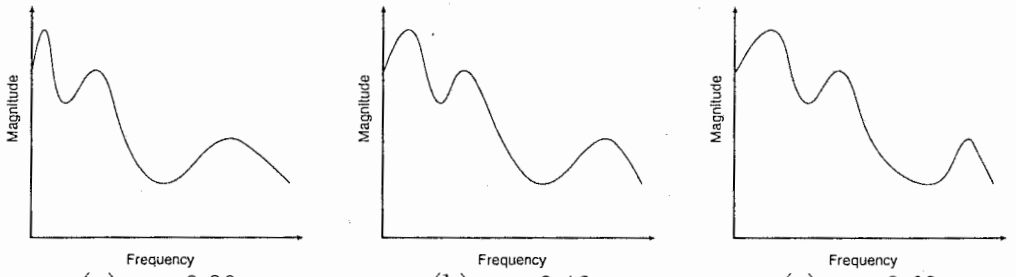


Figure 2.4: Frequency warping for several α values.



(a) $\alpha = 0.30$ (b) $\alpha = 0.46$ (c) $\alpha = 0.62$

Figure 2.5: Spectral envelopes for different α.

2.3.2 Conditions

230 speakers were used for training and 42 speakers for evaluation. A 26-dimensional feature vector (12-dimensional mel-cepstrum + power and their derivatives) computed with a 25.6 msec window duration and a 10 msec frame period were used for acoustic modeling. First, shared-state HMMs (800 states in total) with 5 Gaussian mixture components per state[16] were trained by using an initial warping factor of $\alpha = 0.46$ for all speakers (gender-independent HMM; GI-HMM). Then, we generated two kinds of speaker normalized models (i.e., FWP0 and FWP1) described in Section 2.2. As for the FWP0 training, the best warping factor was determined for each speaker. The normalization session described in Section 2.2.2 was repeated four times. The GI-HMM topology was consistently used for every iteration. For feature parameter sets, 9 kinds of warping factors ($D = 9$) were considered in steps of 0.04 from $\alpha = 0.30$ to 0.62. Gender-dependent HMMs (GD-HMM) were also used for comparison. We used spontaneous speech recognizer using

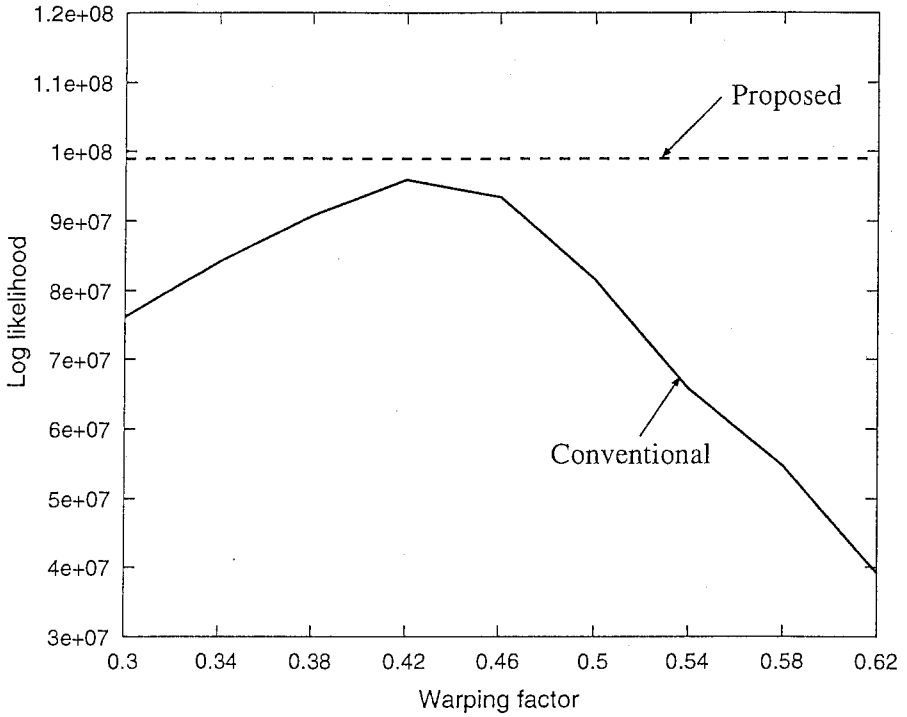


Figure 2.6: Log likelihood for each warping factor.

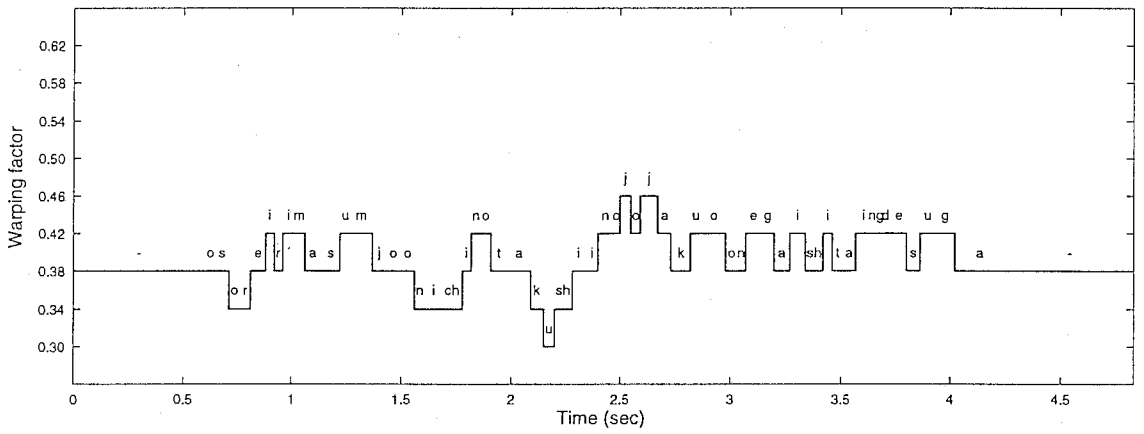


Figure 2.7: Warping factors obtained by the 3D Viterbi decoding (`/osoreirimasu mjoonichino takushiino jojakuo onegaishitaingdesuga/` , (*Excuse me but I would like to make a reservation for a taxi for tomorrow.*)).

cross-word context constrained word graphs[17]. The test vocabulary consists of about 7,000 words, and the variable-length N-gram[18] was used for the language model.

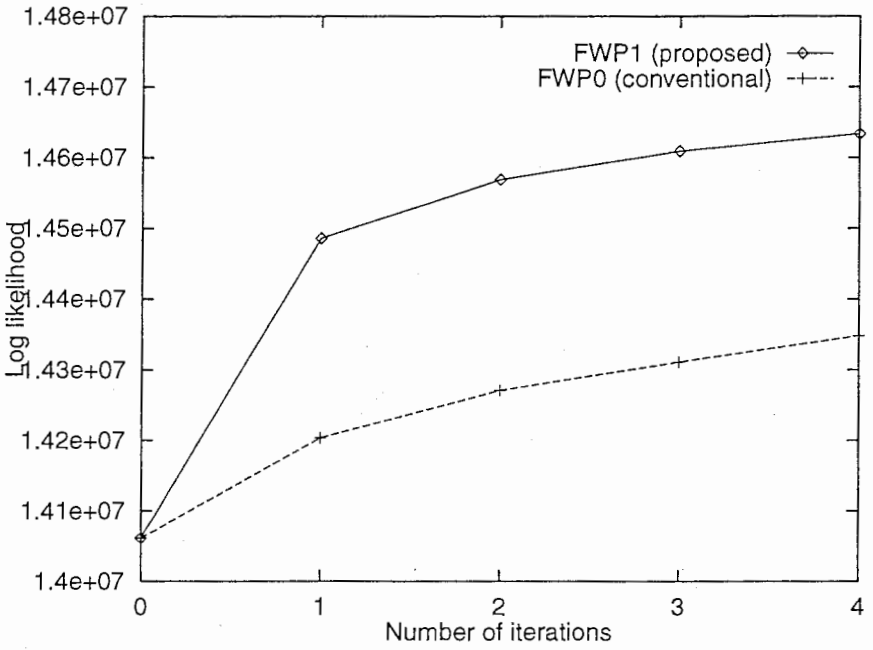


Figure 2.8: Log likelihood for each iteration.

2.3.3 Comparison of speaker normalized models

Figure 2.6 shows the log likelihoods for a female utterance obtained by the forced alignment using the Viterbi decoding algorithm with several warping factors. The log likelihood obtained by the proposed method is also shown as the dotted line. The phoneme dependent warping factors for this utterance obtained by the proposed method is shown in Figure 2.7. First, the conventional method gives the highest likelihood at $\alpha = 0.42$ which is the smaller warping factor than the initial warping factor, $\alpha = 0.46$. That is, their spectra are warped to lower frequency. This will be reasonable for female speech, since the acoustic model is trained by using both the male and female speech. Second, the likelihood obtained by the proposed method is higher than any other likelihoods obtained by the conventional method. Third, the phoneme dependent warping factors seem to be selected appropriately, since bigger α (i.e., $0.50 \sim 0.62$), which will often be selected for male speech, are not selected during the utterance.

The increase of the total log-likelihood during the iterative acoustic model training can be seen in Figure 2.8. The solid line shows the case for FWP1 and the dotted line for FWP0. The likelihood of iteration 0 is the likelihood of the GI-HMM. FWP1 yielded a consistently higher acoustic likelihood than FWP0 for each iteration. From these results, we can expect that the proposed speaker normalized model based on 3-D Viterbi decoding reduces interspeaker variability more than the conventional normalization method and results in a certain improvement in speech recognition.

Table 2.2: Word accuracy and relative improvement from GI-HMM (%).

acoustic model	accuracy	improvement
GI-HMM	74.6	—
GD-HMM	74.0	-2.3
FWP0	73.3	-5.0
FWP1	**77.1**	**9.7**

The mean and standard deviation of the warping factors for each iteration are shown in Figure 2.9 and Figure 2.10, respectively. These statistics are calculated from the distribution (histogram) of the warping factors obtained from the 3-D Viterbi based alignment. We can see from these figures that the standard deviations of the proposed method (FWP1) are greater than those of the conventional method (FWP0). This is because the proposed method has a chance to vary the warping factor for each phoneme during the utterance, while the warping factor is fixed for each speaker in the conventional method.

2.3.4 Recognition results

The recognition results are shown in Table 2.2. From these results, it can be seen that the proposed speaker normalized model (FWP1) yielded a better performance than GI-HMM, GD-HMM and the conventional speaker normalized model (FWP0). Recognition performances of multiple iterations are shown in Table 2.3. In this table, FWP0 and FWP1 with no iteration mean that 9 kinds of feature parameters were used as inputs and recognition was performed using the GI-HMM with $w = 0$ for FWP0 and $w = 1$ for FWP1. From this table, multiple training iterations improve the recognition performances for both FWP0 and FWP1 cases. It is also interesting that using the 3-D Viterbi based decoding procedure with the unnormalized model (i.e., GI-HMM) still gives a 6.1 % improvement over the GI-HMM (from 74.6 % to 76.2 %).

In our experiment the recognition result with the highest likelihood among the several frequency warping factors, is determined in a time-synchronous one-pass beam search. The conventional speaker normalized model FWP0 (73.3 %) was surprisingly slightly worse than the GI-HMM (74.6 %). This could happen because we kept a constant beam-width, limited by memory requirements of the search engine, across all experiments, which produced search errors in the FWP0 case due to large local fluctuations in likelihood. Nevertheless, FWP1 achieved a 9.7 % improvement compared to the GI-HMM (from 74.6 % to 77.1 %). Note that in most current FWP-based approaches, a multiple-pass search is required at the recognition stage, while in the proposed method, the recognition procedure to find the best hypothesis and simultaneously select the best (time-dependent) warping factor can be performed with a one-pass search.

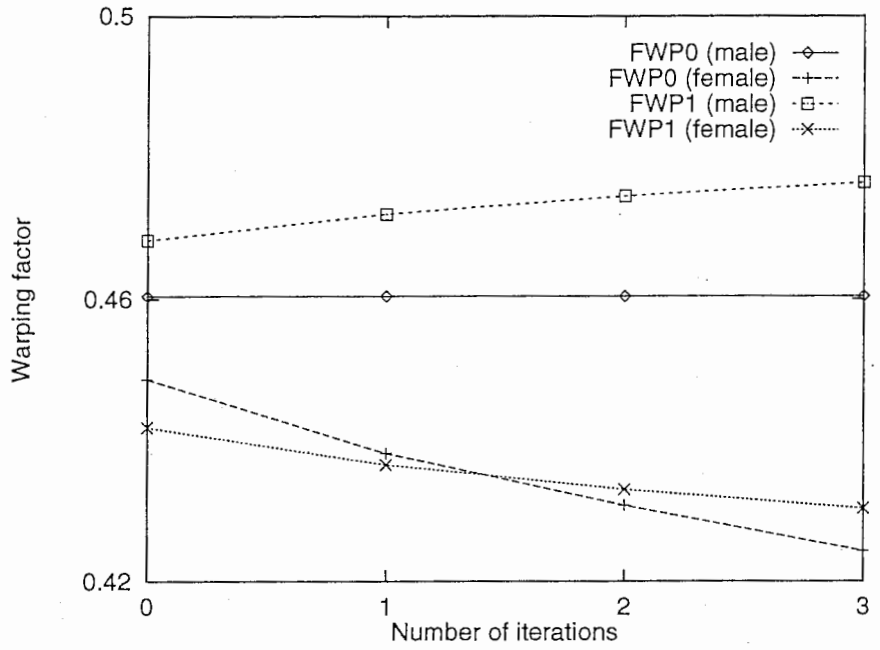


Figure 2.9: Mean of warping factors.

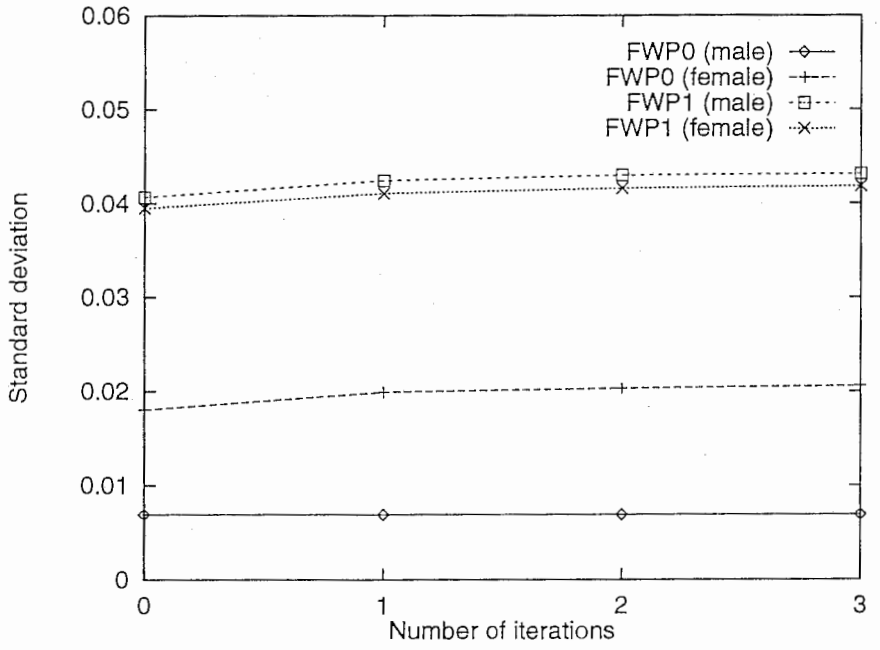


Figure 2.10: Standard deviation of warping factors.

2.4 Conclusion

In this chapter, we have proposed a new acoustic modeling technique based on a 3-D Viterbi decoding procedure which aims at normalizing speaker's variability. This method

Table 2.3: Recognition performance improvements for FWP0 and FWP1 after four times of training iterations.

acoustic model	no. of iterations	accuracy (%)
FWP0	0	72.7
FWP0	4	73.3
FWP1	0	76.2
FWP1	4	77.1

has a framework which makes it possible to vary the frequency warping factor with arbitrary units (i.e., state, phoneme, word, etc.) during an utterance. The conventional frequency warping based acoustic modeling can be viewed as a special case of the proposed modeling (i.e., $w = 0$ in Eq. (2.1)). The experimental results on spontaneous speech recognition showed that the proposed models yielded a 9.7 % improvement in word accuracy compared to the standard speaker-independent model.

Chapter 3

Incorporating segment boundary information

This chapter describes a segment (e.g. phoneme) boundary estimation method based on bidirectional recurrent neural networks (BRNNs). The proposed method only requires acoustic observations to accurately estimate segment boundaries. Experimental results show that the proposed method can estimate segment boundaries significantly better than an HMM based method. Furthermore, we incorporate the BRNN based segment boundary estimator into the HMM based and segment based recognition systems. The BRNN is also applied for the purpose of automatic segmentation.

3.1 Introduction

Accurately estimating segment boundaries is one of the most important techniques in (1) automatic segmentation [19][20] for acoustic model training and in (2) preprocessing for segment based speech recognition [21]. Conventional segmentation algorithms attempt to locate optimal segment boundaries either by minimizing distortion metrics through dynamic programming based methods [19] or by maximizing the metric score of acoustic models [20]. These algorithms, however, require acoustic (and language or duration) models to obtain adequate results. Nevertheless, even with such models, the estimated results are generally still poor because the approaches are not designed to detect boundaries, but rather to minimize or maximize scores for acoustic observations (e.g. cepstrum). Neural networks (NNs) that show a high performance for many classification tasks are suitable for estimating accurate boundaries. There have recently been several reports that boundary information obtained from feed-forward multilayer perceptrons (MLP) improves recognition performance [22][23].

In this chapter, we propose a segment (e.g. phoneme) boundary estimation method based on bi-directional recurrent neural networks (BRNNs). A BRNN can be trained without the

limitation of using a fixed size input window, and it gave better classification performance than a regular RNN on test problems [24]. The proposed method only requires acoustic observations to estimate segment boundaries, and networks are trained to accurately detect segment boundaries. We apply segment boundary estimation

1. to improve recognition performance using the network outputs and

2. to reduce computational complexity of segment based recognition using estimated candidates.

3.2 Segment boundary estimation using neural network

3.2.1 BRNN structure

Bi-directional Recurrent Neural Networks (BRNNs) [24] are used for segment boundary estimation, and their structure is illustrated in Figure 3.1. BRNNs can recursively accommodate forward and backward inputs to predict current output by only using a single network. A conventional RNN only uses input information from one side for the currently estimated output.

3.2.2 Input and output

Feature parameter vectors (e.g. cepstrum) are used for the BRNN input, and the outputs (target values) are chosen according to whether the current frame is a boundary (out=1) or not (out=0).

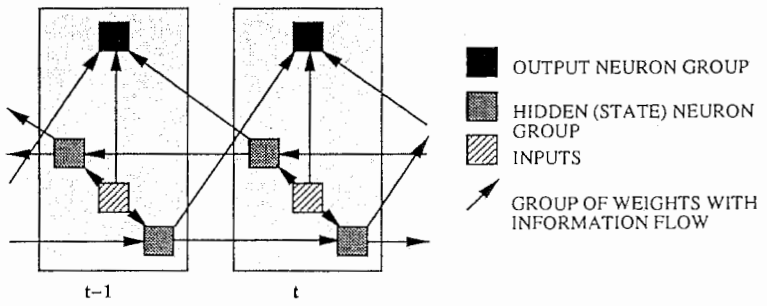


Figure 3.1: Bi-directional recurrent neural networks.

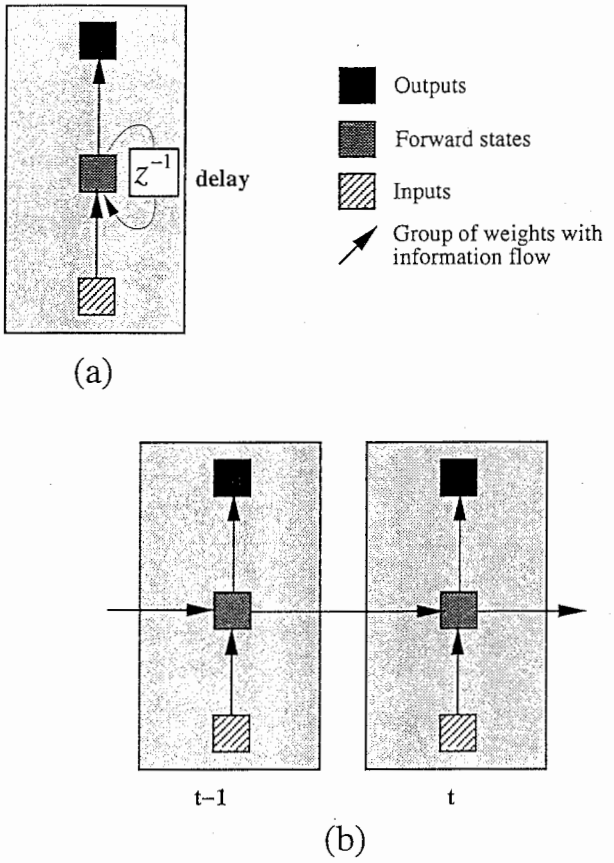


Figure 3.2: Recurrent neural networks. ((a) with a delay line, (b) unfolded in time for two time steps)

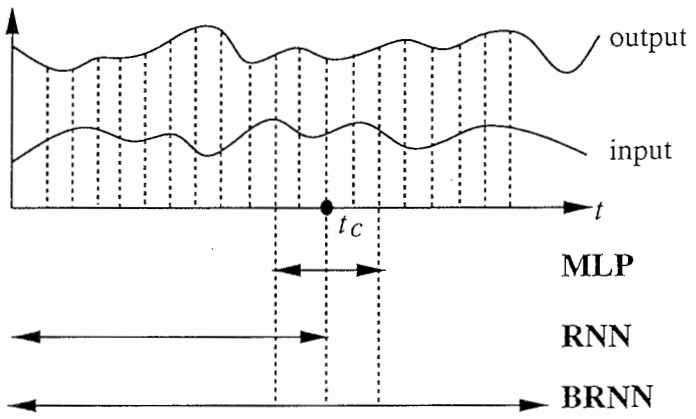


Figure 3.3: Visualization of the amount of input information used for prediction by different network structures.

3.2.3 Training

The same algorithm as used in regular RNN training can be used for BRNN training [24]. The Euclidean distance measure between the desired and the estimated output vectors

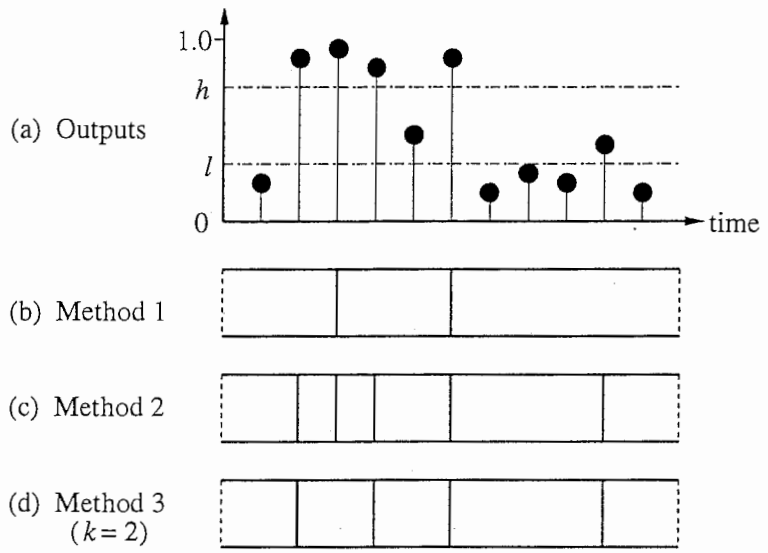


Figure 3.4: Boundary estimation results for method 1, 2 and 3.

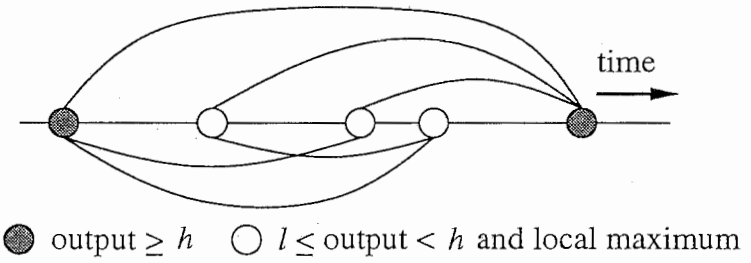


Figure 3.5: Lattice representation of segment boundary candidates.

on the mean squared error criterion is used as objective function. Connection weights is updated with some variation of a gradient descent method.

3.2.4 Segment boundary estimation algorithm

To determine segment boundaries from the BRNN outputs as shown in Figure 3.4 (a), the following three methods are used. A certain time point (frame) t is said to be a boundary if:

1. the output at t is above threshold h and is a local maximum;

2. the output at t is above threshold h or is a local maximum between a lower threshold $l(< h)$ and h;

3. the same as method 2, but for segment boundaries whose outputs are above threshold h, only every k-th time point is taken.

The results of application of the method 1, 2 and 3 for the output shown in 3.4 (a) are depicted in 3.4(b), (c) and (d), respectively (in this case, k in method 3 is set to 2). Method 1 is the simplest method and can be directly used to determine segment boundaries. Method 2 and 3 can be used to characterize possible boundaries. In these methods, while a lot of correct boundaries can be obtained, insertion errors might increase compared to method 1. The reason that boundary candidates are taken every k-th time point is to reduce a number of insertion errors.

Method 3 would be useful technique when obtained segment boundaries can be evaluated with another post processing for determining the optimal boundaries among them. For example, we call a segment boundary detected above h as *main boundary* and detected between l and h as *secondary boundary*. As shown in Figure 3.5 main and secondary boundaries are represented as black and white circles, respectively. When all segment boundary candidates between two main boundaries are connected (linked), segment lattice can be generated as shown in Figure 3.5. As each arc is considered as phoneme segment, most probable path in terms of ML (maximum likelihood) criterion can be determined by performing phoneme recognition with acoustic model (e.g. HMM or segment model).

3.3 Evaluation of segment boundary estimation

To investigate whether the proposed methods are useful, (1) a comparison between estimated boundaries obtained by method 1 and HMM based phoneme recognition results, (2) a comparison between BRNN and MLP and (3) a comparison among methods 1, 2 and 3 were done using the TIMIT database[25].

3.3.1 Conditions

26-dimensional MFCCs (12-dimensional MFCC + power and their derivatives) computed with a 25.6 msec window duration and a 10 msec frame period were used as the BRNN inputs. Based on the phoneme label information of the database, 1.0 is given for outputs if the current frame is a segment (phoneme) boundary, 0.5 is given if the current frame is right next to the boundary, and for anything else 0 is given. BRNN training was done using 462 speakers with 1,000 iterations. 10 forward and backward states and 30 hidden states for the BRNN were used. A total of weights, W_{all} is given by:

$$\begin{aligned} W_{all} &= (U_I + 1)U_H + (U_H + 1)U_O + (U_F + U_I + 1)U_F \\ &\quad + U_F U_H + (U_B + U_I + 1)U_B + U_B U_H, \end{aligned} \tag{3.1}$$

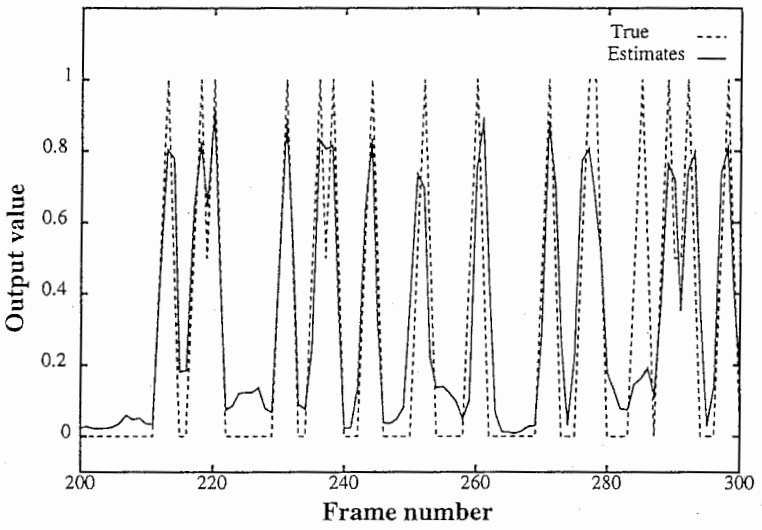


Figure 3.6: An example of BRNN outputs.

where U_I, U_O, U_F, U_B and U_H are numbers of input, output, forward, backward and hidden units, respectively. Thus W_{all} in this experimental setting becomes 2,181. Test data was used for 168 speakers (50,318 boundaries, about 410,000 frames). The mean squared errors between true (target) and estimates were 0.0604 for training data and 0.0621 for testing data, respectively. Thresholds in methods 1, 2 and 3 were experimentally set to $h = 0.4$ and $l = 0.1$. This network is trained using the regular sigmoid activation function and a slightly modified version of a first order nonlinear optimization technique named "resilient propagation" (RPROP) [26], expanded to a RPROP through time variant.

Figure 3.6 shows an example of outputs for BRNN based segment boundary estimation. The dotted line represents a true (target) output and the solid line represents an estimated output. These results were obtained for open test data using the network trained as described in Section 3.3.1.

3.3.2 Evaluation criterion

To evaluate the estimated results, $Correct = H/N \times 100(\%)$ and $Accuracy = (N - D - I)/N \times 100(\%)$ were used, where

- H (Hit) : estimated boundary was within a $\pm M$ frame margin of the true boundary
- D (Deletion) : no estimated boundary was within $\pm M$
- I (Insertion) : estimated boundary was not within $\pm M$
- N : total number of true boundaries ($N = H + D$).

Table 3.1: Estimation results based on the BRNN (method1).

	Margin		
	0	1	2
Hit	23,175	38,248	40,056
Deletion	27,143	12,070	10,262
Insertion	18,983	4,066	2,293
Correct	46.06	76.01	79.61
Accuracy	8.33	67.93	75.05

Table 3.2: Estimation results based on the HMM.

(a) context-independent model

	Margin		
	0	1	2
Hit	8,806	28,214	38,847
Deletion	41,512	22,104	11,471
Insertion	35,372	16,253	5,915
Correct	17.50	56.07	77.20
Accuracy	-52.80	23.77	65.45

(b) context-dependent model

	Margin		
	0	1	2
Hit	14,198	35,967	42,611
Deletion	36,120	14,351	7,707
Insertion	32,970	11,521	5,110
Correct	28.22	71.47	84.68
Accuracy	-37.31	48.58	74.53

Note that if several estimated boundaries i were within $\pm M$, $i-1$ were treated as insertions.

3.3.3 Results

Comparison between method 1 and an HMM based approach

Estimation results with margins of $M = 0, 1, 2$ are shown in Table 3.1. To produce reference results, an evaluation was performed by using the boundaries obtained through HMM based phoneme recognition with HMMs and a phoneme bigram language model.

To compare the proposed method with the HMM based method, context-independent

Table 3.3: Comparison of root mean squared error between BRNN and MLP. ($\times 10^{-2}$)

Structure (weights)	Training	Test
MLP-1 (2,241)	6.85	7.00
MLP-3 (2,241)	6.53	6.67
MLP-5 (2,245)	6.51	6.66
MLP-7 (2,209)	6.58	6.71
BRNN (2,181)	6.04	6.21

Table 3.4: Comparison of estimation performance between BRNN and MLP (Accuracy %).

Structure	Margin		
	0	1	2
MLP-1	1.46	61.90	68.64
MLP-3	6.12	64.16	70.94
MLP-5	6.20	64.69	71.64
MLP-7	6.19	64.46	71.38
BRNN	8.33	67.93	75.05

(CI) models, context-dependent (CD) models and a phoneme bigram language model were generated for 61 TIMIT phonemes. Left-to-right HMMs with 3 states for each phoneme and 5 Gaussian mixture components per state were trained for the CI models. As for the CD models, shared-state HMMs (600 states in total) with 3 Gaussian mixture components per state were trained [27]. The feature parameters and the training data were the same as for the BRNN conditions.

The results for the context-independent HMMs and the context-dependent HMMs are listed in Table 3.2(a) and Table 3.2(b), respectively. Comparing Table 3.1 with Table 3.2, the proposed method gives considerably higher accuracy than the HMM based approach, especially for $M = 0$ or $M = 1$, even though the BRNN based approach does not use any linguistic knowledge. The reason might be that the BRNNs are trained to accurately detect segment boundaries, while the HMMs are trained based on maximum likelihood criteria.

Comparison between BRNN and MLP

The MLP structure was tested here for three different structures allowing the use of the following four amounts of acoustic context: (1) one frame as input (MLP-1), (2) three frames (middle, left and right) as input (MLP-3), and (3) five frames (middle, two left and two right) as input (MLP-5). (4) seven frames (middle, three left and three right) as

Table 3.5: Comparison of root mean squared error for several MLP structures with 30 hidden units. ($\times 10^{-2}$)

Structure (weights)	Training	Test
MLP-1 (841)	7.05	7.14
MLP-3 (2,401)	6.49	6.63
MLP-5 (3,961)	6.34	6.51
MLP-7 (5,521)	6.28	6.48

Table 3.6: Comparison of estimation performance for several MLP structures with 30 hidden units (Accuracy %).

Structure	Margin		
	0	1	2
MLP-1	0.51	60.98	67.81
MLP-3	6.63	64.50	71.15
MLP-5	7.17	65.78	72.57
MLP-7	6.68	66.40	73.37

input (MLP-7). The structures of these networks were adjusted so each of them had about the same number of free parameters for the BRNN, that is 2,181 (Details are shown in Table 3.3). Here, the number of hidden units for MLP-1, MLP-3, MLP-5 and MLP-7, were 80, 28, 17 and 12, respectively. The feature parameters, the training data, the iterations, the thresholds in method 1 (i.e. h), the activate function etc. were the same as for the BRNN conditions.

Table 3.3 shows the comparison of the root mean squared error (RMSE) between the desired and the estimated outputs for training and test data depending on four amounts of acoustic context. Table 3.4 shows the comparison of estimation performances (Accuracy %) between BRNN and MLP for method 1. We can see from these tables that MLP-5 gives the best performance in terms of RMSE and Accuracy. However, the BRNN outperform MLP-5 for both RMSE and Accuracy. Moreover, it has the advantage that one does not have to choose the optimum number of consecutive frames to define an input window size.

As a reference, the MLP performances for RMSE and Accuracy with 30 hidden units, whose number is equal to those used in the BRNN, are shown in Table 3.5 and Table 3.6, respectively.

By comparing these results to Table 3.3 and Table 3.4, the improvements of the performance are observed for MLP-3, MLP-5 and MLP-7 cases. BRNN, however, gives better performance than any MLP results. From these experimental results, we adopt BRNN as a segment boundary estimator for the purpose of improving the performance of the con-

Table 3.7: Estimation performance for the three kinds of BRNN based methods.

	Method		
	1	2	3
Hit	40,056	48,856	48,856
Deletion	10,262	1,462	1,462
Insertion	2,293	67,570	30,629
Correct	79.61	97.10	97.10
Accuracy	75.05	-37.19	36.22

ventional speech recognition systems, which is described in Section 3.4. Note that MLP is also applicable to the recognition systems in a similar way and it would give useful information for them, since the estimation performance of MLP (Table 3.4) is comparable to that of BRNN compared to the HMM performance (Table 3.2).

Comparison among methods 1, 2 and 3

Estimation results for methods 1, 2 and 3 are shown in Table 3.7. Skip step k in method 3 and M were both set to 2. Method 1 gave the highest accuracy, but there were a large number of deletion errors. This indicates that method 1 would not be appropriate when boundary candidates could be evaluated with other techniques as described in Section 3.4.2. Here, method 3 would be applicable because 97.10% of the correct boundaries remain in the results with smaller insertion errors compared to those of method 2.

3.4 Application to speech recognition system

From the experimental results, we can expect that the BRNN based segment boundary estimator gives useful information for existing speech recognition systems. In this section, we apply the BRNN based segment boundary estimator to two kinds of speech recognition systems in order to achieve better recognition performance or reduce search space. The following two ways are considered for applying the segment boundary information into speech recognition systems.

- Use BRNN output as probability (segment boundaries are not fixed but used "softly")[23]
- Determine segment boundaries explicitly with method $1 \sim 3$ in Section 3.2.4 [28][21][22]

The former method can be incorporated into recognition systems directly and the latter method can be used as pre-processing for recognition systems for the purpose of reducing search space in decoding. In the following sections, the former method is applied into an

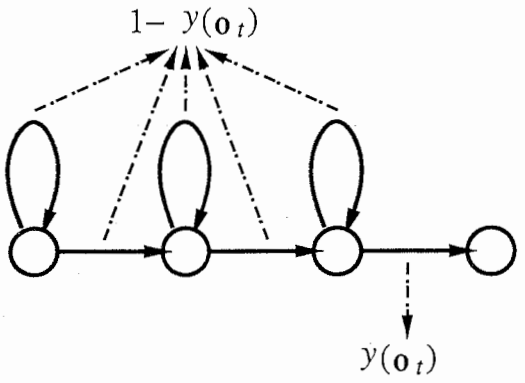


Figure 3.7: Integration of phoneme boundary information into an HMM.

HMM based recognition system and the latter method is applied into a segment model based system. For both systems, we investigate the effectiveness of phoneme boundary information through phoneme recognition experiments using the TIMIT database (462 speakers for training and 168 speakers for testing).

3.4.1 HMM based system

Integration as phoneme boundary probability

If we consider the outputs $y(\mathbf{o}_t)$ $(0 \leq y(\mathbf{o}_t) \leq 1)$ of the BRNN for the observation vector $\mathbf{o}_t$ at time t as the *phoneme boundary probability*, $1 - y(\mathbf{o}_t)$ can be regarded as the *intra-phoneme probability*. These probabilities can be easily incorporated into the transition probabilities of conventional HMMs by taking the product of the two terms to be:

$$\bar{a}_{ij}(\mathbf{o}_t) = a_{ij} \cdot s_{ij}(\mathbf{o}_t), \tag{3.2}$$

where a_{ij} is the transition probabilities from state i to state j and

$$s_{ij}(\mathbf{o}_t) = \begin{cases} y(\mathbf{o}_t), & \text{if } j \text{ is the final state} \\ 1 - y(\mathbf{o}_t) & \text{else.} \end{cases} \tag{3.3}$$

Figure 3.7 shows an example of $s_{ij}(\mathbf{o}_t)$ for a three-state HMM. The time (observation) dependent transition probabilities $\bar{a}_{ij}$ are used in the recognition process.

HMM based recognition experiments

The shared-state context-dependent HMMs described in Section 3.3.1 and a phoneme bigram language model were trained for 61 phonemes. Phoneme recognition was performed using a time-synchronous beam search based decoder [17]. Table 3.8 shows the

Table 3.8: HMM based recognition results. Recognition results evaluated with 39 phoneme sets are shown in brackets.

	without language model		with language model	
	phoneme accuracy (%)	CPU time (%)	phoneme accuracy (%)	CPU time (%)
w/o boundary prob.	50.05 (58.69)	114.1	57.37 (64.71)	476.6
with boundary prob.	53.13 (61.75)	92.5	58.03 (65.47)	332.7
improvement (%)	6.2 (7.4)	18.9	1.5 (2.2)	30.2

recognition results and computational requirements compared to the total time of the utterances. Recognition results evaluated with 39 phoneme sets[29] [1] are shown in brackets. We can see from this table that the BRNN-derived boundary probabilities improve not only recognition performance, but also computational requirements. Note that boundary probabilities can be obtained with small computational requirements: about 0.17 seconds are required for 3.06 second utterances on an HP735 workstation.

3.4.2 Segment model based system

Phoneme segment lattice creation

Recently, a variety of segment models (SMs) have been proposed for relaxing the independence assumption of observation, which is a shortcoming of conventional HMMs. SM based recognition systems, however, generally require much more computation than HMM systems. Therefore, to use SMs in in real-time systems, we have to reduce the computational costs by rescoring N-best candidates or word lattices obtained through HMM based recognition [30][31], or by generating segment lattices with a simple phoneme boundary detector [21]. However, it is not easy to improve performance unless accurate segment boundaries are included in the lattices. The fact that the segmentation probability gave a statistically significant improvement of recognition [23] indicates that accurate boundary estimation in a segment based recognition system is very important.

In method 3 described in Section 3.2.4, a boundary that is detected from the output at t and above threshold h is called a *main boundary*, and a boundary that is a local maximum between a lower threshold $l(< h)$ and h is called a *secondary boundary*. A segment lattice can be created by fully connecting boundaries existing between main boundaries. This lattice is used for phoneme recognition based on polynomial segment models (PSM) [30][32]. Figure 3.8 shows a block diagram of the BRNN-PSM based recognition system.

[1] For example, closure of phone /b/, /d/, /g/, /p/, /t/, /k/ are treated as silence /sil/.

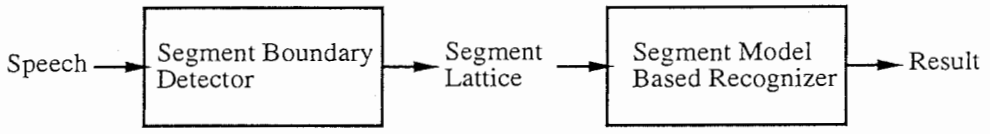


Figure 3.8: Block diagram of the BRNN-PSM based recognition system.

Table 3.9: Computational reduction.

	# of segments	reduction rate
fully connected	6.72×10^7	—
proposed	73,348	1/917

BRNN-PSM based recognition experiments

We generated a context-independent PSM with a single mixture for 61 phonemes. The regression order of the mean trajectories was set to 2. The variance was time invariant throughout a segment. The duration probabilities, which were computed from a histogram of the training segment durations, were used in the recognition. No language model was used in this experiment. Thresholds were set to $h = 0.6$ and $l = 0.25$.

Table 3.9 shows the number of connections in the lattice (i.e. the number of segments to be evaluated) for all test data. "fully connected" indicates all possible connections (i.e. the total number of segments in the case of full search) with durations are more than three frames[2]. Recognition results are listed in Table 3.10. For comparison, results obtained using three-state context-independent HMMs with a single mixture per state are also listed in the table. According to these results, the BRNN-PSM based method achieved both better recognition performance than the HMM system and a considerable computational reduction.

Note that the recognition performance of the BRNN-PSM based method will be improved by using a more precise PSM whose variant is time variance through a segment [32].

3.5 Application to automatic segmentation system

3.5.1 Automatic segmentation

Accurate and automatic segmentation, which generates a time-aligned transcription for speech given a non-aligned transcriptions, is important technique for acoustic model train-

[2] At least three frame segment is required for recognition with the quadratic segment models.

Table 3.10: Recognition results using BRNN based phoneme lattices and polynomial segment models (BRNN-PSM). Recognition results evaluated with 39 phoneme sets are shown in brackets.

	phoneme accuracy (%)
HMM	40.08 (49.64)
BRNN-PSM	41.80 (52.40)

ing for speech recognition [33] or unit selection for speech synthesis [34]. Dynamic programming, which minimizes a distortion between a non-aligned input speech and a time-aligned reference speech [19], or Viterbi segmentation, which maximizes a likelihood for an input speech by using a set of acoustic models [20], have been widely used for automatic segmentation. These methods, however, are not designed so as to estimate the segmentation boundaries accurately but produce the segment boundaries secondarily from the results of minimizing distortion or maximizing likelihood given an observation sequence (e.g. cepstrum). Consequently, we consider that the obtained segmentation boundaries are not accurate enough. In this section, we try to use the segment boundary information obtained from the BRNN, which is trained so as to detect segmentation boundaries accurately, for generating accurate segmentation.

3.5.2 Automatic segmentation using BRNN output

Automatic segmentation with BRNN output can be performed by using the conventional Viterbi algorithm with slight modification of the HMM parameters. As described in Section 3.4.1, BRNN output (i.e. segment boundary information) can be incorporated into the transition probabilities of the HMM by Eq.(3.2).

Now, the acoustic log likelihood $\log \mathrm{P}(\mathbf{o}_t|k)$ for model k given the observation $\mathbf{o}_t$ is obtained by

$$\log \mathrm{P}(\mathbf{o}_t|k) = \log b_j(\mathbf{o}_t) + \log a_{ij} + w \log s_j(\mathbf{o}_t), \tag{3.4}$$

where b_j, a_{ij} and s_j are the output probability, the transition probability and the segment boundary probability, respectively. w is weighting factor for the segment boundary probability obtained from the BRNN output as Eq. (3.3). Note that Eq.(3.4) is equivalent to the conventional log likelihood (i.e. without BRNN output) when $w = 0$.

3.5.3 Experiment

The proposed method was evaluated on TIMIT database.

Table 3.11: Absolute errors for two types of HMM training (msec)(training data / test data)

	Label training	Embedded training
Conventional	11.22 / 11.98	12.19 / 12.64
Proposed	7.78 / 8.48	8.23 / 8.61
Improvement (%)	30.65 / 29.23	32.43 / 31.83

Conditions

As for the phoneme HMMs, shared-state HMM topology (1000 states in total) were generated with the ML-SSS (maximum likelihood successive state splitting) algorithm[16]. This topology was trained with 5 Gaussian mixture components per state using a *label training* (Baum-Welch re-estimation for each label) with 20 iterations. Then, the parameters of the HMMs obtained by label training were re-estimated using an *embedded training* (Baum-Welch re-estimation for each utterance) with 10 iterations. The same feature parameter and the BRNN described in Section 3.3.1 was used.

Results

The absolute errors and the root mean squared errors (RMSE) between the *correct* time-alignments segmented by human experts and the estimates segmented by the Viterbi algorithm are shown in Fig 3.9 and Fig 3.10, respectively. For both figures, the horizontal axis shows the weighting factor w in Eq.(3.4). The results when $w = 0$ indicate the performances without segment boundary probability, that is, the performances of the conventional method. The best performances both for the training and the test sets are obtained at $w = 20$. At this weight, the relative improvements for the test set from the conventional method (i.e. $w = 0$) are 31.83% in absolute error and 16.44% in RMSE.

Table 3.11 shows the differences of the absolute errors for the label training and the embedded training. In this table, "Conventional" indicates the results when $w = 0$ and "Proposed" indicates the results which give the best performance on the training set ($w = 10$ for the label training, $w = 20$ for the embedded training). We can see from this table that the better segmentation boundaries can be obtained by using the HMMs trained with the label training rather than the HMMs with the additional embedded training. Again, the proposed method achieved about 30% relative improvements in all cases.

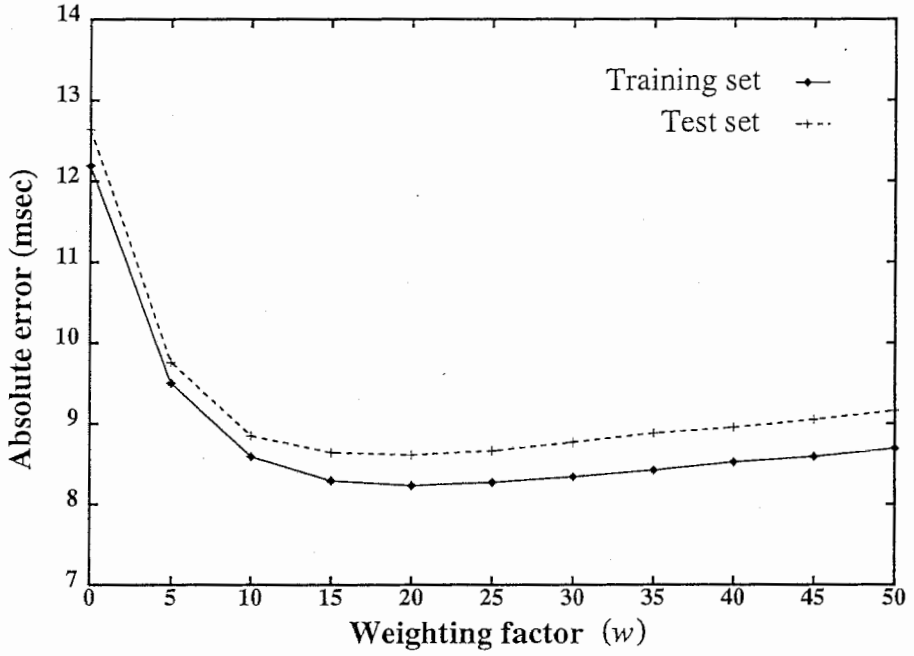


Figure 3.9: Absolute error as a function of the weighting factor w.

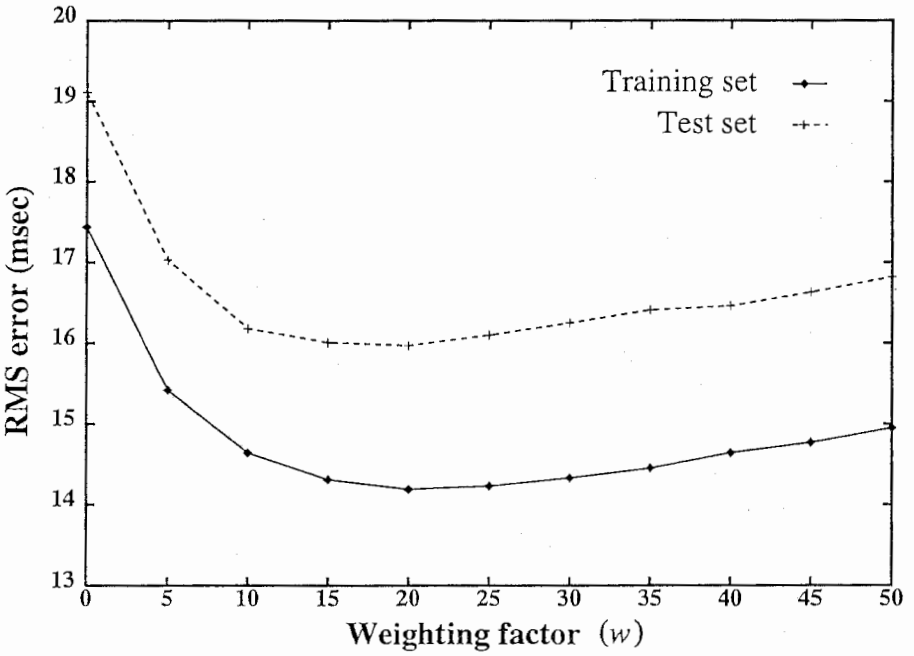


Figure 3.10: Root mean squared error as a function of the weighting factor w.

3.6 Conclusions

A segment boundary estimation method based on a BRNN has been proposed. The proposed method can accurately estimate segment boundaries by only using time series

feature parameters. We applied this method to a speech recognition system and showed that (1) the usage of BRNN outputs was effective for improving the recognition rate and reducing computational time in an HMM based recognition system and (2) segment lattices obtained by the proposed methods dramatically reduce the computational complexity of segment model based recognition.

In Section 3.5, the application of the segment boundary information obtained from BRNN to automatic segmentation has been described. The proposed method achieved 30 % error reduction in absolute error compared to the conventional segmentation method. The improvement of the speech recognition performance, the auditory effectiveness in the speech synthesis and the comparison with the automatic segmentation using speaker adaptation technique[33] are remaining issues.

Chapter 4

Segmental acoustic modeling

In this chapter, we propose parameter estimation techniques for mixture density polynomial segment models (MDPSM) where their trajectories are specified with an arbitrary regression order. MDPSM parameters can be trained in one of three different ways : (1) segment clustering; (2) expectation maximization (EM) training of mean trajectories; and (3) EM training of mean and variance trajectories. These parameter estimation methods were evaluated in TIMIT vowel classification experiments. The experimental results showed that modeling both the mean and variance trajectories is consistently superior to modeling only the mean trajectory. We also found that modeling both trajectories results in significant improvements over the conventional HMM.

4.1 Segment model

To date, one of the most successful approaches for large vocabulary continuous speech recognition has been based on the hidden Markov model (HMM). Although HMMs will continue to play an important role in most recognition systems for a long time to come, many alternative models have been proposed in recent years that enable some of the shortcomings of HMMs to be addressed. Broadly speaking, there are two HMM limitations that various models have tried to address: (1) weak duration modeling and (2) assumption of the conditional independence of observations given the state sequence. The first problem, where an HMM state duration model is implicitly given by a geometric distribution, has been addressed by introducing semi-Markov models with explicit state duration distributions. The second problem has been widely acknowledged to be more serious, and a number of alternative solutions that address this problem have been studied [35] [36] [37] [30] [38] [39] [40] [41] [42] [43]. Delta parameters offer the simplest way of representing the time dependency of observations, and have been shown to tremendously boost performance. Other alternatives are more elegant in representing the time dependency. The polynomial segment modeling proposed by Gish and Ng [30] is one such technique for relaxing the

independence assumption. This modeling technique, however, has a serious shortcoming; it assumes the variance to be time invariant within a segment. This will be disadvantageous with respect to the conventional HMMs which can represent variance changes in a segment by dividing the segment into a number of states with different variances.

In the remainder of this chapter, we consider the case where both mean and covariance are varying with time. We present a model parameter estimation method for mixture density polynomial segment models (MDPSMs) to deal with this type of time-varying case. The model parameters of the MDPSM are the mean trajectory coefficients, the covariance coefficients and the mixture weights. In our segmental modeling approach, higher order regression models are used not only for mean trajectory modeling but also for time-varying covariance modeling. The paper proposed by Gish and Ng [30] can be viewed as a special case (i.e. 0-th order regression for modeling the covariance coefficients) if our method is considered. Recently, a similar approach was also proposed by Gish and Ng [44]. However, they restricted the time variation of the covariance coefficients to be limited to having three different covariance matrices existing over a segment, while there is no restriction of this type in our modeling.

First, we starts with an overview of single Gaussian segment modeling, goes on to describe two methods of model parameter estimation for MDPSM with time-invariant variance, and finally provides model parameter estimation formulation for the time-variant variance case[1]. To confirm the performance of the three kinds of MDPSM, preliminary classification experiments are performed.

4.2 Polynomial segment models

Consider an L (in frames) length sequence of observation vectors $\{\mathbf{y}_1, \ldots, \mathbf{y}_L\}$, where $\mathbf{y}_t = [y_{t,1}, y_{t,2}, \ldots, y_{t,D}]$ is a D-dimensional observation (e.g. cepstrum) vector at time t. This sequence defines a segment which can be expressed in the form of an $L \times D$ matrix

$$\mathbf{Y} = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{1,D} \\ y_{2,1} & y_{2,2} & \cdots & y_{2,D} \\ \vdots & \vdots & & \vdots \\ y_{L,1} & y_{L,2} & \cdots & y_{L,D} \end{bmatrix}. \tag{4.1}$$

In the polynomial segment model, this segment is represented by an R-th order trajectory model as follows:

$$\mathbf{Y} = \mathbf{Z}\mathbf{B} + \mathbf{E}, \tag{4.2}$$

[1] In this chapter, we provide the MDPSM formulation for diagonal covariance matrices only. However, it can be easily extended to the full covariance case.

where $\mathbf{Z}$ is an $L \times (R+1)$ design matrix defined by

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & \frac{1}{L-1} & \dots & \left(\frac{1}{L-1}\right)^R \\ \vdots & \vdots & & \vdots \\ 1 & \frac{t-1}{L-1} & \dots & \left(\frac{t-1}{L-1}\right)^R \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix}, \tag{4.3}$$

$\mathbf{B}$ is an $(R+1) \times D$ trajectory parameter matrix

$$\mathbf{B} = \begin{bmatrix} b_{0,1} & b_{0,2} & \dots & b_{0,D} \\ b_{1,1} & b_{1,2} & \dots & b_{1,D} \\ \vdots & \vdots & & \vdots \\ b_{R,1} & b_{R,2} & \dots & b_{R,D} \end{bmatrix}, \tag{4.4}$$

and $\mathbf{E}$ is an $L \times D$ residual error matrix

$$\mathbf{E} = \begin{bmatrix} e_{1,1} & e_{1,2} & \dots & e_{1,D} \\ e_{2,1} & e_{2,2} & \dots & e_{2,D} \\ \vdots & \vdots & & \vdots \\ e_{L,1} & e_{L,2} & \dots & e_{L,D} \end{bmatrix}. \tag{4.5}$$

Design matrix $\mathbf{Z}$ deals with normalizing different length of segments uniformly between times 0 and 1.

4.2.1 Single Gaussian segment model

The likelihood of the segment $\mathbf{Y}$ given that it is generated by a label a can be expressed as

$$\mathrm{P}(\mathbf{Y}|a) = \prod_{t=1}^{L} f(\mathbf{y}_t). \tag{4.6}$$

In this equation, $f(\mathbf{y}_t)$ is the likelihood of the feature vector $\mathbf{y}_t$ conditioned on the label a and is given by

$$f(\mathbf{y}_t) = \frac{1}{(2\pi)^{\frac{D}{2}} \mid \Sigma_a \mid^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{y}_t - \mathbf{z}_t\mathbf{B}_a)^T \Sigma_a^{-1}(\mathbf{y}_t - \mathbf{z}_t\mathbf{B}_a)\right\}, \tag{4.7}$$

where $\mathbf{B}_a$ and Σ_a are the parameters of the single Gaussian segment model describing the label a. In Eq. (4.7), the vector $\mathbf{z}_t$ is given by

$$\mathbf{z}_t = \left[1, \frac{t-1}{L-1}, \dots, \left(\frac{t-1}{L-1}\right)^R\right]. \tag{4.8}$$

Assuming that we are given K segments $\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K$ in the training data for label a, we want to compute model parameters $\mathbf{B}_a$ and $\mathbf{\Sigma}_a$ for single Gaussian segment model. Probability of these segments given $\mathbf{B}_a$ and $\mathbf{\Sigma}_a$ is given as

$$\begin{aligned} \mathrm{P}(\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_K | \mathbf{B}_a, \mathbf{\Sigma}_a) &= \prod_{k=1}^{K} \mathrm{P}(\mathbf{Y}_k | \mathbf{B}_a, \mathbf{\Sigma}_a) \\ &= \prod_{k=1}^{K} \prod_{t=1}^{L} f(\mathbf{y}_{k,t}). \end{aligned} \tag{4.9}$$

These model parameters can be obtained by maximizing this probability with respect to $\mathbf{B}_a$ and $\mathbf{\Sigma}_a$. Their estimates can be computed as follows:

$$\bar{\mathbf{B}}_a = \left[\sum_{k=1}^{K} \mathbf{Z}_k^T \mathbf{Z}_k \right]^{-1} \left[\sum_{k=1}^{K} \mathbf{Z}_k^T \mathbf{Y}_k \right], \tag{4.10}$$

$$\bar{\mathbf{\Sigma}}_a = \frac{\sum_{k=1}^{K} (\mathbf{Y}_k - \mathbf{Z}_k \bar{\mathbf{B}}_a)^T (\mathbf{Y}_k - \mathbf{Z}_k \bar{\mathbf{B}}_a)}{\sum_{k=1}^{K} L_k}. \tag{4.11}$$

From now on, we omit the subscript a from the model parameters $\mathbf{B}_a$ and $\mathbf{\Sigma}_a$ for simplification.

4.2.2 Mixture density polynomial segment model

The discussion in the previous section was concerned with single Gaussian segment modeling. In this section, we extend it to a mixture density case. In this case, the likelihood $f(\mathbf{y}_t)$ is represented by a mixture of M Gaussians; i.e.

$$f(\mathbf{y}_t) = \sum_{m=1}^{M} w_m \, f_m(\mathbf{y}_t), \tag{4.12}$$

where

$$f_m(\mathbf{y}_t) = \frac{1}{(2\pi)^{\frac{D}{2}} \mid \mathbf{\Sigma}_m \mid^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2} (\mathbf{y}_t - \mathbf{z}_t \mathbf{B}_m)^T \mathbf{\Sigma}_m^{-1} (\mathbf{y}_t - \mathbf{z}_t \mathbf{B}_m) \right\}, \tag{4.13}$$

and w_m is the weight of the m-th mixture component. The mixture components satisfy the relation $\sum_{m=1}^{M} w_m = 1$. The model parameters $\mathbf{B}_m$, $\mathbf{\Sigma}_m$, and w_m in Eq.(4.12) can be estimated by segment clustering or by EM training. These methods are described in detail in Section 4.2.2 and Section 4.2.2, respectively. These methods are developed here under the assumption that the covariance matrices $\{\mathbf{\Sigma}_m, m = 1, 2, ..., M\}$ for the M mixture components are diagonal; i.e.

$$\mathbf{\Sigma}_m = \mathrm{diag}[c_{m,1}, c_{m,2}, \ldots, c_{m,D}]$$

$$= \begin{bmatrix} c_{m,1} & 0 & \dots & 0 \\ 0 & c_{m,2} & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & c_{m,D} \end{bmatrix}. \tag{4.14}$$

Under this assumption, Eq.(4.13) becomes

$$f_m(\mathbf{y}_t) = \frac{1}{(2\pi)^{\frac{D}{2}}(\prod_{d=1}^{D} c_{m,d})^{\frac{1}{2}}} \exp\left\{-\sum_{d=1}^{D} \frac{(y_{t,d} - \mathbf{z}_t \mathbf{b}_{m,d})^2}{2c_{m,d}}\right\}, \tag{4.15}$$

where $\bar{\mathbf{b}}_{m,d} = [\bar{b}_{m,0,d}, \bar{b}_{m,1,d}, \dots, \bar{b}_{m,R,d}]^T$ is the d-th dimensional mean trajectory parameters of the m-th mixture. We describe below different methods for estimating model parameters $\mathbf{B}_m$, Σ_m, and w_m under the assumption that Σ_m is diagonal. However, these methods can be easily extended to the full-covariance case.

Clustering method

One simple way of estimating MDPSM parameters $\mathbf{B}_m, \Sigma_m$, and w_m is based on segment clustering. That is, the training segments $\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_K$ for the label a are partitioned into M regions using the K-means clustering algorithm. The K-means clustering algorithm requires a definition of distance between a segment $\mathbf{Y}$ and cluster m. The distance measure used here during the clustering is a "multivariate Gaussian distance measure":

$$\text{Dist}(\mathbf{Y}, m) = \frac{1}{2}LD\log 2\pi + \frac{1}{2}L\sum_{d=1}^{D}\log c_{m,d} + \frac{1}{2}\sum_{t=1}^{L}\sum_{d=1}^{D}\frac{(y_{t,d} - \mathbf{z}_t\mathbf{b}_{m,d})^2}{c_{m,d}}. \tag{4.16}$$

Estimates of $\mathbf{B}_m$ and Σ_m can be obtained in the same way as in the single mixture case using the segments assigned to cluster m. w_m is calculated as the relative frequency of the segments:

$$w_m = \frac{N_m}{\sum_{j=1}^{M} N_j}, \quad m = 1, \dots, M, \tag{4.17}$$

where N_m is the number of segments for cluster m.

EM method

Let $\mathbf{Y}_1^K = \{\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_K\}$ be a set of training segments belonging to the label a. Our aim here is to estimate the model parameters $\mathbf{B}_m, \Sigma_m$, and w_m from these training segments using the expectation-maximization (EM) algorithm. For this, we derive re-estimation formulas by maximizing $\text{P}(\mathbf{Y}_1^K|a)$ based on the EM algorithm. This can be

done by maximizing the following auxiliary function Q with the mixture components being the hidden variables:

$$Q(\bar{\Phi}|\Phi) = E\left[\log \mathrm{P}(\mathbf{Y}_1^K, m|\bar{\Phi})|\mathbf{Y}_1^K, \Phi\right] = \sum_{m=1}^{M} \frac{\mathrm{P}(\mathbf{Y}_1^K, m|\Phi)}{\mathrm{P}(\mathbf{Y}_1^K|\Phi)} \log \mathrm{P}(\mathbf{Y}_1^K, m|\bar{\Phi}), \tag{4.18}$$

where Φ and $\bar{\Phi}$ are the sets of the current model parameters and the re-estimated model parameters, respectively. m denotes the index of a mixture component. Since $\log \mathrm{P}(\mathbf{Y}_1^K, m|\bar{\Phi})$ in Eq.(4.18) can be rewritten as

$$\log \mathrm{P}(\mathbf{Y}_1^K, m|\bar{\Phi}) = \log \mathrm{P}(\mathbf{Y}_1^K|m, \bar{\Phi}) + \log \mathrm{P}(m|\bar{\Phi}), \tag{4.19}$$

maximizing Eq.(4.18) is equivalent to individually maximizing the following two functions:

$$Q_1(\bar{\Phi}|\Phi) = \sum_{m=1}^{M} \frac{\mathrm{P}(\mathbf{Y}_1^K, m|\Phi)}{\mathrm{P}(\mathbf{Y}_1^K|\Phi)} \log \mathrm{P}(\mathbf{Y}_1^K|m, \bar{\Phi}), \tag{4.20}$$

with respect to $\mathbf{B}_m$ and Σ_m, and

$$Q_2(\bar{\Phi}|\Phi) = \sum_{m=1}^{M} \frac{\mathrm{P}(\mathbf{Y}_1^K, m|\Phi)}{\mathrm{P}(\mathbf{Y}_1^K|\Phi)} \log \mathrm{P}(m|\bar{\Phi}), \tag{4.21}$$

with respect to w_m.

Let the probability $\mathrm{P}(\mathbf{Y}_1^K, m|\Phi)/\mathrm{P}(\mathbf{Y}_1^K|\Phi)$ in Eq.(4.20) and Eq.(4.21) be denoted as $\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}$ using the current model parameters Φ. Then, we can estimate $\gamma_{k,m,t}$ efficiently as

$$\gamma_{k,m,t} = \begin{cases} \dfrac{\alpha_{k,t}\beta_{k,t+1}w_m f_m(\mathbf{y}_{k,t+1})}{\mathrm{P}(\mathbf{Y}_1^K|\Phi)}, & t = 1, \ldots, L_k - 1 \\ \dfrac{\alpha_{k,T}}{\mathrm{P}(\mathbf{Y}_1^K|\Phi)}, & t = L_k, \end{cases} \tag{4.22}$$

where $\alpha_{k,t}$ and $\beta_{k,t}$ are obtained recursively:

$$\alpha_{k,t} = \begin{cases} f(\mathbf{y}_{k,1}), & t = 1 \\ \alpha_{k,t-1}f(\mathbf{y}_{k,t}), & t = 2, \ldots, L_k, \end{cases} \tag{4.23}$$

$$\beta_{k,t} = \begin{cases} 1, & t = L_k \\ \beta_{k,t+1}f(\mathbf{y}_{k,t+1}), & t = L_k - 1, \ldots, 1. \end{cases} \tag{4.24}$$

First, we consider obtaining the d-th dimensional mean trajectory parameters of the m-th mixture, $\bar{\mathbf{b}}_{m,d} = [\bar{b}_{m,0,d}, \bar{b}_{m,1,d}, \ldots, \bar{b}_{m,R,d}]^T$. These parameters can be obtained through differentiation of Eq.(4.20) with respect to $\bar{b}_{m,r,d}$ and solving the equation:

$$\frac{\partial Q_1}{\partial \bar{b}_{m,r,d}} = \sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\partial \log \bar{f}_m(\mathbf{y}_{k,t})}{\partial \bar{b}_{m,r,d}} = 0, \quad r = 0, \ldots, R. \tag{4.25}$$

From Eq.(4.15),

$$\frac{\partial \log \bar{f}_m(\mathbf{y}_{k,t})}{\partial \bar{b}_{m,r,d}} = \frac{(y_{k,t,d} - \mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})}{\bar{c}_{m,d}} \left(\frac{t-1}{L_k - 1}\right)^r. \tag{4.26}$$

Substituting this equation into Eq.(4.25) and noting that $\bar{c}_{m,d}$ is an independent constant of time t, we obtain

$$\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\left\{y_{k,t,d}-\sum_{u=0}^{R}\bar{b}_{m,u,d}\left(\frac{t-1}{L_k-1}\right)^u\right\}\left(\frac{t-1}{L_k-1}\right)^r=0,\quad r=0,\ldots,R. \tag{4.27}$$

Eq.(4.27) can be rewritten as

$$\sum_{u=0}^{R} g_m(u+r)\bar{b}_{m,u,d}=v_{m,d}(r),\quad r=0,\ldots,R, \tag{4.28}$$

where

$$g_m(l) = \sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\left(\frac{t-1}{L_k-1}\right)^l,\quad l=0,\ldots,2R, \tag{4.29}$$

and

$$v_{m,d}(r) = \sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\,y_{k,t,d}\left(\frac{t-1}{L_k-1}\right)^r,\quad r=0,\ldots,R. \tag{4.30}$$

Note that $g_m(l)$ is independent of the dimension d of the feature parameter and $v_{m,d}(r)$ is dependent of dimension d. The $(R+1)$ parameters, $\{\bar{b}_{m,u,d}, u=0,\ldots,R\}$, can be obtained by solving the set of $(R+1)$ simultaneous linear equations given by Eq.(4.28). These equations can be written in a matrix form as follows:

$$\begin{bmatrix} g_m(0) & g_m(1) & \cdots & g_m(R) \\ g_m(1) & g_m(2) & \cdots & g_m(R+1) \\ \vdots & \vdots & \ddots & \vdots \\ g_m(R) & g_m(R+1) & \cdots & g_m(2R) \end{bmatrix}\begin{bmatrix} \bar{b}_{m,0,d} \\ \bar{b}_{m,1,d} \\ \vdots \\ \bar{b}_{m,R,d} \end{bmatrix}=\begin{bmatrix} v_{m,d}(0) \\ v_{m,d}(1) \\ \vdots \\ v_{m,d}(R) \end{bmatrix}. \tag{4.31}$$

In order to compute the d-th diagonal component of the covariance matrix $\bar{\Sigma}_m$, we differentiate Eq.(4.20) with respect to the $\bar{c}_{m,d}$ and solve the following equation:

$$\frac{\partial Q_1}{\partial \bar{c}_{m,d}}=\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\frac{\partial \log \bar{f}_m(\mathbf{y}_{k,t})}{\partial \bar{c}_{m,d}}=0. \tag{4.32}$$

From Eq.(4.15), we can derive

$$\frac{\partial \log \bar{f}_m(\mathbf{y}_{k,t})}{\partial \bar{c}_{m,d}}=-\frac{1}{2\bar{c}_{m,d}}+\frac{(y_{k,t,d}-\mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2}{2\bar{c}_{m,d}^2}. \tag{4.33}$$

Using Eq.(4.33), Eq.(4.32) can be rewritten as

$$\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\left\{\bar{c}_{m,d}-(y_{k,t,d}-\mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2\right\}=0. \tag{4.34}$$

Then, $\bar{c}_{m,d}$ can be obtained by

$$\bar{c}_{m,d}=\frac{\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}\,(y_{k,t,d}-\mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2}{\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}}. \tag{4.35}$$

The weighting coefficient $\bar{w}_m$ can be obtained from Equation (4.21) by application of a Lagrange optimization using Lagrange multipliers:

$$\bar{w}_m = \frac{\sum_{k=1}^{K}\sum_{t=1}^{L_k}\gamma_{k,m,t}}{\sum_{k=1}^{K}\sum_{t=1}^{L_k}\sum_{j=1}^{M}\gamma_{k,j,t}}. \tag{4.36}$$

The results of the model parameters obtained from the clustering method described in Section 4.2.2 can be used as initial model parameters for the EM algorithm.

4.2.3 Variance trajectory model

In the previous subsection, we have discussed the mixture density polynomial segment model (MDPSM), where the mean feature vector of the m-th mixture is varying with time, but the covariance matrix Σ_m remains time invariant. In this subsection, we modify the segment model so that both the mean vector and the covariance matrix can vary with time within a segment. We believe that this will allow more precise modeling. In order to characterize the time-varying behavior of the mean vector and the covariance matrix, we represent their trajectories by a polynomial segment model. In the case of the covariance matrix $\Sigma_{m,t} = \mathrm{diag}[c_{m,t,1}, \ldots, c_{m,t,D}]$, this is represented as an R-th order polynomial:

$$\mathbf{c}_{m,t} = \mathbf{z}_t \mathbf{S}_m, \tag{4.37}$$

where

$$\mathbf{c}_{m,t} = [c_{m,t,1}, \ldots, c_{m,t,D}], \tag{4.38}$$

$$\mathbf{z}_t = \left[1, \frac{t-1}{L-1}, \ldots, \left(\frac{t-1}{L-1}\right)^R\right], \tag{4.39}$$

and

$$\mathbf{S}_m = \begin{bmatrix} s_{m,0,1} & s_{m,0,2} & \cdots & s_{m,0,D} \\ s_{m,1,1} & s_{m,1,2} & \cdots & s_{m,1,D} \\ \vdots & \vdots & & \vdots \\ s_{m,R,1} & s_{m,R,2} & \cdots & s_{m,R,D} \end{bmatrix}. \tag{4.40}$$

Note that we have represented the trajectories of the mean vector and the covariance matrix by the polynomial segment models of the same order R, though they can, in principle, be different. With this model, the likelihood of the vector $\mathbf{y}_t$ is given by

$$f_m(\mathbf{y}_t) = \frac{1}{(2\pi)^{\frac{D}{2}}\left(\prod_{d=1}^{D} c_{m,t,d}\right)^{\frac{1}{2}}} \exp\left\{-\sum_{d=1}^{D}\frac{(y_{t,d} - \mathbf{z}_t \mathbf{b}_{m,d})^2}{2c_{m,t,d}}\right\}. \tag{4.41}$$

In this model, estimates of the mean trajectory and weight parameters can be obtained in a way similar to that described in 4.2.2, except that $\bar{c}_{m,d}$ in Equation (4.26) is replaced

by $\sum_{n=0}^{R} s_{m,n,d} \left(\frac{t-1}{L_k-1}\right)^n$ to reflect its time dependence. The computation of the mean trajectory and weight parameters can be obtained from Eq.(4.31) and Eq.(4.36), respectively. Note that $g_m(l)$ and $v_{m,d}(r)$ in Equation (4.31) and $\gamma_{k,m,t}$ in Eq.(4.36) are different from the time-invariant case, because the likelihood is computed by Eq.(4.41) instead of Equation (4.15). The computation of variance differs as follows. The ML estimates of the d-th diagonal component of the covariance matrix $\bar{\Sigma}_m$ can be obtained through differentiation of Eq.(4.20) with respect to $\bar{s}_{m,r,d}$ and solving the equation:

$$\frac{\partial Q_1}{\partial \bar{s}_{m,r,d}} = \sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\partial \log \bar{f}_m(\mathbf{y}_{k,t})}{\partial \bar{s}_{m,r,d}} = 0. \tag{4.42}$$

It gives

$$\sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\left(\frac{t-1}{L_k-1}\right)^r}{\left\{\sum_{n=0}^{R} \bar{s}_{m,n,d}\left(\frac{t-1}{L_k-1}\right)^n\right\}^2} \left\{\sum_{u=0}^{R} \bar{s}_{m,u,d}\left(\frac{t-1}{L_k-1}\right)^u - (y_{k,t,d} - \mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2\right\} = 0. \tag{4.43}$$

This is a non-linear equation in $\bar{s}_{m,n,d}$. In order to make it linear, we use an approximation assuming $\bar{s}_{m,n,d}$ in the denominator is replaced by the current value, $s_{m,n,d}$, as

$$\sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\left(\frac{t-1}{L_k-1}\right)^r}{\left\{\sum_{n=0}^{R} s_{m,n,d}\left(\frac{t-1}{L_k-1}\right)^n\right\}^2} \left\{\sum_{u=0}^{R} \bar{s}_{m,u,d}\left(\frac{t-1}{L_k-1}\right)^u - (y_{k,t,d} - \mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2\right\} = 0. \tag{4.44}$$

Now Eq.(4.44) can be rewritten as

$$\sum_{u=0}^{R} h_{m,d}(u+r)\ \bar{s}_{m,u,d} = x_{m,d}(r), \quad r = 0, \ldots, R, \tag{4.45}$$

where

$$h_{m,d}(l) = \sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\left(\frac{t-1}{L_k-1}\right)^l}{\left\{\sum_{n=0}^{R} s_{m,n,d}\left(\frac{t-1}{L_k-1}\right)^n\right\}^2}, \quad l = 0, \ldots, 2R, \tag{4.46}$$

and

$$x_{m,d}(r) = \sum_{k=1}^{K}\sum_{t=1}^{L_k} \gamma_{k,m,t} \frac{\left(\frac{t-1}{L_k-1}\right)^r (y_{k,t,d} - \mathbf{z}_{k,t}\bar{\mathbf{b}}_{m,d})^2}{\left\{\sum_{n=0}^{R} s_{m,n,d}\left(\frac{t-1}{L_k-1}\right)^n\right\}^2}, \quad r = 0, \ldots, R. \tag{4.47}$$

The $(R+1)$ parameters, $\{\bar{s}_{m,u,d}, u = 0, \ldots, R\}$, can be obtained by solving the set of $(R+1)$ simultaneous linear equations given by Eq.(4.45). These equations can be written

in a matrix form as follows:

$$\begin{bmatrix} h_{m,d}(0) & h_{m,d}(1) & \dots & h_{m,d}(R) \\ h_{m,d}(1) & h_{m,d}(2) & \dots & h_{m,d}(R+1) \\ \vdots & \vdots & \ddots & \vdots \\ h_{m,d}(R) & h_{m,d}(R+1) & \dots & h_{m,d}(2R) \end{bmatrix} \begin{bmatrix} \bar{s}_{m,0,d} \\ \bar{s}_{m,1,d} \\ \vdots \\ \bar{s}_{m,R,d} \end{bmatrix} = \begin{bmatrix} x_{m,d}(0) \\ x_{m,d}(1) \\ \vdots \\ x_{m,d}(R) \end{bmatrix}. \quad (4.48)$$

Note that both $h_{m,d}(l)$ and $x_{m,d}(r)$ are dependent on dimension in this equation.

4.3 Evaluation of variance trajectory models

4.3.1 Conditions

To investigate the relative effectiveness of the three kinds of MDPSM, we perform experiments on a speaker-independent 16-vowel classification task using the TIMIT corpus. Sixteen vowels include 13 monothongs /aa, ae, ah, ao, eh, er, ey, ih, iy, ow, uh, uw, ux/ and three diphthongs /aw, ay, oy/. A total of 462 speakers (41,014 tokens) are employed for context-independent MDPSM training and 168 speakers (14,981 tokens) are employed for testing. The regression order of the mean trajectories and the time-varying variance trajectories are set to 2. We generate MDPSMs with diagonal covariance matrices from 10-dimensional Mel-Frequency Cepstral Coefficients (MFCCs) and their derivatives with a 5 ms frame rate. The experimental conditions are listed in Table 4.1. As for the initial variances for the variance trajectory model, the estimates obtained from the EM method as described in Section 4.2.2 are used. That is, $s_{m,1,d}$ and $s_{m,2,d}$ in Equation(4.40) are set to zero for the initial values. Segment $\mathbf{Y}$ is classified as phoneme $\hat{m}$ if

$$\hat{m} = \underset{m}{\text{argmax}} \log \mathrm{P}(\mathbf{Y}|m). \quad (4.49)$$

4.3.2 Effectiveness of variance trajectory modeling

Figure 4.1 shows the differences between the conventional constant variance MDPSM (Figure 4.1(a)) and the variance trajectory model (Figure 4.1(b)). These trajectories are obtained from the model parameters estimated for the /ay/ vowel segments with single mixture. The solid lines show the trajectories μ_t of the first and the second MFCC values. The dotted lines show the trajectories $\bar{\mu}_t$ calculated as:

$$\bar{\mu}_t = \mu_t \pm \sigma_t. \quad (4.50)$$

where σ_t represents the standard derivation. Note that σ_t is constant throughout the segment for Figure 4.1(a) and σ_t is time-variant for Figure 4.1(b). In general, variances of central parts of vowel segments are smaller than those of the beginning or the ending parts

Table 4.1: Experimental conditions.

Analysis	
Sampling frequency	16 kHz
Preemphasis	$1 - 0.98\ z^{-1}$
Frame length	25.6 msec (Hamming window)
Frame period	5.0 msec
Feature vector	10-order MFCC + 10-order Δ MFCC
Training	
Number of speakers	462
Number of tokens	41,014
MDPSM	Context independent models with diagonal covariance matrices
Regression order	2
EM iterations	20
Test	
Number of speakers	168
Number of tokens	14,981

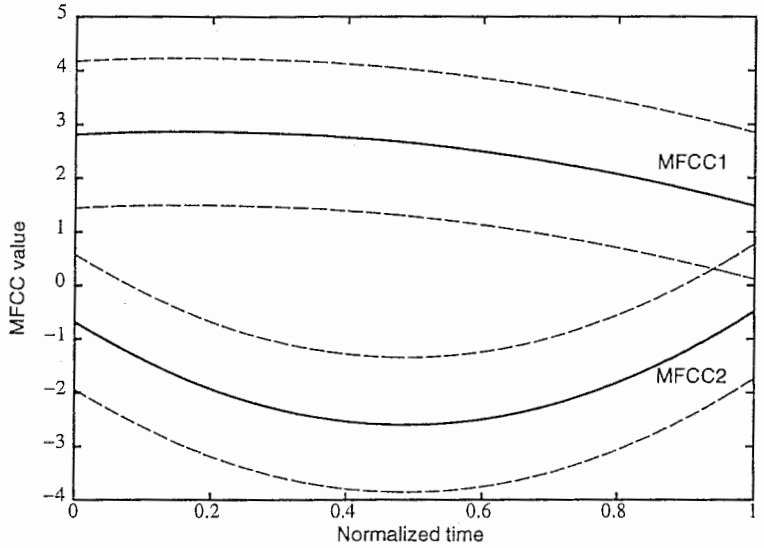
of them. We can see from these figures that the variance trajectory model can capture these phenomena.

Figure 4.2 shows log likelihood as a function of iterations on the /aa/ vowel segments (3,054 segments in total). The solid line shows the log likelihood for three component MDPSMs with constant variance as described in Section 4.2.2. The dotted line represents the for three component MDPSMs with the variance trajectory model (VTM). We can see from this figure that the VTM gives higher log likelihood than the constant variance model at more than five iterations.

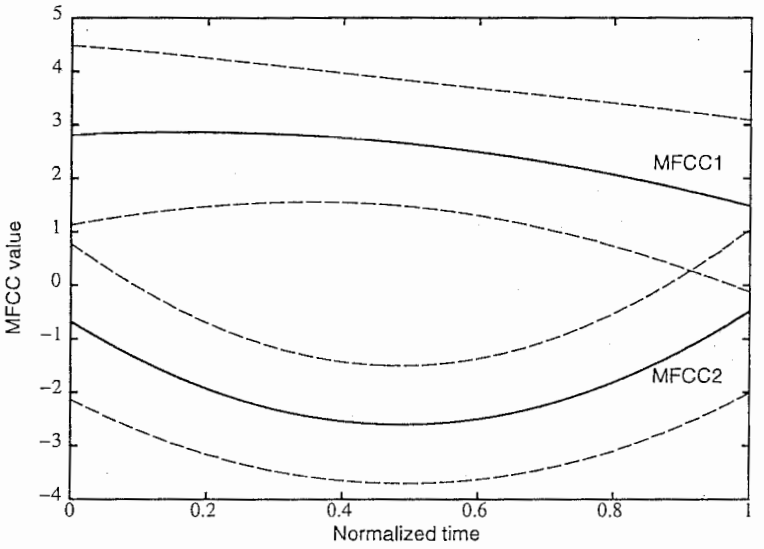
4.3.3 Classification results

Baseline performances

The classification results based on Equation(4.49) are shown in Table 4.2. In this table, clustering, EM and VTM stand for the MDPSMs described in Section 4.2.2, Section 4.2.2 and Section 4.2.3, respectively. Note that the clustering and EM methods for single mixture give the same performances. Duration probabilities are not considered. From these results, we can say that the variance trajectory model consistently outperformed clustering and EM-based models. It indicates that time-variant modeling of variances is effective for improving the classification performances.



(a) Conventional PSM (variance is constant)



(b) Variance trajectory model

Figure 4.1: Comparison between (a) constant variance model and (b) variance trajectory model.

Utilization of duration probabilities

In general, duration probabilities provide useful information for speech recognition. Therefore, we investigate here the use of the duration probabilities for improving speech recognition performance. These probabilities are computed from a histogram of the training segment durations.

In order to match the dynamic ranges of $\log \mathrm{P}(\mathrm{Y}|m)$ and $\log \mathrm{P}(L|m)$, here we use the

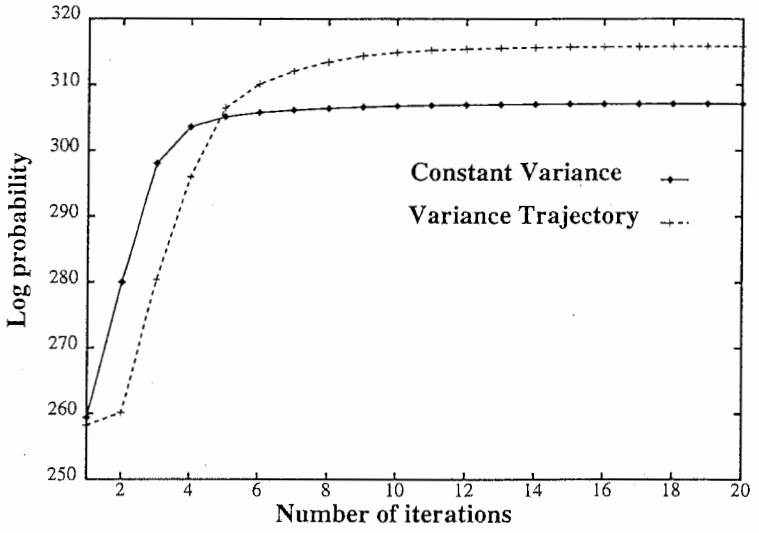


Figure 4.2: Log likelihood as function of iterations on training data (vowel /aa/).

Table 4.2: Classification rate (%)(without duration probability).

Method	Number of mixtures				
	1	3	5	7	9
Clustering	54.3	56.2	58.9	59.5	60.1
EM	54.3	59.6	61.4	63.0	63.2
VTM	56.7	60.7	62.6	63.6	63.8

following two approaches. In the first approach, we use a fixed (or, static) weighting and perform classification using the following equation:

$$\hat{m} = \underset{m}{\operatorname{argmax}} \{\log \mathrm{P}(\mathbf{Y}|m) + \alpha \log \mathrm{P}(L|m)\}, \tag{4.51}$$

where $\mathrm{P}(L|m)$ is L-frame duration probability given phoneme m and α is a static weighting factor. In order to obtain the optimal value of the static weighting factor α, we conduct classification experiments using the training data for training as well as for testing. In these experiments, three mixture variance trajectory models are used. Figure 4.3 shows the classification performance as a function of α. We have computed the average segment length from all the segments in the training data and found it to be 21.99 frames. In Figure 4.3, we have put a vertical dotted line at $\alpha = 21.99$. We can see from this figure that the classification performance initially increases with α, attains a maximum at $\alpha = 20$, and then starts decreasing. Thus, the best classification performance occurs when the weighting factor is approximately equal to the average segment length.

In the second approach, we use a variable (or, dynamic) weighting for matching the dynamic ranges of $\log \mathrm{P}(\mathbf{Y}|m)$ and $\log \mathrm{P}(L|m)$ and perform classification using the following

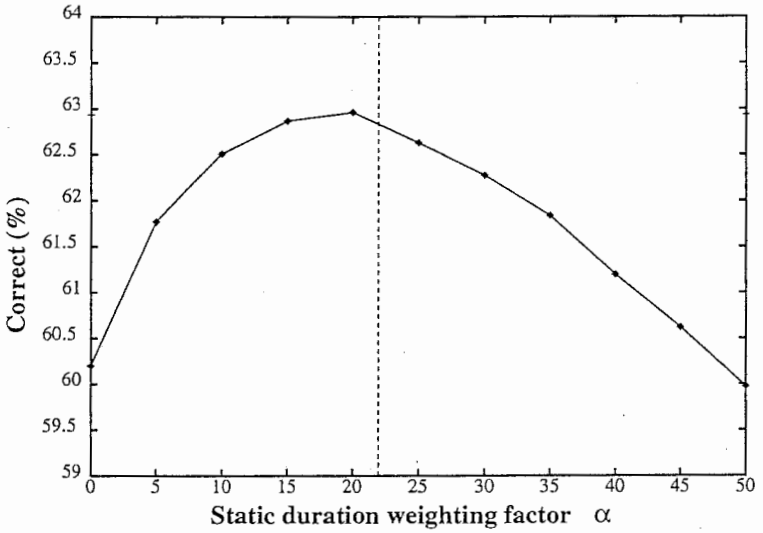


Figure 4.3: Classification performance as a function of static weighting factor α of duration probability. A vertical line is put at α = average segment duration (=21.99 frames).

equation:

$$\hat{m} = \underset{m}{\text{argmax}} \{\log \mathrm{P}(\mathbf{Y}|m) + \beta L \log \mathrm{P}(L|m)\}, \tag{4.52}$$

where βL can be considered as a segment-length-dependent dynamic weighting factor. Figure 4.4 shows the classification performance on the training data as a function of dynamic weighting factor, β. It can be seen from this figure that we get the best classification performance when $\beta = 1.0$. Thus, in both the approaches, the optimal value of the weight is approximately equal to the segment length. However, the dynamic weighting is intuitively more appealing, because duration probabilities are weighted depending on segment lengths. Moreover, it is easy to use, as we do not need to calculate the average segment length from the training data. We use the dynamic weighting with $\beta = 1$ in all the classification experiments reported here-after.

The classification results on the test data using the duration probability (dynamic weighting and $\beta = 1.0$) are shown in Table 4.3. From this table, it can be seen that VTM gives consistently higher classification rates compared to constant variance models (EM).

Comparison between VTM and HMM

In order to compare the performance of VTM with a conventional HMM system, we investigate a three-state context independent HMM. Continuous density HMMs for 16 vowels are trained using the EM algorithm with 20 iterations. No model parameter is tied. No state skip is allowed in the transition, that is, three frames are required for the minimum duration. The total number of the free parameters for each phoneme HMM is $S(MD +$

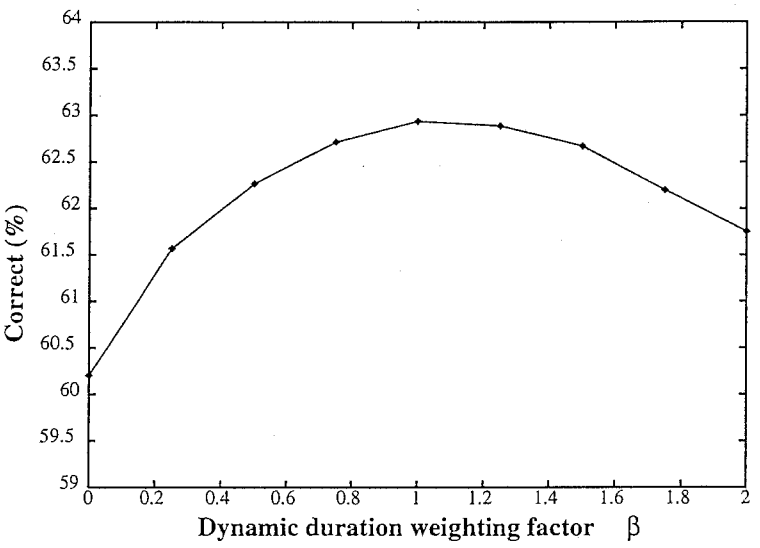


Figure 4.4: Classification performance as a function of dynamic weighting factor β of duration probability.

Table 4.3: Classification rate (%)(with duration probability).

Method	Number of mixtures				
	1	3	5	7	9
Clustering	56.8	59.5	61.6	62.2	62.4
EM	56.8	62.3	63.8	65.4	65.9
VTM	58.7	63.4	65.0	66.0	66.2

$MD + M + 1$) (SMD for means, SMD for variances, SM for mixture weights, and S for transition probabilities), where S is the number of HMM states. As for VTM, the total number of free parameters for each phoneme VTM is $(R+1)(MD+MD)+M+68$ ($(R+1)MD$ for means, $(R+1)MD$ for variances, M for mixture weights, and 68 for duration probabilities (In this chapter, the minimum and maximum duration length are set to 3 and 70, respectively.). When $M = 5$, the total number of the free parameters for HMM, VTM without duration probability, and VTM with duration probability become 618, 605 and 673, respectively. Classification results for the VTM and HMM systems are listed in Table 4.4. From this table, it can be seen that VTMs provide about 4% improvement for each mixture against the three-state HMM whose number of free parameters is equal to that of the VTMs'.

Table 4.4: Classification results (%) for the VTM and HMM systems.

Method	Number of mixtures				
	1	3	5	7	9
VTM(without duration)	56.7	60.7	62.6	63.6	63.8
VTM(with duration)	58.7	63.4	65.0	66.0	66.2
HMM	54.1	58.7	60.9	62.0	62.4

4.4 Time scaling

Most of segment modeling can be divided into two broad categories in terms of time scaling used for segment modeling: absolute time scaling[36][43] and normalized time scaling[35][30][42][41]. However, these studies do not report the relative merits and demerits of these time scaling methods for segment-based speech modeling.

Here, we investigate three kinds of time scaling methods through the polynomial segment model[30] and compare them in terms of their recognition performance on the same database.

4.4.1 Normalized time scaling

Consider an L (in frames) length sequence of observation vectors $\{\mathbf{y}_1,\ldots,\mathbf{y}_L\}$, where $\mathbf{y}_t$ is a D-dimensional observation (e.g. cepstrum) vector at time t. This sequence defines a segment which can be expressed in the form of an $L \times D$ matrix $\mathbf{Y}$. In the polynomial segment model[30], this segment is represented by an R-th order trajectory model as follows:

$$\mathbf{Y} = \mathbf{Z}\mathbf{B} + \mathbf{E}, \tag{4.53}$$

where $\mathbf{Z}$ is an $L \times (R+1)$ design matrix defined by

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 1 & \frac{1}{L-1} & \cdots & \left(\frac{1}{L-1}\right)^R \\ \vdots & \vdots & & \vdots \\ 1 & \frac{t-1}{L-1} & \cdots & \left(\frac{t-1}{L-1}\right)^R \\ \vdots & \vdots & & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}, \tag{4.54}$$

$\mathbf{B}$ is an $(R+1) \times D$ trajectory parameter matrix and $\mathbf{E}$ is an $L \times D$ residual error matrix. Design matrix $\mathbf{Z}$ deals with normalizing different length segments uniformly between time periods 0 and 1. The residual error vectors are considered independent between frames and

identically distributed as a Gaussian with zero mean and an invariant covariance matrix Σ.

4.4.2 Absolute time scaling

For the absolute time scaling case, we use the following design matrix:

$$\mathrm{Z} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 2 & \dots & 2^R \\ \vdots & \vdots & & \vdots \\ 1 & t & \dots & t^R \\ \vdots & \vdots & & \vdots \\ 1 & L & \dots & L^R \end{bmatrix}. \tag{4.55}$$

In Eq.(4.55), the starting frames (i.e. 1) are fixed as reference points. This type of time scaling has been used by Deng[36].

We can also consider the case where the center frames of segments (i.e. $L/2$) are fixed as reference points[43]. In this case, the design matrix is given by

$$\mathrm{Z} = \begin{bmatrix} 1 & [L/2]-L+1 & \dots & \{[L/2]-L+1\}^R \\ 1 & [L/2]-L+2 & \dots & \{[L/2]-L+2\}^R \\ \vdots & \vdots & & \vdots \\ 1 & t & \dots & t^R \\ \vdots & \vdots & & \vdots \\ 1 & [L/2] & \dots & [L/2]^R \end{bmatrix}, \tag{4.56}$$

where $[\cdot]$ indicates the integer of the expression in the brackets.

4.4.3 Experiments

Conditions

To investigate the relative performances of three time scaling methods (i.e. Eq.(4.54), Eq.(4.55) and Equation (4.56)), we performed experiments on a speaker-independent classification task using the TIMIT corpus. A total of 462 speakers (3,696 sentences) were employed for PSM training and 168 speakers (1,344 sentences) were employed for testing. The regression order of the mean trajectories, R, was set to 0 (constant), 1 (linear) or 2 (quadratic). We generated single mixture context independent PSMs with diagonal covariance matrices for 61 phoneme sets from 26-dimensional MFCCs (12-dimensional MFCC + power and their derivatives) computed with 25.6 msec window duration and 10 msec frame period. Segment Y was classified as phoneme $\hat{m}$ by $\hat{m} = \mathrm{argmax}_m \log \mathrm{P}(\mathrm{Y}|m)$.

Table 4.5: Classification rates for 61 phoneme sets without duration probability (%). Evaluation results for 39 phoneme sets are shown in brackets.

Design matrix (Scaling type)	Regression order		
	0	1	2
Eq.(4.3)(normalized)	43.0 (55.5)	47.9 (60.3)	49.5 (62.1)
Eq.(4.55)(absolute)	43.0 (55.5)	46.3 (58.9)	47.9 (60.7)
Eq.(4.56)(absolute)	43.0 (55.5)	47.2 (59.9)	48.3 (61.3)

Table 4.6: Classification rates for 61 phoneme sets with duration probability (%). Evaluation results for 39 phoneme sets are shown in brackets.

Design matrix (Scaling type)	Regression order		
	0	1	2
Eq.(4.3)(normalized)	45.5 (57.9)	50.0 (62.2)	51.5 (63.9)
Eq.(4.55)(absolute)	45.5 (57.9)	47.8 (60.4)	49.2 (61.9)
Eq.(4.56)(absolute)	45.5 (57.9)	48.8 (61.4)	49.5 (62.6)

Results

Table 4.5 shows the classification rates without duration probabilities. Classification rates evaluated with 39 phoneme sets are shown in brackets. Table 4.6 shows the classification rates with duration probabilities. In this case, segment $\mathbf{Y}$ was classified as phoneme $\hat{m}$ by $\hat{m} = \text{argmax}_m \{\log \mathrm{P}(\mathbf{Y}|m) + L \log \mathrm{P}(L|m)\}$, where $\mathrm{P}(L|m)$ is the L-frame duration probability given phoneme m computed as a histogram of the training segment duration. Segment-length-dependent weighting L was used for matching the dynamic ranges of $\log \mathrm{P}(\mathbf{Y}|m)$ and $\log \mathrm{P}(L|m)$. From these tables, we can see that the normalized time scaling method whose design matrix is given by Eq.(4.3) gives consistently better performances for the all cases compared to the absolute time scaling method whose design matrix is given by Eq.(4.55) or Eq.(4.56). Note that the 0-th order PSMs give the same performance for all three methods.

4.5 Conclusions

In this chapter, we have proposed mixture density polynomial segment models (MDPSM), where higher order regression models are used not only for mean trajectory modeling but also for time-varying variance modeling. We have developed a theoretical formulation for estimating the model parameters using the EM algorithm. We have conducted speaker-independent vowel classification experiments using the TIMIT database and reported the classification results. These results indicate that the proposed model gives a consistently

better performance than the MDPSM proposed by Gish and Ng [30]. In addition, the proposed model shows significant improvement in classification performance over the conventional HMM.

As the proposed modeling requires the explicit evaluation of different segmentations, the computational requirements in PSM or VTM decoding are generally higher than those in HMM decoding. To reduce the cost of segment evaluations, several techniques have been studied [45] [46] [47]. These researches are important to apply segment models to continuous speech recognition systems.

In Section 4.4, three time scaling methods used for segment modeling were compared under the same conditions (i.e. recognition system and database). The experimental results on the TIMIT classification task show that the normalized time scaling method consistently gives the best performance.

Chapter 5

Pronunciation modeling

This chapter outlines a method for automatically generating a pronunciation dictionary based on a pronunciation neural network that can predict plausible pronunciations (*realized pronunciations*) from the canonical pronunciation. This method can generate multiple forms of realized pronunciations using the pronunciation network. For generating a sophisticated realized pronunciation dictionary, two techniques are described: (1) realized pronunciations with likelihoods and (2) realized pronunciations for word boundary phonemes.

We define canonical and realized pronunciations as follows.

- *Canonical pronunciation*: Standard phoneme sequences assumed to be pronounced in read speech. Pronunciation variations such as speaker variability, dialect, or coarticulation in conversational speech are not considered.
- *Realized pronunciation*: Actual phoneme sequences pronounced in speech. Various pronunciation variations due to speaker or conversational speech can be included.

In the following sections, we first present training and generation procedures based on a pronunciation neural network. In Section 5.3, the proposed method is applied to a task of pronunciation dictionary generation for spontaneous speech recognition. Section 5.4 shows results of recognition experiments and Section 5.5 gives a discussion of the presented work. Finally, preliminary study of pronunciation modeling based on Acoustic Sub-Word Unit (ASWU) is presented in Section 5.6.

5.1 A historical review

The creation of an appropriate pronunciation dictionary is widely acknowledged to be an important component for a speech recognition system. One of the earliest successful

attempts based on phonological rules was made at IBM[48]. Generating a sophisticated pronunciation dictionary is still considered to be quite effective for improving the system performance on large vocabulary continuous speech recognition (LVCSR) tasks[49]. However, constructing a pronunciation dictionary manually or by a rule-based system requires time and expertise. Consequently, research efforts have been directed at constructing a pronunciation dictionary automatically. In the early 1990s, the emergence of phonetically-transcribed (hand-labeled) medium-size databases (e.g. TIMIT[25] and Resource Management[50]) encouraged a lot of researchers to explore pronunciation modeling [51][52][53]. Although all of these approaches are able to automatically generate pronunciation rules, hand-labeled transcriptions by expert phoneticians are required. As a result, automatic phone transcriptions generated by a phoneme recognizer, which enable one to cope with a large amount of training data, have been used in pronunciation modeling [54][55][56][57]. Recently, LVCSR systems have started to treat spontaneous, conversational speech, such as the Switchboard corpus and consequently, pronunciation modeling has become an important topic because word pronunciations vary more here than in read speech [58][59][60].

5.2 Automatic generation of multiple pronunciations

5.2.1 Pronunciation network

To predict realized pronunciations from a canonical pronunciation, we employ a multilayer perceptron as shown in Figure 5.1. In this chapter, a realized pronunciation $A(m)$ for a canonical pronunciation $L(m)$ is predicted from the five phonemes (i.e. quintphone) of canonical pronunciations $L(m-2), \ldots, L(m+2)$[1].

Now we have two questions: (1) how to train a pronunciation network; and (2) how to generate multiple realized pronunciations by using the trained pronunciation network. These questions are answered in the following sections.

5.2.2 Training procedures

Training data preparation

To train a pronunciation network, first we have to prepare the training data, that is, input (canonical pronunciation) and output (realized pronunciation) pairs. The training data can be prepared by generating a realized pronunciation sequence and mapping it to the

[1]This network structure is similar to that employed in NETtalk [61], which can predict an English word pronunciation from its spelling. Note that the pronunciation network is designed to predict realized pronunciations, for the purpose of improving the performance in spontaneous speech recognition, while NETtalk is designed to predict canonical pronunciations for text-to-speech systems.

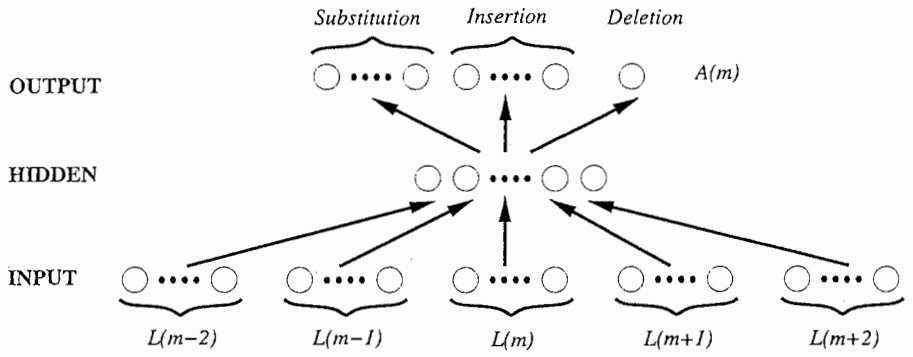


Figure 5.1: Pronunciation network.

canonical pronunciation as follows.

1. Conduct phoneme recognition using speech training data for dictionary generation. The recognized phoneme strings are taken as the realized pronunciation sequence.
2. Align the canonical pronunciation sequence to the realized pronunciation sequence using a dynamic programming algorithm.

For example, if the phoneme recognition result (i.e. realized pronunciation) for the canonical pronunciation /a r a y u r u/, is /a w a u r i u/, the correspondence between the canonical pronunciation and the realized pronunciation can be determined as follows:

```
a  r  a  y  u  r     u  (canonical pron.)
a  w  a     u  r  i  u  (realized pron.),
```

where the second phoneme of the canonical pronunciation, /r/, is substituted by /w/, and /y/ is deleted and /i/ is inserted for the sixth phoneme of the canonical pronunciation, /r/. That is, $L(2) = \mathtt{r}$, $A(2) = \mathtt{w}$, $L(4) = \mathtt{y}$, $A(4) = \mathtt{x}$ (deletion), $L(6) = \mathtt{r}$ and $A(6) = \{\mathtt{r}, \mathtt{i}\}$ (/i/ is an insertion). The correctly recognized phonemes are also treated as substitutions (e.g. /a/ is substituted by /a/). Phoneme recognition is conducted using all of the training data and the aligned results are used as the data for input and output, for the pronunciation neural network training (described in the following section). Note that both the phoneme recognition and alignment procedures are not performed for each word but for each utterance.

Structure of pronunciation network

To train a pronunciation network, a context of five phonemes in the canonical pronunciations, $L(m-2)$, ..., $L(m+2)$, are given as inputs; $A(m)$ aligned to $L(m)$ is given for the outputs. A total of 130 units (26 Japanese phoneme sets times five contexts) are used

in the input layer. The representation of realized pronunciations at the output layer is localized, with one unit representing deletion, 26 units representing substitution, and 26 units representing insertion, providing a total of 53 output units[2].

In the previous example, when / /(deletion), which corresponds to the fourth canonical string /y/, is used as $A(m)$, and /r a y u r/ are used as $L(m-2),\ldots,L(m+2)$. Here, 1.0 is given as the output unit for deletion and as the input units for the /r/ in $L(m-2)$, /a/ in $L(m-1)$, /y/ in $L(m)$, /u/ in $L(m+1)$ and /r/ in $L(m+2)$; 0.0 is given for the other input and output units.

5.2.3 Generation procedures

Realized pronunciation generation

Assume that we want to find the best (i.e. most probable) realized pronunciation for a word W in terms of pronunciation network outputs. Let the canonical pronunciation of W be denoted as $L = [L(1),\ldots,L(|W|)]$, where $|W|$ is the number of phonemes of the canonical pronunciation ($|W| \geq 5$). Realized pronunciation $A = [A(1),\ldots,A(|W|)]$ for L can be obtained in the following steps.

1. Set $i = 3$, $A(1) = L(1)$, and $A(2) = L(2)$.
2. For the quintphone context of the i-th phoneme, $l = [L(i-2),\ldots,L(i+2)]$, input 1.0 in the corresponding input units of the pronunciation network.
3. Find the maximum unit $U1_{out}$ in all of the output units.
   (a) If $U1_{out}$ is found in the substitution units, set $A(i)$ to the phoneme of $U1_{out}$.
   (b) If $U1_{out}$ is found in the insertion units, find another maximum unit $U2_{out}$ in the substitution units. Set $A(i)$ to the phoneme list of $U2_{out}$ and $U1_{out}$, respectively.
   (c) If $U1_{out}$ is the deletion unit, set $A(i) =$ x.
4. Set $i = i + 1$.
5. Repeat step 2 to step 4 until $i = |W| - 1$.
6. Set $A(|W|-1) = L(|W|-1)$ and $A(|W|) = L(|W|)$.

[2]In this chapter, we do not treat insertions of more than two phonemes, because there are relatively very few of them and the number of weights can be reduced.

Table 5.1: Example of inputs and outputs for the canonical pronunciation /a r i g a t o o/. /x/ denotes deletion.

input	phoneme (raw output/normalized output)		
	1	2	3
a r i g a	i (0.9/1.0)	u (0.2/0.22)	o (0.1/0.11)
r i g a t	x (0.4/1.0)	g (0.3/0.75)	b (0.1/0.25)
i g a t o	a (0.8/1.0)	e (0.4/0.5)	o (0.2/0.25)
g a t o o	t (0.5/1.0)	d (0.3/0.6)	k (0.2/0.4)

Multiple pronunciations with likelihoods

Multiple realized pronunciations can be obtained by finding the N-best candidates based on the output values of the network. Suppose that the outputs for the canonical pronunciation /a r i g a t o o/ are obtained as shown in Table 5.1. Then, multiple realized pronunciations can be determined by multiplying each normalized output for all possible combinations and choosing the probable candidates. Although multiple pronunciations can be obtained by setting the number of candidates N as in [62][63], in this chapter we use a likelihood cut-off threshold for the multiplied normalized output. When the threshold is set to 0.4, we get the following six realized pronunciations for /a r i g a t o o/ (normalized pronunciation likelihoods are in brackets).

1. a r i a t o o (1.0)
2. a r i g a t o o (0.75)
3. a r i a d o o (0.6)
4. a r i e t o o (0.5)
5. a r i g a d o o (0.45)
6. a r i a k o o (0.4)

Note that in this example, 1.0 is given as the pronunciation likelihood for the word boundary phonemes; the beginning two phonemes (/a r/) and the ending two phonemes (/o o/).

Integrating the pronunciation likelihood into speech recognition

In conventional speech recognition systems, recognized word sequence $\hat{W}$ given observation O can be obtained by $\hat{W} = \text{argmax}_W P(W|O)$. In this chapter, we extend this formula by considering the realized pronunciation Prn for the word W as follows:

$$\hat{W} = \underset{W \in \mathbf{W}}{\operatorname{argmax}} \sum_{Prn \in W} P(Prn, W|O). \tag{5.1}$$

Using Bayes' Rule, the right-hand side of Eq.(5.1) can be written as

$$\underset{W \in \mathbf{W}}{\operatorname{argmax}} \sum_{Prn \in W} P(O|Prn, W)\ P(W)\ P(Prn|W). \tag{5.2}$$

The first term in Eq.(5.2), $P(O|Prn, W)$, is the probability of a sequence of acoustic observations, conditioned on the pronunciation and word string. This probability can be computed using an acoustic model. The second term in Eq.(5.2), $P(W)$, is the language model likelihood and can be computed using an n-gram word model. We call the third term in Eq.(5.2), $P(Prn|W)$, the *pronunciation model.* In this chapter, the pronunciation network is used as the pronunciation model.

We consider that multiple realized pronunciations mainly represent the pronunciation variability caused by speaker or context differences. That is, for a certain speaker and in a certain context, only one realized pronunciation can be taken for a word pronunciation. Therefore, we omit the summation in Eq.(5.2). Furthermore, by applying exponential weighting to the language probability and pronunciation probability, the acoustic observation O can be decoded by the word sequence based on the following equation:

$$\underset{W \in \mathbf{W}, Prn \in W}{\operatorname{argmax}} P(O|Prn, W)\ P(W)^{\alpha}\ P(Prn|W)^{\beta}, \tag{5.3}$$

where α and β are weighting factors for the language model and the pronunciation model, respectively.

Realized pronunciations for word boundary phonemes

In the previous section, the four phonemes at word boundaries, $L(1)$, $L(2)$, $L(|W|-1)$ and $L(|W|)$, are not predicted by the pronunciation network and the canonical pronunciations are simply used as the realized pronunciations for these phonemes, since the preceding and succeeding words of W are not known at the stage of generating a dictionary. The optimal solution would be to apply the pronunciation network during the decoding stage to generate realized pronunciations on the fly based on hypotheses, but this is technically difficult[64].

To avoid this, an N-best rescoring paradigm has been proposed, by applying decision tree based pronunciation models to the hypothesis generated using the conventional dictionary [58][59]. Although this approach can evaluate pronunciation variations even for word boundary phonemes depending on the preceding and succeeding words, the obtained

improvement is not significant. We suspect that this is mainly due to the fact, that the N-best hypotheses are not obtained using an realized pronunciation dictionary but the baseline dictionary whose decision tree models are not applied.

In [57], cross-word effects are roughly incorporated into the pronunciation modeling through the inclusion of word boundary information as an additional feature in the decision tree clustering. No improvement, however, is obtained by the implementation of cross-word pronunciation modeling. This implies that the contextual dependency for each word, such as a word A which is often followed by a word B, has to be taken into account when predicting realized pronunciations for word boundary phonemes.

Therefore, we take the approach to generate a realized pronunciation dictionary whose variations are considered not only within-word phonemes but also word-boundary phonemes, and use this dictionary to the first pass in decoding. Pronunciation variations for word-boundary phonemes can be taken into account based on language statistics. As language statistics, we employ word bigram models here. Their probabilities are employed to generate realized pronunciations. Because word bigram models give all possible preceding and succeeding words and their frequencies for a certain word, five phoneme contexts (quintphone) of word boundary phonemes are statistically determined.

Consider that we want to find realized pronunciations for the first canonical phoneme $L_{W_C}(1)$ for a word W_C and its canonical pronunciation is $L_{W_C} = [L_{W_C}(1), \ldots, L_{W_C}(|W_C|)]$, where $|W_C|$ is the number of phonemes of the canonical pronunciation. Let a word which can be preceded by W_C be denoted as W_P whose canonical pronunciation is $L_{W_P} = [L_{W_P}(1), \ldots, L_{W_P}(|W_P|)]$, where $|W_P|$ is the number of phonemes of the canonical pronunciation. Then, the quintphone for $L_{W_C}(1)$ is fixed as $[L_{W_P}(|W_P| - 1)$, $L_{W_P}(|W_P|)$, $L_{W_C}(1)$, $L_{W_C}(2)$, $L_{W_C}(3)]$ and the output values of the pronunciation network can be computed. By computing output values for all possible preceding words for L_{W_C}, the output value of the i-th output unit, $\bar{S}_{W_C,i}(1)$, is statistically computed as

$$\bar{S}_{W_C,i}(1) = \sum_{W_P \in \mathrm{W}} P(W_C|W_P) S_{W_C,W_P,i}(1), \tag{5.4}$$

where W is the set of all possible words, $P(W_C|W_P)$ is the conditional probability of W_C given by the word bigram models, and $S_{W_C,W_P,i}(1)$ is the output of the i-th output units computed by the quintphone input using W_C and W_P. Similarly, the output values for other word boundary phonemes, e.g., $L_{W_C}(2)$, $L_{W_C}(|W_C| - 1)$, and $L_{W_C}(|W_C|)$, can be statistically computed. Once the outputs for each output unit are computed, multiple realized pronunciations for W_C can be obtained as shown in Section 5.2.3.

Table 5.2: $w(m-i)$ in Eq.(5.6).

context	$\|i\|$		
	0	1	2
known	1.0	1.0	1.0
statistically known	-	0.8	0.9
unknown	-	0.7	0.8

Table 5.3: Examples of simple and expert pronunciation dictionaries. '-' denotes silence. {|−} represents that both no silence and silence can be used.

example	simple dictionary	expert dictionary
1.	h e y a {|-}	{h|b} e y a {|-}
2.	s u m i m a s e ng {|-}	s u {{m|}i|ng} m a s e ng {|-}
3.	h o sh i {|-}	h o sh i

Reliability weighting for pronunciation likelihoods

Although $P(A|W)$ in Eq.(5.3) given as the normalized likelihood can be used as the score for the pronunciation models, the reliability of $P(A|W)$ for the following three kinds of realized pronunciations will decrease in the following order: (1) obtained from quintphone input (*known*), (2) obtained using language statistics (*statistically known*), and (3) substituted with the canonical pronunciation (*unknown*). Therefore, we introduce a modified pronunciation likelihood $P'(A|W)$ computed by multiplying a weighting factor k to $P(A|W)$ as

$$P'(A|W) = k \cdot P(A|W), \tag{5.5}$$

where k is a $|W|$ dependent constant factor $(0 < k < 1)$ and is defined as

$$k = \frac{\sum_{m=1}^{|W|} \prod_{i=-2}^{2} w(m-i)}{|W|}, \tag{5.6}$$

where $w(m-i)$ is heuristically defined as in Table 5.2. Here, values for $|i| = 2$ are set to bigger than $|i| = 1$ because two phonemes away from the center phoneme would affect the output less than adjacent phonemes.

5.3 Pronunciation dictionary for spontaneous speech recognition

5.3.1 Conditions

A total of 230 speaker (100 male and 130 female) dialogues were used for the pronunciation network and acoustic model training. A 26-dimensional feature vector (12-dimensional mel-cepstrum + power and their derivatives) was computed using a 25.6 msec window duration and a 10 msec frame period. A set of 26 phonemes was used as the Japanese pronunciation representations.

Shared-state context dependent HMMs (CD-HMMs) with five Gaussian mixture components per state [16] were trained. The total number of states was set to 800. By using the CD-HMMs and Japanese syllabic constraints, phoneme recognition was performed on the training data. The phoneme sequences of the recognition results were taken as the realized pronunciations. For each utterance, these realized pronunciations were aligned to their canonical pronunciations transcribed by human experts.

5.3.2 Pronunciation network training

Canonical pronunciations with quintphone context and their correspondent realized pronunciations (about 120,000 samples in total) were used as the inputs and outputs for the pronunciation network training. The structure of the pronunciation network is shown in Figure 5.1, where 130 input units, 100 hidden units, and 53 output units are used. There is also a bias that acts as an additional input constantly set to one. The total number of network weights including the biases becomes 18,453 ($131\times100+101\times53$). For output and hidden units, the sigmoid function with the mean squared error criterion is used because each output produces a number between 0 and 1 but the sum of all outputs does not sum up to one. The network was trained using 1,000 batch iterations and an intermediate network after 500 iterations was used in the following experiments. The differences in the recognition performance for the number of iterations are discussed in Section 5.5.1. The phone recognition accuracy between the canonical pronunciation and the training data is 81.1%. In order to indicate how the pronunciation network can predict pronunciation variation, we evaluated performances of the pronunciation network by the coincidence rate and by the mean squared error (MSE) for the training data. Figure 5.2 shows the coincidence rates of target pronunciation and estimated pronunciation (solid line), and the MSE between the targets and the estimates (dotted line) as a function of the number of training iterations. The coincidence rate for target and canonical pronunciation (shown as *Original Correct* in the figure) is 77.2 %.

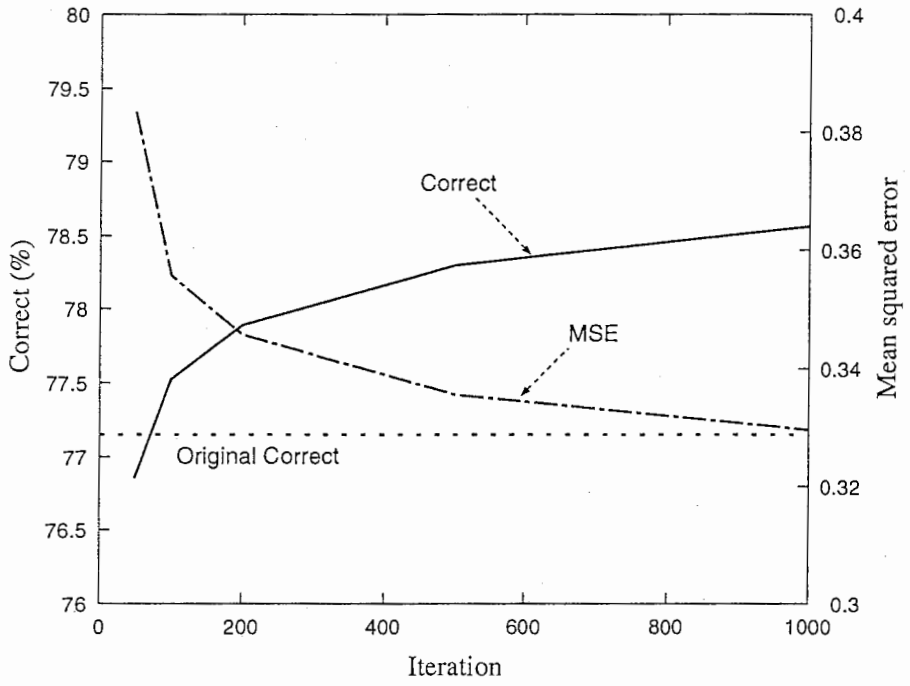


Figure 5.2: Coincidence rates (solid line) and mean squared error (dotted line) of target and estimates for training data as a function of number of training iterations. Coincidence rate for target and canonical pronunciation (shown as *Original Correct*) is 77.2 %.

5.3.3 Generation of realized pronunciation dictionary

We applied the trained pronunciation network to the following two kinds of Japanese pronunciation dictionaries with 7,484 word entries [3] developed for spontaneous speech recognition on a travel arrangement task [14].

- *Simple dictionary*: Each word entry has a single standard canonical pronunciation. This dictionary is automatically generated from its reading represented by Japanese syllabic symbols (*katakana characters*). All entries can be followed by silence (i.e. pause) in recognition.

- *Expert dictionary*: This dictionary is constructed by human experts considering pronunciation variabilities such as successive voicings [4], insertion and substitution of phonemes occurring in spontaneous speech, and possible insertions of a pause.

[3] Multi-words, which were automatically generated by the language modeling [18], were also included in the entries.

[4] Some Japanese word pronunciations change when a compound word is formed. For example, the conjunction of /k o d o m o/ (*child*) and /h e y a/ (*room*) is pronounced /k o d o m o b e y a/.

Table 5.4: Total number of multiple pronunciations for 7,484 word entries.

	simple	expert
before bracket expression	7,484	7,484
after bracket expansion	14,968	17,210
Prop 1 (w/o boundary phonemes)	28,663	33,198
Prop 2 (with boundary phonemes)	33,742	42,103

Table 5.3 shows examples of pronunciations obtained for these dictionaries. '-' denotes silence. In example 2 of the expert dictionary, /s u {{m|}i|ng} m a s e ng {|-}/ represents the following six kinds of multiple pronunciations.

1. s u m i m a s e ng
2. s u i m a s e ng
3. s u ng m a s e ng
4. s u m i m a s e ng -
5. s u i m a s e ng -
6. s u ng m a s e ng -

By expanding the brackets shown in Table 5.3 as above and applying these canonical pronunciations to the pronunciation network, a realized pronunciation dictionary is automatically generated as described in Section 5.2.3. Table 5.4 shows the total number of multiple pronunciations for 7,484 word entries. In this table, Prop 1 and Prop 2 denote the proposed realized pronunciation dictionaries without and with the application of the word boundary phonemes described in Section 5.2.3, respectively. The threshold for limiting the number of realized pronunciations was set to 0.4 for all cases. We used the expanded dictionaries as the baseline dictionaries in the following recognition experiments, but note that the bracket expansion did not affect the recognition performance.

For example, the multiple realized pronunciations obtained from the pronunciation network for a word /w a z u k a/ are shown in Table 5.5. In this example, word boundary phonemes (i.e. /w a/ and /k a/) are also applied to the pronunciation network.

5.4 Evaluation of automatic pronunciation modeling

To investigate the relative effectiveness of the proposed dictionary generated in Section 5.3, we conducted continuous speech recognition experiments on a Japanese spontaneous speech database [14].

Table 5.5: Examples of realized pronunciations with normalized likelihoods for /w a z u k a/.

pronunciation	normalized likelihood
w a z u k a	1.0
a z u k a	0.896
w a z u t a	0.662
a z u t a	0.593

5.4.1 Experimental conditions

The same training data, front-end, and acoustic model described in Section 5.3.1 were used. For the open test set, 42 speaker (17 male and 25 female) dialogues were used. Variable-order n-grams [18] were used as the language model. A multi-pass beam search technique was used for decoding [17]. The language and pronunciation probability weights, α and β in Eq.(5.3) were equally set.

Four kinds of realized pronunciation dictionaries were generated for each baseline dictionary (i.e. the simple and expert dictionaries):

1. Generate a realized pronunciation dictionary with no pronunciation likelihood or language statistics for word boundary phonemes. (Dict 1)
2. The same as for Dict 1 but with the pronunciation likelihoods described in Section 5.2.3. (Dict 2)
3. Generate a dictionary by using language statistics as described in Section 5.2.3. The reliability weighting described in Section 5.2.3 is also applied. (Dict 3)
4. The same as for Dict 3 but with pronunciation likelihoods. (Dict 4)

Note that the total number of multiple pronunciations for Dict 1 and Dict 2 are the same as Prop 1 in Table 5.4. The total number for Dict 3 and Dict 4 are the same as Prop 2 in this table.

5.4.2 Recognition results

Simple dictionary

Recognition results in word error rate (WER) (%) for the simple dictionary are shown in Table 5.6. All four proposed dictionaries outperform the baseline dictionary. Also, the application of pronunciation likelihoods or language statistics (Dict 2, Dict 3, or Dict

Table 5.6: Recognition results for the simple dictionary.

dictionary	likelihood	language stat.	WER (%)
Baseline	–	–	34.5
Dict 1	no	no	33.2
Dict 2	yes	no	31.2
Dict 3	no	yes	32.9
Dict 4	yes	yes	31.1

Table 5.7: Recognition results for the expert dictionary.

dictionary	likelihood	language stat.	WER (%)
Baseline	–	–	29.0
Dict 1	no	no	27.9
Dict 2	yes	no	26.0
Dict 3	no	yes	27.4
Dict 4	yes	yes	26.4

4) boosts the recognition performance of our previous approach (Dict 1). Note that the proposed dictionary generated using both pronunciation likelihoods and language statistics achieved about a 10% error reduction in word error rate compared to the baseline performance.

Expert dictionary

Recognition results in word error rate (%) for the expert dictionary are shown in Table 5.7. First, comparing the two baseline results for the simple and expert dictionaries in Tables 5.6 and 5.7, the expert dictionary (29.0%) can be observed to be significantly superior to the simple dictionary (34.5%). Second, we can see similar improvements by applying the proposed method to the baseline expert dictionary as achieved with the simple dictionary. Note that the proposed dictionary with pronunciation likelihoods and language statistics (Dict 4) achieved about a 9% error reduction in word error rate compared to the baseline performance.

Error analysis

From Tables 5.6 and 5.7, the proposed dictionaries gave consistently better performances than the baseline dictionaries. Here, we found from the recognition results that the number of insertion and substitution errors for the proposed dictionaries significantly decreased compared to the baseline dictionaries. We believe that this is because the proposed dic-

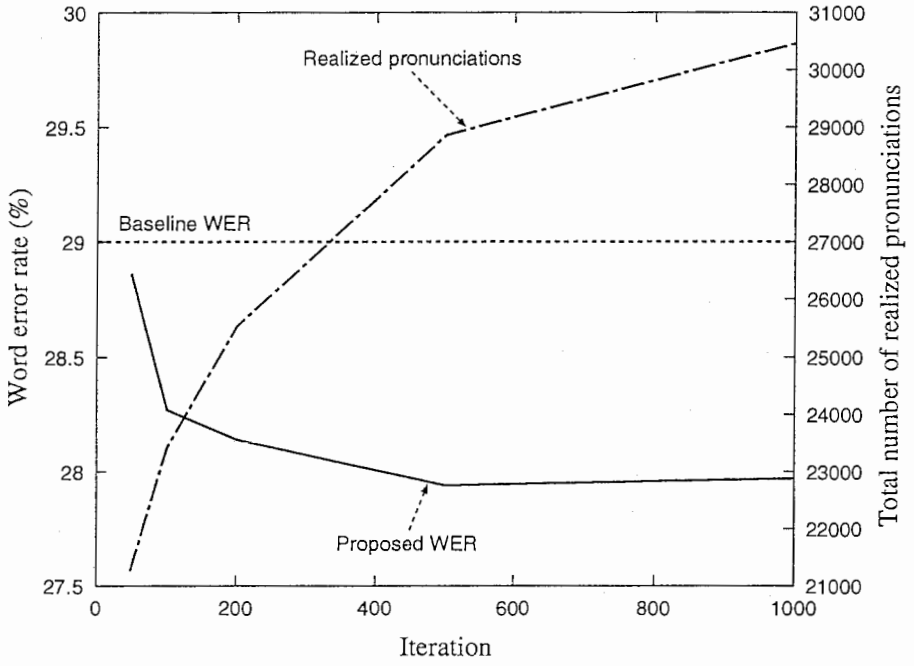


Figure 5.3: Word error rate and total number of realized pronunciations as functions of neural network training iterations.

tionary reduces the errors when long word is incorrectly recognized as a sequence of short words, or a correct word is substituted as another word when the actual pronunciation is slightly different from that in the baseline dictionary.

5.5 Discussion

5.5.1 Number of iterations for NN training

The word accuracy and the total number of realized pronunciations as functions of neural network training iterations (50, 100, 200, 500 and 1,000) are shown in Figure 5.3. The experimental conditions were the same as those described in Section 5.4.1, except that the threshold for the normalized likelihood was set to 0.5. The baseline expert dictionary was used for generating the realized pronunciations. No pronunciation likelihoods or language statistics were used in this experiment. From these results, it can be seen that the recognition performance was improved up to 500 iterations and then saturated, while the realized pronunciations kept increasing. Note that all created dictionaries outperformed the baseline dictionary.

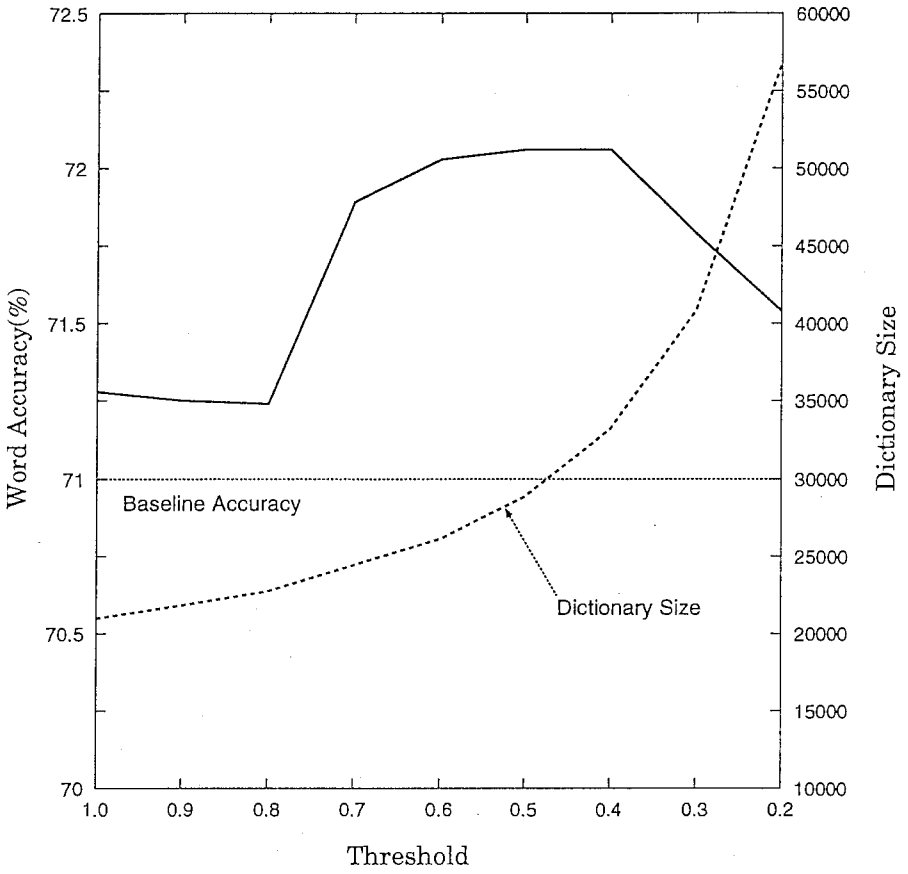


Figure 5.4: Word error rate and total number of realized pronunciations as functions of thresholds, for limiting the number of pronunciations.

5.5.2 Probability cut-off threshold

The word accuracy and total number of realized pronunciations as functions of thresholds (in steps of 0.1 from 0.2 to 1.0), for limiting the number of realized pronunciations, are shown in Figure 5.4. The experimental conditions were the same as those described in Section 5.4.1. The baseline expert dictionary was used for generating the realized pronunciations. No pronunciation likelihoods or language statistics was used in this experiment. From these results, we can see that the recognition accuracy showed a flat peak around 0.4 to 0.6, and the smaller the used threshold is, the lower the word accuracy, and the exponentially larger the pronunciations became. Again, note that all created dictionaries outperformed the baseline dictionary.

Table 5.8: Computational requirements and pronunciation entries for the expert dictionaries normalized against those of the baseline system.

dictionary	CPU time	entries
Dict 1	1.04	1.93
Dict 2	1.20	1.93
Dict 3	1.19	2.45
Dict 4	1.33	2.45

5.5.3 Computational requirements

As the proposed dictionaries have significantly more entries than the baseline dictionaries (see Table 5.4), we investigated the computational requirements of the experiments run for the expert dictionary. The computational requirements and pronunciation entries for the expert dictionaries, which are normalized against those of the baseline system, are shown in Table 5.8. From these results, although the proposed dictionaries required some additional search time compared to the baseline system, the increase in search time was much less than the increase in actual pronunciation entries in the dictionary. We believe that this phenomen is due to the fact that the proposed realized pronunciation dictionaries appropriately represent actually occurring pronunciations in conversational speech. Note that the computational requirements of realized pronunciations highly depend on the recognizer's representation of pronunciations. In our case, the multiple pronunciations are represented in a network with the sharing of common phones, and this will be much faster than a linear lexicon.

5.5.4 Application to another recognition system

To see the effectiveness of the obtained pronunciation dictionary on other recognition systems, is an interesting topic, since we do not know whether the proposed method generates a universal dictionary able to be effective in other systems. To construct another system, we used Janus Recognition Toolkit (JRTk) [65]. Although the same training and test sets were used, the front-end, acoustic modeling, language modeling, and decoder were totally different from those in the previous experiment. Unfortunately, no significant improvement was observed (the WER slightly increased by 0.2%). We therefore suspect that a pronunciation dictionary generated based on phone recognition results, i.e., the proposed method or other similar approaches [54][57][58][?], is difficult to use as a universal dictionary unless the inappropriate pronunciations caused by the phone recognizer (i.e., recognition errors) are filtered out.

5.6 From phonemic units to acoustic units

5.6.1 Background

In speech recognition, current successful approaches are mainly based on context-dependent phone modeling with distribution clustering techniques. These approaches achieve 90% recognition accuracy for unlimited-vocabulary read Wall Street Journal speech and 97% accuracy for a roughly 5000 word vocabulary spontaneous human-computer database query task. However, in the case of human-human dialog utterances, for example, the Switchboard corpus, we have a word error rate of more than 30% even if state-of-the-art acoustic models are used [66]. At ATR, we are collecting spontaneous human-human dialog utterances [14]. While the Switchboard corpus is telephone bandwidth speech, ATR spontaneous database is wideband (16 kHz sampling) speech. Nevertheless, our current system achieves only about 75% word accuracy for speaker-independent models. This suggests that a radical shift in modeling is needed to handle some of the phenomena found in spontaneous speech.

Statistical acoustic models have been studied mainly based on phone units which have been pre-determined independently of real acoustic characteristics of spoken utterances. This pre-determination of phone units causes serious mismatches between input speech characteristics and recognition unit characteristics invoked by corresponding phone sequences, especially when it is applied to highly co-articulated spontaneous speech.

To cope with these mismatches, we combined two advances proposed in previous work [67][30]. The first is the use of acoustically derived segment units (ASUs), which was an active research topic over the last decade [68][69][67] [70]. Secondly, the ASUs are represented by stochastic segment trajectory models where the trajectories can be specified with an arbitrary regression order [30].

In order to implement an ASU-based modeling approach in a speech recognition system, we must solve two problems: (1) How do we design an inventory of acoustically-derived segmental units and (2) How do we model the pronunciations of lexical entries in terms of the ASUs. For the first question, Bacchiani et al. have proposed automatic generation schemes [71]. As for the second question, if we have a large number of word speech to be recognized, we can construct an ASU-based statistical word model[72][67]. In general, however, it is difficult to construct such database especially for large vocabulary systems. To overcome this problem, we have developed an ASU-based word model generation method by composing the ASU statistics.

We start with a brief explanation on ASU generation in Section 5.6.2. Section 5.6.3 then presents a method of mapping scheme between ASUs and a lexicon. Word recognition experiments on speaker-dependent spontaneous speech are described in Section 5.6.4.

5.6.2 Acoustically derived unit generation

Polynomial trajectory models

Each ASU can be represented by stochastic segment trajectory models where the trajectories is specified with an arbitrary regression order. This modeling is the same as that proposed by Gish et al.[30] except that we adopt this modeling not for phones but for ASUs.

Unit design

The ASU generation is carried out as follows. First, acoustic segmentation is done as an initialization of the unit design procedure. Second, the segments resulting from the acoustic segmentation are clustered to form an initial inventory of ASUs. As the acoustic segment boundaries obtained by the first step are sub-optimal for the initial inventory, iterative re-estimation is done. We have confirmed experimentally that only a few iterations are quite enough[70].

Segmentation example

Figure 5.5 shows an example of ASU-based segmentation. The boundaries include (a) hand-labeled phoneme boundaries, (b) acoustically segmented initial boundaries, (c) Viterbi segmented boundaries by initial ASUs (calculated from the acoustic segmentation) and (d) Viterbi segmented boundaries using a secondly calculated ASU set via Viterbi segmentation.

Using this iterative algorithm, we have confirmed experimentally that (1) the test set likelihood using the ASUs is higher than that based on traditional phone models and (2) the likelihood as a function of the iterations on the testing data increases monotonously[71].

5.6.3 Lexical mapping

In ASU-based speech recognition, the main question to be answered is how to represent words in the recognition vocabulary in terms of an appropriate sequence of ASUs. Several techniques have been proposed for the case in which a large number of utterances for each vocabulary word are seen in the training set[67][72]. However, no method has been proposed for unseen words. To cope with this problem, we propose an ASU-based composition method which enables the production of lexicon-based word models. These word models are made in a three-step process: (1) phoneme level composition, (2) word level composition, and (3) hybrid composition using the results of (1) and (2). These steps

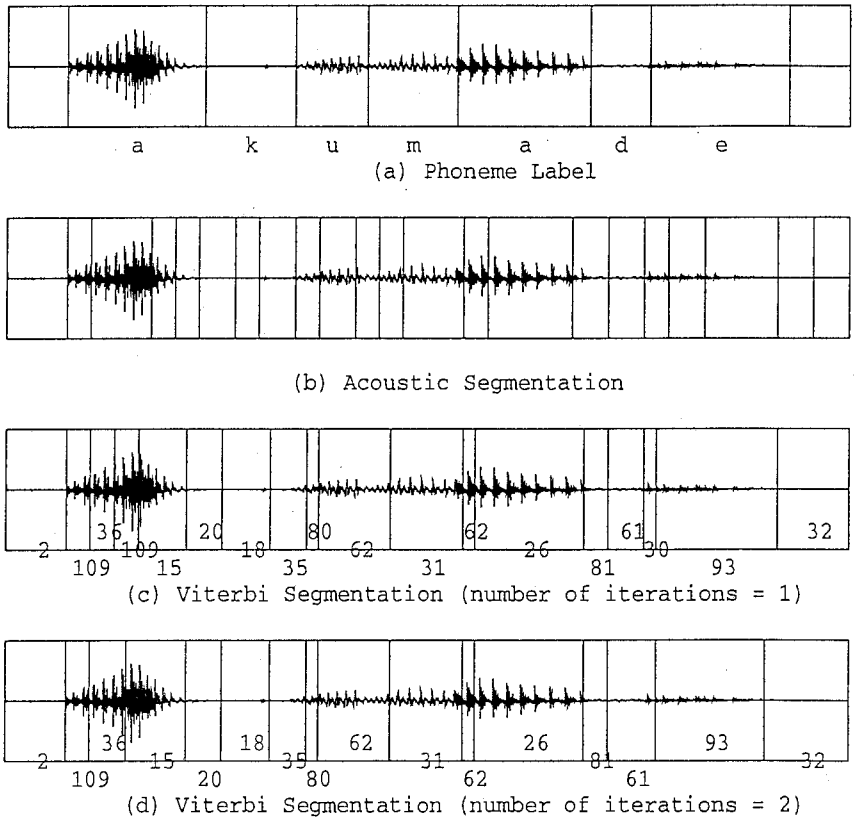


Figure 5.5: Example of ASU-based segmentation.

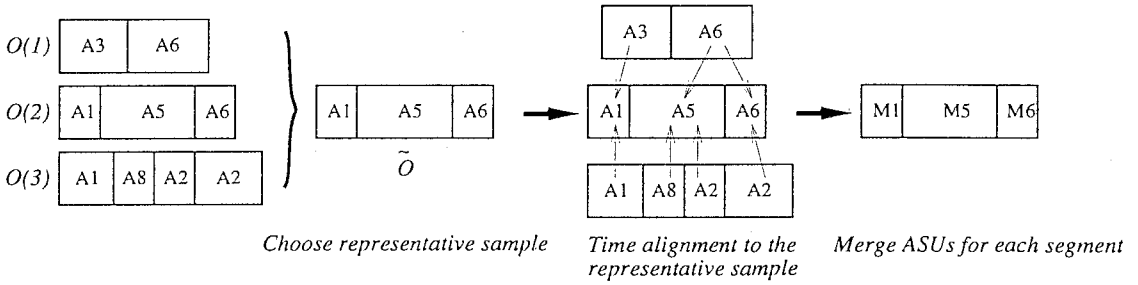


Figure 5.6: Example of phoneme model composition scheme. A3 represents an ASU whose code is three and M1 represents a composed segment code.

are outlined below.

Phoneme-based ASU composition

First, the training data is acoustically segmented by performing a Viterbi segmentation given the ASUs from the training data as described in Section 5.6.2 Then, using a hand-labeled or automatically labeled time-aligned phonemic transcription, segmented ASU sequences are divided into phonemic units. A phoneme model is generated through the following steps:

1. Choose a representative sample $\tilde{O}$ for segment alignment from M samples $O(i)(i = 1, \ldots, M)$ which has the corresponding phoneme as a central phoneme:

$$\tilde{O} = \underset{i}{\operatorname{argmax}} \sum_{m=1}^{M} P(O(m), O(i)) \tag{5.7}$$

   where $P(\cdot)$ indicates the probability between samples that can be calculated using ASU statistics (i.e., the ASU means, the ASU variances and the durations of the time-aligned ASU transcription) and weights according to the contextual coincidence.

2. Align the segments of $O(i)$ to those of $\tilde{O}$ by dynamic programming with their means.

3. For ASUs $(k = 1, \ldots, K)$ which are assigned to one ASU of the representative sample, perform temporal composition to obtain the mean $x_{ph}(m)$ and the variance $\sigma_{ph}(m)$ of the phonemic unit:

$$x_{ph}(m) = \frac{\sum_{k=1}^{K} l_k(m) x_k(m)}{\sum_{k=1}^{K} l_k(m)} \tag{5.8}$$

$$\sigma_{ph}(m) = \frac{\sum_{k=1}^{K} l_k(m) \left[\sigma_k(m) + \{x_k(m) - x_{ph}(m)\}^2\right]}{\sum_{k=1}^{K} l_k(m)} \tag{5.9}$$

   where $x_k(m)$, $\sigma_k(m)$ and $l_k(m)$ are the mean, variance and duration of the k-th ASU segment, respectively.

4. Perform contextual composition by merging ASUs for each phonemic segment using their means $x_{ph}(m)$ and variances $\sigma_{ph}(m)$:

$$\bar{x}_{ph} = \frac{\sum_{m=1}^{M} w_{ph}(m) x_{ph}(m)}{\sum_{m=1}^{M} w_{ph}(m)} \tag{5.10}$$

$$\bar{\sigma}_{ph} = \frac{\sum_{m=1}^{M} w_{ph}(m) \left[\sigma_{ph}(m) + \{x_{ph}(m) - \bar{x}_{ph}\}^2\right]}{\sum_{m=1}^{M} w_{ph}(m)} \tag{5.11}$$

where $\bar{x}_{ph}$ and $\bar{\sigma}_{ph}$ are the composed mean and variance for the phoneme of the lexicon, and $w_{ph}(m)$ is a weight determined based on the contextual coincidence.

Currently, six phonemes (three left and three right) are considered as a phonetic context. The duration statistics is calculated from phonetic durations weighted by the contextual coincidence and represented as a Gaussian distribution. Figure 5.6 illustrates a process of generating a phoneme model from three samples (i.e., $O(1), O(2)$ and $O(3)$). Finally, a word model can be constructed by concatenating the phoneme models. Each word model is made by its own composed phoneme model generation and concatenation.

Word-based ASU composition

Now, we extend the phoneme-based composition method to the word level to generate more precise word model. To this end, in step 1 described in Section 5.6.3, we choose a representative vocabulary speech sample $\tilde{O}_{wd}$ for segment alignment from M samples $O_{wd}(i)(i = 1, \ldots, M)$ whose utterance is the same. Then a word model can be generated in the same way as steps 2 and 3. Note that phoneme context dependent weighting is not performed here.

$$\bar{x}_{wd} = \frac{1}{N} \sum_{n=1}^{N} x_{wd}(n) \tag{5.12}$$

$$\bar{\sigma}_{wd} = \frac{1}{N} \sum_{n=1}^{N} \left[\sigma_{wd}(n) + \{x_{wd}(n) - \bar{x}_{wd}\}^2\right] \tag{5.13}$$

To obtain reliable whole word models, the number of word utterances in the training set needs to be sufficiently large. In general, however, the recording of such homogeneous amounts of speech data is both impractical and unthinkable especially for large vocabulary recognition system. Therefore, we take an approach here to generate reliable (i.e. robust and precise) whole word model by composing this word model and the phoneme concatenated word model. The whole word model can be obtained by DP alignment using their composed means. Under this composing, word sample (i.e. M) dependent weighting w_N is performed.

$$\bar{x}_{word} = \frac{\bar{x}_{ph} + w_N \bar{x}_{wd}}{1 + w_N} \tag{5.14}$$

$$\bar{\sigma}_{word} = \frac{\bar{\sigma}_{ph} + w_N \bar{\sigma}_{wd}}{1 + w_N} + \frac{(\bar{x}_{ph} - \bar{x}_{word})^2 + w_N (\bar{x}_{wd} - \bar{x}_{word})^2}{1 + w_N} \tag{5.15}$$

We expect that the additional use of vocabulary speech data will enable one to construct a robust and precise word model according to the number of lexical utterances in the training data.

Table 5.9: Speech analysis.

Sampling freq.	16 kHz
Preemphasis	0.98
Window	Hamming window, 25.6 ms
Feature parameter	10-dimensional MFCC + energy
Frame shift	10 ms

Table 5.10: ASU generation conditions.

Acoustic segmentation	Distortion threshold	1.0
	Regression order	0
	Distortion measure	Mahalanobis
Clustering	Codebook size	120
	Distortion measure	ML
	Covariance type	Diagonal

5.6.4 Preliminary experiments

Conditions

To confirm the baseline performance of the ASU composition method, we performed 200-word recognition experiments on speaker-dependent spontaneous speech, using the "ATR Travel Arrangement Corpus"[73][14]. The conditions for the feature parameter extraction and ASU generation are listed in Table 5.9 and Table 5.10, respectively. To generate ASUs and word models, 237 spontaneous speech utterances by one male speaker were used. Phoneme label information, which was needed for the word model generation step, was obtained by performing automatic segmentation using speaker dependent HMMs. The context dependent weighting w_{ph} discussed in Section 5.6.3 was selected as $w_{ph} = i + j + k$, where i and j are the number of coincidences of the left and right phoneme context, respectively. If both i and j were greater than or equal to 1, k was set to 20. Otherwise, k was set to zero. Word sample dependent weighting w_N was set to $0.1n$, where n is the number of word samples, so as to be weighted equally between whole word model which was constructed from ten samples and phoneme concatenated model.

Results

As our reference system, we used HMM-based context dependent phoneme models[27] with 400 states and a single Gaussian density function per state. The model topology generation and training were performed using the same 237 spontaneous speech utterances.

The recognition experiments showed that the recognition rate of the phoneme-based ASU composition method described in Section 5.6.3 was 80.5%, while conventional recognition rate was 80.0%. Furthermore, by word-based ASU composition method described in Section 5.6.3, the recognition rate was improved to 82.0%. These results support our approach and superior performance for continuous spontaneous speech recognition.

5.7 Conclusions

In this chapter, a method for automatically generating a pronunciation dictionary based on a pronunciation neural network has been proposed. The proposed method can generate multiple forms of realized pronunciations using the pronunciation network. Our approach is based on a pronunciation neural network that can predict plausible pronunciations (realized pronunciations) from the canonical pronunciation; most other approaches use decision trees for pronunciation modeling [51][52] [57][58][59][60]. We focused on two techniques: (1) realized pronunciations with likelihoods based on the neural network outputs and (2) realized pronunciations for word boundary phonemes using word bigram based language statistics. Experimental results on spontaneous speech recognition show that automatically-derived pronunciation dictionaries give higher recognition rates than the conventional dictionary. We also confirmed that the proposed method can enhance the recognition performance of a dictionary constructed based on expertise. In this chapter, only a quintphone context is used for predicting pronunciation variations, that is, words whose quintphone contexts are the same, have the same pronunciation variation. However, other factors (e.g. part-of-speech) can be easily incorporated into the pronunciation network by having additional units for these factors. Although the proposed method requires the fixed input window (i.e. a context of five phonemes), this requirement could be relaxed by adding word boundary phones (pad phones) to the beginning and ending of the word. In addition, we expect the multiple pronunciation dictionary to be a useful resource for acoustic model retraining by realigning the training data [55][58][59][60].

In Section 5.6, we have presented a new method of ASU-based word model generation. An ASU-based approach has the advantages of growing out of human pre-determined phonemes and of consistently generating acoustic units by using the ML criterion. The effectiveness of the proposed method is shown through spontaneous word recognition experiments. With the ASU composition method described in Section 5.6.3, each ASU is transformed by composition to produce lexicon dependent word models. As a result, the model parameters increase in proportion to the size of the lexicon, yet robust and precise word models are still generated. In the word recognition experiment, the regression order of the mean trajectories was set to zero. Though higher regression order modeling would produce better recognition performance than 0-th order modeling, the composition procedure becomes complicated. To overcome these problems, several approaches that

use the original ASU statistics as pronunciation networks, and not based on composing, could exist[70][74]. Recently, more sophisticated approaches have been proposed and the promising results compared to the conventional phone-based systems are obtained on the Resource Management (RM) database [75][76]. In spontaneous speech recognition tasks, these approach are expected to outperform a phone-based systems using canonical pronunciations, since speakers use canonical pronunciations much less frequently in conversational speech.

Chapter 6

Conclusions

This report has addressed some of the important issues in terms of acoustic and pronunciation modeling in automatic speech recognition. In short, the conventional acoustic modeling has the following serious problems.

- Acoustic models are trained using a large amount of speech uttered by a wide variety of speakers. This results in a saturation of recognition performance, since the spectral distributions often exhibit high overlap among different phonemes.

- Only cepstral features and power, and their derivatives are used as a feature vector in most of acoustic modeling, though adequate parameterization is also an important aspect in speech recognition system.

- HMM has the assumption of conditional independence of observations given the state sequence.

Pronunciation modeling is important to compensate for pronunciation variations of words especially found in spontaneous speech. In Chapter 5, we have shown that an expert dictionary designed for spontaneous speech recognition gave significantly better performance than the simple dictionary (a relative improvement of 15.9 % in word accuracy). This result indicates that appropriate automatic pronunciation modeling is indispensable for spontaneous speech recognition.

The purpose of this work was to find a more flexible, sophisticated solution to these problems. In this chapter, we summarize the contributions of the report and suggest some directions for future work.

6.1 Original contribution revisited

The original contributions are summarized as follows. It may be useful to read this summary in conjunction with the "original contribution" list given at the start of this report (Section 1.3).

In Chapter 2, we proposed an acoustic modeling technique based on a 3-D Viterbi decoding procedure which aims at normalizing speaker's variability. The conventional frequency warping based acoustic modeling can be viewed as a special case of the proposed modeling. Experimental results on Japanese spontaneous speech recognition showed that the proposed models yielded a 9.7 % improvement in word accuracy compared to the standard speaker-independent model in the recognition system using a one-pass search.

In Chapter 3, phoneme boundary information was introduced as a supplementary feature in a speech recognition system and a bi-directional recurrent neural network (BRNN) was successfully applied to estimate phoneme boundaries. This feature was incorporated into two kinds of recognition systems; an HMM based system and a segment based system. Experimental results on the TIMIT database showed that phoneme boundary information was effective for reducing recognition time. In the HMM based system, the usage of this feature achieved a 30.2 % reduction in CPU time. In the segment model based system, 99.9% of the segment hypotheses in the full search were pruned by using estimated phoneme boundaries.

In Chapter 4, we proposed mixture density polynomial segment models, where higher order regression models are used to model the mean and the variance as trajectories. Conventional polynomial segment modeling, which is able to relax the independence assumption, can be viewed as a special case of the proposed method (i.e. 0-th order regression for modeling the covariance coefficients). The result of classification experiments on the TIMIT database showed that the proposed model gives consistently better performance than the HMM and the conventional polynomial segment models.

In Chapter 5, a method for automatically generating a pronunciation dictionary based on a pronunciation neural network was proposed to address the problem of pronunciation variation in spontaneous speech. Language statistics (i.e., word bigram) were also successfully used for generation of generalized pronunciations for word boundary phonemes. Experimental results on spontaneous speech recognition showed that automatically-derived pronunciation dictionaries achieved about a 10% error reduction in word accuracy compared to the conventional dictionary. We also confirmed that the proposed method can enhance the recognition performance of a dictionary constructed based on expertise from 71.0% to 73.6%.

6.2 Some directions for further work

There are a number of possibilities for further extensions of the issues addressed in this work. As for acoustic modeling, the following extensions can be considered.

Reduction of computational complexity Although the speaker normalization method described in Chapter 2 achieved better recognition performance compared to the conventional approaches, a much higher computational complexity is required compared to the conventional gender-independent models (typically, it is increased by a factor of 10.). Therefore, the reduction of computational complexity for the proposed method is an important issue for real-time processing.

Optimal constraint In the proposed speaker normalization method, the way of constraints of warping factor transition was not investigated. Two kinds of constraints which were anticipated as reasonable, were tried for inter-phoneme and intra-phoneme state transitions. In addition, the transition probabilities of the warping factor were set heuristically. These probabilities could be estimated from the Viterbi alignments of the training data.

Training with automatically found boundaries In Chapter 3, the BRNN was trained using hand-labeled phone boundaries. However, most of existing speech databases do not have hand-labeled phoneme boundary information. It would be interesting to run BRNN training also with automatically segmented boundaries which are obtained by Viterbi decoding.

Design for left-to-right system One shortcoming of the BRNN is that input information on both sides of the currently evaluated time point is needed. This is disadvantageous for on-line (left-to-right) systems, since a time delay is required.

Segmental feature One of the advantages in the framework of segmental modeling is the potential for incorporating segmental features. In Chapter 4, only duration probabilities were used as segmental features and no spectral features were considered. Although segmental features are theoretically problematic for joint segmentation and recognition problems [77], these features will be helpful for improving the performance.

Non-uniform time scaling In Section 4.4, we have compared three kinds of time scaling and found that the uniformly normalized time scaling showed best classification performance. However, we believe that non-uniform time scaling is superior to uniform scaling, since uniform duration warping often generates unnatural sounds in text-to-speech systems.

As for the pronunciation modeling, the following extensions can be considered.

Incorporating other factors Other factors such as part-of-speech can be easily incorporated into the pronunciation network by having additional units for these factors. Also, pronunciation variations at word boundaries can be explicitly modeled by adding pad phones between word boundaries.

Filtering out noisy pronunciations As the pronunciation network is trained by using phone recognition results, the phone recognition accuracy is an important factor for pronunciation modeling. Better phone recognizers can be constructed by inclusion of a phone language model. Such LMs can be constructed either directly from pronunciation dictionaries or from phonetic transcriptions of speech aligned to the canonical forms from a pronunciation dictionary. These LMs encourage the phone recognizer to produce output which is consistent with canonical pronunciations and as such were considered to be too restraining for the application to pronunciation modeling. Confidence measures may be a good technique to filter out inappropriate pronunciations.

Acoustic model retraining The multiple pronunciation dictionary was shown to be a useful resource for acoustic model retraining by realigning the training data. However, we also believe that acoustic model retraining would be effective, i.e., improve the recognition performance, but only when the initial labels (transcriptions) are erroneous. In other words, acoustic model retraining with a noisy dictionary that includes inappropriate pronunciations might degrade the performance for correctly labeled training data. Whether acoustic model retraining really helps or not depends on the pronunciation dictionary as well as the database. For our database, a big improvement is not expected from retraining, since the initial labels are checked by human experts.

Dynamic pronunciation In this report, pronunciation variations for word-boundary phonemes were taken into account based on language statistics. However, the optimal solution would be to apply the pronunciation network during decoding stage to generate realized pronunciations on the fly based on the hypotheses.

Adaptive pronunciation dictionary Multiple pronunciation models could be prepared and be combined as a weighted mixture, or as a hierarchy.

Dialect modeling A wider issue but still associated with pronunciation variation is dialect, namely the grammatical and lexical variation of speakers, typically as a function of their geographic origin. It would be interesting to incorporate effects associated with these variations into the language model.

Comparison with the decision tree based approach Decision tree based pronunciation modeling [51][52] [57][58][59][60] is the major alternative approach. One theoretical difference is that the proposed pronunciation network is generated based on discriminative training using the whole training data, while the decision tree is

grown based on the maximum mutual information estimation (MMIE) training using a part of the training data that belongs to a certain node. Although the proposed method achieves relatively better results than the approach using decision trees, a conclusion cannot be drawn, since the two approaches were not compared under the same conditions. Therefore, a comparison between the neural network based approach and the decision tree based approach would be an interesting topic.

Acknowledgments

First, I would like to thank Dr. Seiichi Yamamoto, the President of ATR Interpreting Telecommunications Research Laboratories, Dr. Yasuhiro Yamazaki, the previous President of ATR Interpreting Telecommunications Research Laboratories, for their encouragement and for giving me the opportunity to achieve this study. I thank Dr. Hideyuki Tamura, the previous Head of the Media Technology Laboratory, Canon Inc., for giving me the opportunity for coming to ATR.

I would also like to express my sincere gratitude to Dr. Yoshinori Sagisaka, the Head of the Department 1, for his support, encouragement, and guidance.

I also owe a great deal to the current and the former members of ATR Interpreting Telecommunications Research Laboratories. Mr. Michael Schuster offered me his neural network programs. Michael gave me numerous improtant suggestions about the works on phoneme boundary estimation and pronunciation modeling described in Chapter 3 and Chapter 5, respectively. Mr. Hirofumi Yamamoto helped me to implement efficiently the 3-D Viterbi based recognition system and the probabilistic multiple pronunciation modeling described in Chapter 2 and Chapter 5, respectively. Prof. Kuldip Paliwal (currently with Griffith University) gave me useful advice on the experiments of polynomial segment modeling described in Chapter 4. Kuldip had patiently corrected my drafts covered with numerical formula mistakes more than twenty times. Mr. Michiel Bacchiani (currently with Boston University) offered me his segment modeling software package used in Chapter 4 and Chapter 5.

I also thank Mr. Yasuhiro Taniguchi from Toyohashi University, Mr. Muneyasu Miyamoto from Nara Institute of Science and Technology, Ms. Sophie Aveline from ENST in France, Mr. Kouji Sugimura from Nara Institute of Science and Technology, Mr. Takayoshi Yoshimura from Nagoya Institute of Technology and Satoru Tsuge from Tokushima University for their support in part of the experiments.

Publications related to this report

Journal

1. Toshiaki Fukada, Mike Schuster and Yoshinori Sagisaka: "Phoneme boundary estimation using recurrent neural networks and its application," *IEICE Trans*, vol. J81-D-II, no. 7, pp. 1481–1490, 1998. *(in Japanese)*. (Chapter 3)

2. Toshiaki Fukada, Kuldip K. Paliwal and Yoshinori Sagisaka: "Model parameter estimation for mixture density polynomial segment models," *Computer Speech and Language*, 12, 1998. (Chapter 4)

3. Toshiaki Fukada and Yoshinori Sagisaka: "Automatic generation of a pronunciation dictionary based on pronunciation networks," *IEICE Trans*, vol. J80-D-II, no. 10, pp. 2626–2635, 1997. *(in Japanese)*. (Chapter 5)

4. Toshiaki Fukada, Takayoshi Yoshimura and Yoshinori Sagisaka: "Automatic generation of multiple pronunciations based on neural networks," *Speech Communication*. *(conditionally accepted)* (Chapter 5)

International Conference

1. Toshiaki Fukada and Yoshinori Sagisaka: "Speaker normalized acoustic modeling based on 3-D Viterbi decoding," *Proc. ICASSP-98*, pp. 437–440, 1998. (Chapter 2)

2. Toshiaki Fukada, Sophie Aveline, Mike Schuster and Yoshinori Sagisaka: "Segment boundary estimation using recurrent neural networks," *Proc. Eurospeech-97*, pp. 2839–2842, 1997. (Chapter 3)

3. Toshiaki Fukada, Yoshinori Sagisaka and Kuldip K. Paliwal: "Model parameter estimation for mixture density polynomial segment models," *Proc. ICASSP-97*, pp. 1403–1406, 1997. (Chapter 4)

4. Toshiaki Fukada and Yoshinori Sagisaka: "Neural network based pronunciation modeling with applications to speech recognition," *Proc. ICSLP-98*, 1998. (Chapter 5)

5. Toshiaki Fukada, Takayoshi Yoshimura and Yoshinori Sagisaka: "Automatic generation of multiple pronunciations based on neural networks and language statistics," *Proc. ESCA workshop on Modeling pronunciation variation for automatic speech recognition*, pp. 41–46, 1998. (Chapter 5)

6. Toshiaki Fukada and Yoshinori Sagisaka: "Automatic generation of a pronunciation dictionary based on a pronunciation network," *Proc. Eurospeech-97*, pp. 2471–2474, 1997. (Chapter 5)

7. Toshiaki Fukada, Michiel Bacchiani, Kuldip K. Paliwal and Yoshinori Sagisaka: "Speech recognition based on acoustically derived segment units," *Proc. ICSLP-96*, pp. 1077–1080, 1996. (Chapter 5)

Conference & Workshop

1. Toshiaki Fukada and Yoshinori Sagisaka: "Speaker normalization based on 3-D Viterbi decoding," *ASJ fall meeting*, 2-1-3, 1998. *(in Japanese).* (Chapter 2)

2. Toshiaki Fukada, Sophie Aveline, Mike Schuster and Yoshinori Sagisaka: "Phoneme boundary estimation using recurrent neural networks and its application to speech recognition," *IEICE Tech. Rep.*, SP97-15, pp. 41–48, June 1997. *(in Japanese).* (Chapter 3)

3. Toshiaki Fukada, Sophie Aveline, Mike Schuster and Yoshinori Sagisaka: "Segment boundary estimation using recurrent neural networks," *ASJ spring meeting*, 3-6-8, pp. 103–104, 1997. *(in Japanese).* (Chapter 3)

4. Toshiaki Fukada and Yoshinori Sagisaka: "Automatic phoneme segmentation using phoneme boundary detection networks," *ASJ fall meeting*, 2-Q-10, pp. 135–136, 1997. *(in Japanese).* (Chapter 3)

5. Toshiaki Fukada, Yasuhiro Taniguchi and Yoshinori Sagisaka: "Model parameter estimation for mixture density stochastic segment models," *IEICE Tech. Rep.*, SP96-24, pp. 31–38, June 1996. *(in Japanese).* (Chapter 4)

6. Toshiaki Fukada and Yoshinori Sagisaka: "Automatic generation of pronunciation dictionary based on pronunciation networks," *IEICE Tech. Rep.*, SP96-40, pp. 15–22, December 1996. *(in Japanese).* (Chapter 5)

7. Toshiaki Fukada, Takayoshi Yoshimura and Yoshinori Sagisaka: "Automatic generation of multiple pronunciations based on neural networks and language statistics," *ASJ spring meeting*, 2-Q-30, pp. 179–180, 1998. *(in Japanese).* (Chapter 5)

8. Toshiaki Fukada and Yoshinori Sagisaka: "Pronunciation network generation using outputs of a phonetic typewriter," *ASJ fall meeting*, 2-3-13, pp. 73–74, 1996. *(in Japanese)*. (Chapter 5)

9. Toshiaki Fukada, Michiel Bacchiani, Mari Ostendorf and Yoshinori Sagisaka: "A study on spontaneous speech recognition using acoustically derived segment units," *ASJ spring meeting*, 1-5-16, pp. 39–40, 1996. *(in Japanese)*. (Chapter 5)

ATR Technical Report

1. Kouji Sugimura and Toshiaki Fukada: "Speaker normalized acoustic modeling based on 3-D Viterbi search algorithm," TR-IT-0233, 1997. *(in Japanese)*. (Chapter 2)

2. Sophie Aveline and Toshiaki Fukada: "A study on continuous speech recognition based on polynomial segment models," TR-IT-0200, 1996. (Chapter 3)

3. Yasuhiro Taniguchi and Toshiaki Fukada: "Experimental results on vowel classification using multiple mixture stochastic segment models," TR-IT-0152, 1996. *(in Japanese)*. (Chapter 4)

4. Takayoshi Yoshimura and Toshiaki Fukada: "Automatic generation of multiple pronunciations based on neural networks and language statistics," TR-IT-0244, 1997. *(in Japanese)*. (Chapter 5)

Bibliography

[1] ViaVoice (speech recognition software by IBM): `http://www.software.ibm.com/is/voicetype/`.

[2] P. S. Gopalakrishnan: "Recent advances in speech recognition at IBM Research," *Proc. ICA/ASA-98*, pp. 605–606, 1998.

[3] X. D. Huang: "Spoken language technology research at Microsoft," *Proc. ICA/ASA-98*, pp. 601–602, 1998.

[4] F. -H. Liu, R. M. Stern, X. Huang and A. Acero: "Efficient cepstral normalization for robust speech recognition," *Proc. of DARPA Speech and Natural Language Workshop*, pp. 69–74, 1993.

[5] E. Eide and H. Gish: "A parametric approach to vocal tract length normalization," *Proc. ICASSP-96*, pp. 346–348, 1996.

[6] L. Lee and R. C. Rose: "Speaker normalization using efficient frequency warping procedures," *Proc. ICASSP-96*, pp. 353–356, 1996.

[7] S. Wegmann, D. McAllaster, J. Orloff and B. Peskin: "Speaker normalization on conversational telephone speech," *Proc. ICASSP-96*, pp. 339–341, 1996.

[8] T. Anastasakos, J. McDonough, R. Schwartz, J. Makhoul: "Compact model for speaker-adaptive training," *Proc. ICSLP-96*, pp. 1137–1140, 1996.

[9] P. Zhan and M. Westphal: "Speaker normalization based on frequency warping," *Proc. ICASSP-97*, pp. 1039–1042, 1997.

[10] T. Kosaka, S. Matsunaga and S. Sagayama: "Speaker-independent speech recognition based on tree-structured speaker clustering," *Computer Speech and Language*, vol. 10, pp. 55–74, 1996.

[11] M. Padmanabhan, L. R. Bahl, D. Nahamoo and M. A. Picheny "Speaker clustering and transformation for speaker adaptation in large-vocabulary speech recognition systems," *Proc. ICASSP-96*, pp. 701–704, 1996.

[12] D. Pye and P. Woodland: "Experiments in speaker normalisation and adaptation for large vocabulary speech recognition," *Proc. ICASSP-97*, pp. 1047–1050, 1997.

[13] P. Zhan, M. Westphal, M. Finke and A. Waibel: "Speaker normalization and speaker adaptation – A combination for conversational speech recognition," *Proc. Eurospeech-97*, pp. 2087–2090, 1997.

[14] A. Nakamura, S. Matsunaga, T. Shimizu, M. Tonomura and Y. Sagisaka: "Japanese speech databases for robust speech recognition," *Proc. ICSLP-96*, pp. 2199–2202, 1996.

[15] T. Fukada, K. Tokuda, T. Kobayashi and S. Imai: "An adaptive algorithm for mel-cepstral analysis of speech," *Proc. ICASSP-92*, pp. I-137–I-140, 1992.

[16] M. Ostendorf and H. Singer: "HMM topology design using maximum likelihood successive state splitting," *Computer Speech and Language*, 11, pp. 17–41, 1997.

[17] T. Shimizu, H. Yamamoto, H. Masataki, S. Matsunaga and Y. Sagisaka: "Spontaneous dialogue speech recognition using cross-word context constrained word graphs," *Proc. ICASSP-96*, pp. 145–148, 1996.

[18] H. Masataki and Y. Sagisaka: "Variable-order n-gram generation by word-class splitting and consecutive word grouping," *Proc. ICASSP-96*, pp. 188–191, 1996.

[19] T. Svendsen and F. K. Soong: "On the automatic segmentation of speech signals," *Proc. ICASSP-87*, pp. 77–80, 1987.

[20] A. Ljolje and M. Riley: "Automatic segmentation and labeling of speech," *Proc. ICASSP-91*, pp. 473–476, 1991.

[21] J. Glass, J. Chang and M. McCandless: "A probabilistic framework for feature-based speech recognition," *Proc. ICSLP-96*, pp. 2277–2280, 1996.

[22] S.-L. Wu, M. Shire, S. Greenberg and N. Morgan: "Integrating syllable boundary information into speech recognition," *Proc. ICASSP-97*, pp. 987–990, 1997.

[23] J. Verhasselt, I. Illina, J.-P. Martens, Y. Gong and J.-P. Haton: "The importance of segmentation probability in segment based speech recognizers," *Proc. ICASSP-97*, pp. 1407–1410, 1997.

[24] M. Schuster and K. Paliwal: "Bidirectional recurrent neural networks," *IEEE Trans. on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[25] W. Fisher, G. Doddington and K. Goudie-Marshall: "The DARPA speech recognition research database: Specifications and status," *Proc. DARPA Workshop on Speech Recognition*, pp. 93–99, February 1986.

[26] M. Riedmiller and H. Braun: "A direct adaptive method for faster back-propagation learning: The RPROP algorithm," *Proc. IEEE Int. Conf. Neural Networks*, pp. 586-591, 1993.

[27] J. Takami and S. Sagayama: "A successive state splitting algorithm for efficient allophone modeling," *Proc. ICASSP-92*, pp. 573–576, 1992.

[28] H. Leung, J. Glass, M. Phillips and V. Zue: "Detection and classification of phonemes using context-independent error back-propagation," *Proc. ICSLP-90*, pp. 1061–1064, 1990.

[29] T. Robinson: "Several improvements to a recurrent error propagation network phone recognition system," Cambridge University Engineering Department Tech. Report CUED/F-INFENG/TR82, September 1991.

[30] H. Gish and K. Ng: "A segmental speech model with applications to word spotting," *Proc. ICASSP-93*, pp. II-447–II-450, 1993.

[31] A. Kannan and M. Ostendorf: "A comparison of constrained trajectory models for large vocabulary speech recognition," Boston Univ. Electrical and Computer Engineering Tech. Report ECE-96-007, 1996.

[32] T. Fukada, Y. Sagisaka and K. K. Paliwal: "Model parameter estimation for mixture density stochastic segment models," *Proc. ICASSP-97*, pp. 1403–1406, 1997.

[33] M. Tonomura: "Speaker-independent phone modeling using speaker specific phoneme alignment," *ASJ/ASA third joint meeting*, pp. 987–992, 1996.

[34] N. Campbell and A. Black: "CHATR: a multi-lingual speech re-sequencing synthesis system," *IEICE Technical Report*, SP96-7, pp. 45–52, May 1996. (*in Japanese*).

[35] M. Ostendorf and S. Roukos: "A stochastic segment model for phoneme-based continuous speech recognition," IEEE Trans. on Acoust., Speech and Signal Proc., vol. 37, no. 12, pp. 1857–1869, 1989.

[36] L. Deng: "A generalized hidden Markov model with state-conditioned trend functions of time for the speech signal," *Signal Processing*, vol. 27, pp. 65-78, 1992.

[37] K. Paliwal: "Use of temporal correlation between successive frames in a hidden Markov model based speech recognizer," *Proc. ICASSP-93*, pp. II-215–II-218, 1990.

[38] M. Gales and S. Young: "Segmental hidden Markov models," *Proc. Eurospeech-93*, pp. 1579–1582, 1993.

[39] O. Ghitza and M. Sondhi: "Hidden Markov models with templates as non-stationary states: An application to speech recognition," *Computer Speech and Language*, vol. 2, pp. 101-119, 1993.

[40] T. Robinson, M. Hochberg and S. Renals: "IPA: Improved phone modelling with recurrent neural networks," *Proc. ICASSP-94*, pp. I-37–I-40, 1994.

[41] W. Goldenthal and J. Glass: "Statistical trajectory models for phonetic recognition," *Proc. ICSLP-94*, pp. 1871-1873, 1994.

[42] Y. Gong and J. P. Haton: "Stochastic trajectory modeling for speech recognition," *Proc. ICASSP-94*, pp. I-57–I-60, 1994.

[43] W. J. Holmes and M. J. Russell: "Speech recognition using a linear dynamic segmental HMM," *Proc. Eurospeech-95*, pp. 1611–1614, 1995.

[44] H. Gish and K. Ng: "Parametric trajectory models for speech recognition," *Proc. ICSLP-96*, pp. 466–469, 1996.

[45] V. Zue, J. Glass, M. Philips and S. Seneff: "Acoustic segmentation and phonetic classification in the SUMMIT system," *Proc. ICASSP-89*, pp. 389–392, 1989.

[46] V. Digalakis, M. Ostendorf and J. R. Rohlicek: "Fast search algorithms for phone classification and recognition using segment-based models," *IEEE Trans. on Signal Proc.*, vol. 40, no. 12, pp. 2885–2896, 1992.

[47] T. Fukada, S. Aveline, M. Schuster and Y. Sagisaka: "Segment boundary estimation using recurrent neural networks," *Proc. Eurospeech-97*, pp. 2839–2842, 1997.

[48] L. Bahl, J. Baker, P. Cohen, F. Jelinek, B. Lewis and R. Mercer: "Recognition of a continuously read natural corpus," *Proc. ICASSP-78*, pp. 422–424, 1978.

[49] L. Lamel and G. Adda: "On designing pronunciation lexicons for large vocabulary, continuous speech recognition," *Proc. ICSLP-96*, pp. 6–9, 1996.

[50] P. Price, W. Fisher, J. Bernstein and D. Pallett: "The DARPA 1000-word resource management database for continuous speech recognition," *Proc. ICASSP-88*, pp. 651–654, 1988.

[51] M. Randolph: "A data-driven method for discovering and predicting allophonic variation," *Proc. ICASSP-90*, pp. 1177–1180, 1990.

[52] M. Riley: "A statistical model for generating pronunciation networks," *Proc. ICASSP-91*, pp. 737–740, 1991.

[53] C. Wooters and A. Stolcke: "Multiple-pronunciation lexical modeling in a speaker independent speech understanding system," *Proc. ICSLP-94*, pp. 1363–1366, 1994.

[54] P. Schmid, R. Cole and M. Fanty: "Automatically generated word pronunciations from phoneme classifier output," *Proc. ICASSP-93*, pp. II-223–II-226, 1993.

[55] T. Sloboda: "Dictionary learning: performance through consistency," *Proc. ICASSP-95*, pp. 453–456, 1995.

[56] T. Imai, A. Ando and E. Miyasaka: "A new method for automatic generation of speaker-dependent phonological rules," *Proc. ICASSP-95*, pp. 864–867, 1995.

[57] J. Humphries: "Accent modelling and adaptation in automatic speech recognition," PhD thesis, University of Cambridge, 1997.

[58] E. Fosler, M. Weintraub, S. Wegmann, Y.-H. Kao, S. Khudanpur, C. Galles and M. Saraclar: "Automatic learning of word pronunciation from data," *Proc. ICSLP-96*, pp. 28–29 (addendum), 1996.

[59] M. Weintraub, E. Fosler, C. Galles, Y.-H. Kao, S. Khudanpur, M. Saraclar, S. Wegmann: "Automatic learning of word pronunciation from data," *JHU Workshop-96 Project Report*, 1996.

[60] B. Byrne, M. Finke, S. Khudanpur, J. McDonough, H. Nock, M. Riley, M. Saraclar, C. Wooters, G. Zavaliagkos: "Pronunciation modelling for conversational speech recognition: A status report from WS97," *Proc. 1997 IEEE Workshop on Speech Recognition and Understanding*, 1997.

[61] T. Sejnowski and C. Rosenberg: "NETtalk: a parallel network that learns to read aloud," The Johns Hopkins Univ. Electrical Engineering and Computer Science Tech. Report JHU/EECS-86/01, 1986.

[62] T. Fukada and Y. Sagisaka: "Automatic generation of a pronunciation dictionary based on pronunciation networks," Trans. IEICE, vol. J80-D-II, no. 10 pp. 2626–2635, October 1997. (*in Japanese*).

[63] T. Fukada and Y. Sagisaka: "Automatic generation of a pronunciation dictionary based on a pronunciation network," *Proc. Eurospeech-97*, pp. 2471–2474, 1997.

[64] M. Riley, F. Pereira and E. Chung: "Lazy transducer composition: A flexible method for on-the-fly expansion of context-dependent grammar networks," *Proc. IEEE Automatic Speech Recognition Workshop*, pp. 139–140, 1995.

[65] M. Finke, P. Geutner, H. Hild, T. Kemp, K. Ries and M. Westphal: "The Karlsruhe-Vermobil speech recognition engine," *Proc. ICASSP-97*, pp. 83–86, 1997.

[66] J. Fritsch and M. Finke: "ACID/HNN: Clustering hierarchies of neural networks for context-dependent connectionist acoustic modeling," *Proc. ICASSP-98*, pp. 505–508, 1998.

[67] K. Paliwal: "Lexicon-building methods for an acoustic sub-word based speech recognizer," *Proc. ICASSP-90*, pp. 729–732, 1990.

[68] C. Lee, B. Juang, F. Soong and L. Rabiner: "Word recognition using whole word and subword models," *Proc. ICASSP-89*, pp. 683–686, 1989.

[69] T. Svendsen, K. Paliwal E. Harborg and P. Husøy: "An improved sub-word based speech recognizer," *Proc. ICASSP-89*, pp. 108–111, 1989.

[70] M. Bacchiani, M. Ostendorf, Y. Sagisaka and K. Paliwal: "Design of a speech recognition system based on acoustically derived segmental units," *Proc. ICASSP-96*, 1996.

[71] M. Bacchiani, M. Ostendorf, Y. Sagisaka and K. Paliwal: "Unsupervised learning of non-uniform segmental units for acoustic modeling in speech recognition," *Proc. IEEE ASR Workshop 95*, pp. 141-142, 1995.

[72] T. Svendsen, F. Soong and H. Purnhagen: "Optimizing baseforms for HMM-based speech recognition," *Proc. Eurospeech-95*, pp. 783–786, 1995.

[73] T. Morimoto, N. Uratani, T. Takezawa, O. Furuse, Y. Sobashima, H. Iida, A. Nakamura, Y. Sagisaka, N. Higuchi and Y. Yamazaki: "A speech and language database for speech translation research," *Proc. of ICSLP94*, pp. 1791-1794, 1994.

[74] T. Fukada, M. Bacchiani, K. K. Paliwal and Y. Sagisaka: "Speech recognition based on acoustically derived segment units," *Proc. ICSLP-96*, pp. 1077–1080, 1996.

[75] M. Bacchiani and M. Ostendorf: "Joint acoustic unit design and lexicon generation," *ESCA Workshop on modeling pronunciation variation for automatic speech recognition*, pp. 7–12, 1998.

[76] T. Holter and T. Svendsen: "Maximum likelihood modelling of pronunciation variation," *ESCA Workshop on modeling pronunciation variation for automatic speech recognition*, pp. 63–66, 1998.

[77] M. Ostendorf, V. Digalakis and O. Kimball: "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," *IEEE Trans. on Speech and Audio Proc.*, vol. 4, no. 5, pp. 360–378, September 1996.