DOCTORAL DISSERTATION

ACOUSTIC MODELING FOR HMM-BASED SPEECH RECOGNITION AND SYNTHESIS

DOCTOR OF ENGINEERING

FEBRUARY 2010

Keiichiro OURA

Supervisor : Dr. Keiichi TOKUDA

Department of Computer Science and Engineering
Nagoya Institute of Technology

Abstract

The topics of automatic speech recognition and synthesis have been active areas of research focus. Hidden Markov models (HMMs) are one of widely used statistical models for representing time series by well-defined algorithms. They have successfully been applied to acoustic modeling in speech recognition. HMM-based speech synthesis also has grown in popularity over the last few years. This framework makes it possible to model different voice characteristics, speaking styles, or emotions without recording large speech database. In this paper, improved acoustic modeling is proposed for HMM-based speech recognition and synthesis.

First, hidden semi-Markov model (HSMM) based speech recognition system is designed. In HMMs, state duration probabilities are implicitly modeled by state transition probabilities; state duration probabilities decrease exponentially with time. Geometric distribution calculated by the state transition probabilities of HMMs would be inappropriate state duration probability distribution representation of the temporal structure of speech. One of the solutions to this problem is using HSMM which integrate state duration probability distributions explicitly into the HMM. A variety of attempts to include explicit duration models in speech recognition systems have been reported. However, they are not fully consistent because various approximations are used. Therefore, I constructed a fully consistent HSMM-based speech recognition system and evaluated its performance. The result showed an obvious improvement.

Next, a technique for reducing the footprints of HMM-based speech synthesis systems by tying all covariance matrices is proposed. One of the attractive points of HMM-based speech synthesis is its small footprints. However, further reduction is essential to put it on embedded devices which have very small memories. We empirically know that covariance matrices have smaller impact on the quality of synthesized speech than mean vectors. Based on the knowledge, this paper proposes a context-clustering technique for mean vectors while tying all covariance matrices. The experimental results showed that the proposed technique efficiently shrinked the footprints of an HMM-based speech synthesis system to less than half of its original size while retaining the quality of synthesized

speech.

Next, I define a new integrated model for simultaneous linguistic and acoustic modeling. Standerd text-to-speech (TTS) systems consist of two major modules: text analysis and speech synthesis modules. Conventionally, these two modules are constructed independently. Therefore, if these two modules were combined and trained simultaneously as a unified model, we would expect improved overall performance of a TTS system. I define a new integrated model and firectly-mormulate the TTS problem of synthesiszing a speech waveform from a word sequence. The experimental results demonstrate that the proposed system achives better F_0 modeling accuracy than that of the conventional system.

Finally, unsupervised cross-lingual speaker adaptation system is designed. Many organization have forcused on speech-to-speech translation (S2ST) research topic. Speaker similarity of synthesized speech should be conform to input speaker similarity. Therefore, we integrate two developments, unsupervised adaptation techniques for HMM-based speech synthesis using a word-based large-vocabulary continuous speech recognizer (LVCSR) and cross-lingual speaker adaptation techniques for HMM-based speech sytnehsis sistem. The listening tests show very promising results: it has been demonstrated that the adapted voices sound more similar to the target speaker than the average voice.

For HMM-based speech recognition and synthesis system, avobe improved techniques were proposed and systems using these techniques improved their performance.

Keywords: Speech recognition, speech synthesis, hidden Markov model, hidden semi-Markov model, acoustic modeling, language modeling, unsupervised speaker adaptation, cross-lingual speaker adaptation, embedded device

Abstract in Japanese

近年，音声を情報伝達の手段としたシステムの需要が高まっており，音声認識・合成の研究が盛んに行われている．音声認識における代表的な枠組みとして，音響モデルに統計モデルの一種である隠れマルコフモデル (Hidden Markov Model; HMM) を用いる枠組みがある．HMM は学習データに基づきパラメータを推定する実現容易なアルゴリズムが存在し，トポロジーを認識対象に応じて設計できる，現実的な計算量で学習・認識を行えるなどの特徴があり，十分な学習データ量が与えられれば高い認識性能を示すことが知られている．また，音声合成の分野でも近年 HMM に基づいた手法の研究が盛んに行われている．HMM 音声合成では尤度最大化基準に基づく音声パラメータ生成アルゴリズムを用いて直接音声パラメータを出力し音声を合成するため，単位選択型の音声合成手法と比べて素片接続歪みが生じない，パラメータを変換することで様々な声質に変換できるなどの特徴がある．本論文では，より高性能な HMM 音声認識・合成システムの構築のために，より高性能なモデル化手法の提案を目的とする．

まず，隠れセミマルコフモデル (Hidden Semi-Markov Model; HSMM) を用いた音声認識システムを設計する．HMM の状態継続長は単純な一次のマルコフ過程に基づく遷移確率によって決定されるが，音声のモデル化において HMM の状態遷移は時間方向に強い相関を持っていると考えられ，状態継続長の分布を考慮したモデル化が必要である．この問題に対して，HMM パラメータおよび状態継続長のパラメータを同時に推定する形の HSMM を用いることで状態継続長を精度良くモデル化することができる．しかし，状態継続長を考慮した認識ではモデルの複雑さから様々な近似が用いられてきた．本論文ではそれらの近似を排除して認識性能を評価し，従来法を上回る性能を示した．

次に，HMM 音声合成における共分散パラメータの共有について考察する．HMM 音声合成の特徴の一つにフットプリントが小さい点がある．中でも組み込み向けのシステムには携帯電話等の用途があるが，必要なメモリ等が制限されることが多く，更なるフットプリントの縮小が必要である．HMM 音声合成にコンテキスト依存モデルを用いることで高性能な音響モデルを構築することができ，決定木に基づくコンテキストクラスタリングを用いて状態共有構造を構築する際に，組み込み用途向けに

決定木のサイズを小さくすることも考えられるが，音質の劣化は避けられない．本論文では平均に比べて共分散が音質に与える影響が小さいことに注目し，全てのパラメータの共分散を共有する手法を示す．このパラメータ共有を仮定した上でのコンテキストクラスタリングを行い，パラメータ数を大幅に削減するのみならず，若干の品質改善を達成した．

次に，言語・音響モデルを統合した新しいモデルとその学習アルゴリズムを提案する．通常，HMM 音声合成システムはテキスト解析部と音声合成部の二つのモジュールで構成されており，言語・音響モデルの学習はそれぞれ違うデータベースから独立に行われているが，統合モデルでは提案する学習アルゴリズムによって同時に学習される．従って，文章から音声を合成するという音声合成システムの問題を直接的に定式化しているので適切なモデルが推定できる．客観評価実験により，より適切な言語モデルの推定を確認した．

最後に，教師無し異言語間話者適応システムを設計する．入力言語の音声を認識し，それを出力言語に翻訳し，そして音声を合成する音声翻訳システムは，長年様々な機関で研究されている．合成音声の話者性は入力音声の話者性と一致すべきなので，入力音声を用いた異言語間の話者適応手法が必要となり，さらに，実環境での稼働では，入力音声の認識結果を用いた教師無し話者適応手法も必要である．そこで，この二つの手法を統合した教師無し異言語間話者適応システムを構築し，主観評価実験を行った．教師有り・無しのシステム共に話者性の向上を確認した．

以上のように，本論文ではより高性能な HMM 音声認識・合成システムの構築のために，より高性能なモデル化手法を提案し，その有効性を示す．

Acknowledgement

First of all, I would like to express my sincere gratitude to Keiichi Tokuda, my advisor, for his support, encouragement, and guidance.

I would like to thank Tadashi Kitamura, Akinobu Lee, Chiyomi Miyajima (currently with Nagoya University), Yoshihiko Nankaku, Shinji Sako, Heiga Zen (currently with Toshiba Research Europe) for their technical supports and helpful discussions. Special thanks go to all the members of Kitamura, Tokuda and Lee laboratories for their technical support and encouragement. If somebody was missed among them, my work would not be completed. I would be remiss if I did not thank Natsuki Kuromiya, a secretary of the laboratory, for their kind assistance.

I am grateful to Satoshi Nakamura (currently with NICT Spoken Language Communication Research Laboratories), Shinsuke Sakai (currently with NICT Spoken Language Communication Research Laboratories), Ranniery Maia (currently with Toshiba Research Europe) and Tomoki Toda (currently with Nara Institute of Science and Technology) for giving me the opportunity to work in ATR Spoken Language Communication Research Laboratories, and for their valuable advice.

I am also grateful to Simon King (currently with Edinburgh University), Junichi Yamagishi (currently with Edinburgh University), and Mirjam Wester (currently with Edinburgh University) for giving me the opportunity to work in the Centre for Speech Technology Research of Edinburgh University, and for their valuable advice.

Finally, I would sincerely like to thank my friends for their encouragement.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Speech is the most important ways for human communication, and a number of research topic for human-machine communication have been proposed. Automatic speech recognition (ASR) and text-to-speech synthesis (TTS) are fundamental technologies for human-machine communication. In recent years, they are used in many application such as car navibation system, information retrieval over the telephone, voice mail, speech-to-speech translation (S2ST) system, and so on. The goal of ASR and TTS systems is perfect speech recognition and speech synthesis with natural human voice characteristics.

Most state-of-art speech recognition and synthesis systems are based on large amounts of speech data. This type of approach is generally called corpus-based systems. This approach makes it possible to dramatically improve the performance compared with early systems such as rule-based one. In these days statistical approaches based on hidden Markov models (HMMs) have been dominant both in ASR [1] and TTS [2–5], due to their ease of implementation and modeling flexibility. In this approach, the HMMs are used for modeling sequences of speech spectra. In this paper, improved techniques for acoustic modeling are proposed for HMM-based speech recognition and synthesis.

First, hidden semi-Markov model (HSMM) [6–8] based speech recognition system is designed. In HMMs, state duration probabilities are implicitly modeled by state transition probabilities; state duration probabilities decrease exponentially with time [9]. Geometric distribution calculated by the state transition probabilities of HMMs would be inappropriate state duration probability distribution representation of the temporal structure of speech. One of the solutions to this problem is using HSMM which integrate state duration probability distributions explicitly into the HMM. A variety of attempts to include explicit duration models in speech recognition systems have been reported. However, they are not fully consistent because various approximations are used [5, 10–12]. Therefore, I constructed a fully consistent HSMM-based speech recognition system and evaluated its

performance. The result showed an obvious improvement.

Next, a technique for reducing the footprints of HMM-based speech synthesis systems by tying all covariance matrices is proposed. One of the attractive points of HMM-based speech synthesis is its small footprints. However, further reduction is essential to put it on embedded devices which have very small memories. We empirically know that covariance matrices have smaller impact on the quality of synthesized speech than mean vectors. Based on the knowledge, this paper proposes a context-clustering technique for mean vectors while tying all covariance matrices. The experimental results showed that the proposed technique efficiently shrinked the footprints of an HMM-based speech synthesis system to less than half of its original size while retaining the quality of synthesized speech.

Next, I define a new integrated model for simultaneous linguistic and acoustic modeling. Standerd text-to-speech (TTS) systems consist of two major modules: text analysis and speech synthesis modules. Conventionally, these two modules are constructed independently. Therefore, if these two modules were combined and trained simultaneously as a unified model, we would expect improved overall performance of a TTS system. I define a new integrated model and firectly-mormulate the TTS problem of synthesiszing a speech waveform from a word sequence. The experimental results demonstrate that the proposed system achives better F_0 modeling accuracy than that of the conventional system.

Finally, unsupervised cross-lingual speaker adaptation system is designed. Many organization have forcused on speech-to-speech translation (S2ST) research topic. Speaker similarity of synthesized speech should be conform to input speaker similarity. Therefore, we integrate two developments, unsupervised adaptation techniques for HMM-based speech synthesis using a word-based large-vocabulary continuous speech recognizer (LVCSR) [13] and cross-lingual speaker adaptation techniques [14] for HMM-based speech sytnehsis sistem. The listening tests show very promising results: it has been demonstrated that the adapted voices sound more similar to the target speaker than the average voice.

For HMM-based speech recognition and synthesis system, avobe improved techniques were proposed and systems using these techniques improved their performance. The rest of the present dissertation is organized as follows. The next chapter introduces basic theories of the HMM. Chapters 3 and 4 describe statistical speech recognition and synthesis framework based on the HMM, respectively. Charpter 5, 6, 7, and 8 show the HSMM-based speech recognition, tied covariance technique for HMM-based speech syntheis, simultaneous lingustic and acoustic model training for TTS conversion system, and unsupervised cross-lingual speaker adaptation, respectively. Concluding remarks and future plans are presented in the final chapter.

Chapter 2

Hidden Markov Models

Recently, hidden Markov models (HMMs) are widely used as statistical models for speech recognition. The advantages of using the HMM are that i) it can represent speech as probability distributions, ii) it is robust, iii) efficient algorithms for estimating its model parameters are provided. Parameter estimation and calculation of output probability distributions are described in this chapter.

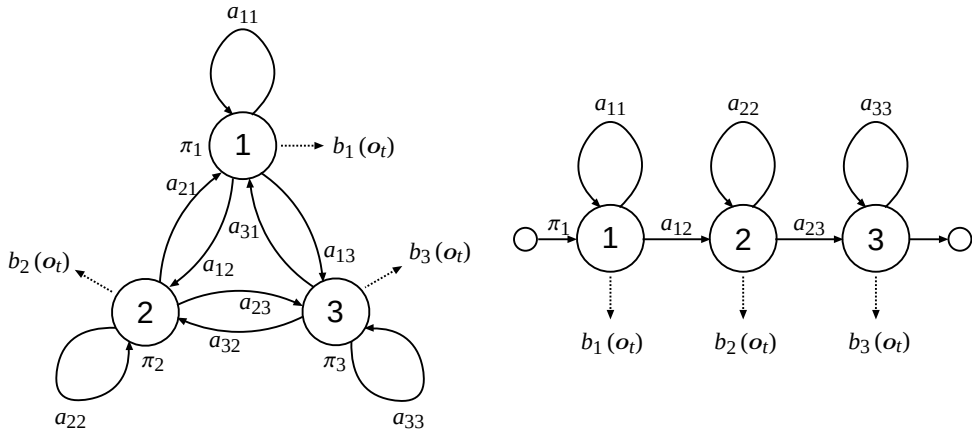
2.1 Definition of HMM

An HMM [15–17] is a finite state machine which generates a sequence of discrete time observations. At each frame it changes states according to its state transition probability distributions, and then generates an observation at time t, $\boldsymbol{o}_t$, according to its output probability distribution of the current state. Therefore, the HMM is a doubly stochastic random process model.

An N-state HMM consist of state transition probability distributions $\{a_{ij}\}_{i,j=1}^{N}$, output probability distributions $\{b_j(\boldsymbol{o}_t)\}_{j=1}^{N}$, and initial state probability distributions $\{\pi_i\}_{i=1}^{N}$. For convenience, the compact notation is used to indicate the parameter set of the model Λ as follows:

$$\Lambda = \left[\{a_{ij}\}_{i,j=1}^{N},\ \{b_j(\cdot)\}_{j=1}^{N},\ \{\pi_i\}_{i=1}^{N}\right] \tag{2.1}$$

Figure 2.1 shows examples of the HMM structure. Figure 2.1(a) shows a 3-state ergodic model, in which every state of the model could be reached from every state of the model in a single step, and Figure 2.1(b) shows a 3-state left-to-right model, in which the state



(a) A 3-state ergodic model (b) A 3-state left-to-right model

Figure 2.1: Examples of HMM structure.

index increases or stays the same state as time increases. The left-to-right HMMs are generally used to model speech parameter sequences, since they can appropriately model signals.

The output probability distributions $\{b_j(\cdot)\}_{j=1}^{N}$ can be discrete or continuous depending on the observations. In continuous distribution HMM (CD-HMM), each output probability distribution is usually modeled by a mixture of multivariate Gaussian components [18] as follows:

$$b_j(\boldsymbol{o}_t) = \sum_{m=1}^{M} w_{jm} \cdot \mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}), \tag{2.2}$$

where M, w_{jm}, $\boldsymbol{\mu}_{jm}$, and $\boldsymbol{\Sigma}_{jm}$ are the number of Gaussian components, the mixture weight, mean vector, and covariance matrix of the m-th Gaussian component of the j-th state, respectively. Each Gaussian component is defined by

$$\mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{jm}, \boldsymbol{\Sigma}_{jm}) = \frac{1}{\sqrt{(2\pi)^K |\boldsymbol{\Sigma}_{jm}|}} \exp\left\{-\frac{1}{2}(\boldsymbol{o}_t - \boldsymbol{\mu}_{jm})^\top \boldsymbol{\Sigma}_{jm}^{-1}(\boldsymbol{o}_t - \boldsymbol{\mu}_{jm})\right\}, \tag{2.3}$$

where symbol $\top$ means transpose of vector or matrix, and K is the dimensionality of an observation vector $\boldsymbol{o}_t$. For each state, $\{w_{jm}\}_{m=1}^{M}$ should satisfy the stochastic constraint

$$\sum_{m=1}^{M} w_{jm} = 1, \quad 1 \leq j \leq N \tag{2.4}$$

$$w_{jm} \geq 0, \quad \begin{matrix} 1 \leq j \leq N \\ 1 \leq m \leq M \end{matrix} \tag{2.5}$$

so that $\{b_j(\cdot)\}_{j=1}^{N}$ are properly normalized, i.e.,

$$\int_{\mathrm{R}^K} b_j(\boldsymbol{o}_t)\, d\boldsymbol{o}_t = 1. \quad 1 \leq j \leq N \tag{2.6}$$

2.2 Calculation of output probability

2.2.1 Total output probability of an observation vector sequence

When a state sequence is determined, a joint probability of an observation vector sequence $\boldsymbol{o} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$ and a state sequence $\boldsymbol{q} = \{q_1, q_2, \ldots, q_T\}$ is calculated by multiplying the state transition probabilities and state output probabilities for each state, that is,

$$p(\boldsymbol{o}, \boldsymbol{q} \mid \Lambda) = \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(\boldsymbol{o}_t), \tag{2.7}$$

where $a_{q_0 q_1}$ denotes π_{q_1}. The total output probability of the observation vector sequence from the HMM is calculated by marginalizing Eq. (2.7) over all possible state sequences,

$$p(\boldsymbol{o} \mid \Lambda) = \sum_{\text{all } \boldsymbol{q}} \prod_{t=1}^{T} a_{q_{t-1}q_t} b_{q_t}(\boldsymbol{o}_t). \tag{2.8}$$

The order of $2T \cdot N^T$ calculation is required, since at every $t = 1, 2, \ldots, T$ there are N possible states that can be reached (i.e., there are N^T possible state sequences). This calculation is computationally infeasible, even for small values of N and T; e.g., for $N = 5$ (states), $T = 100$ (observations), there are on the order of $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations. Fortunately, there is an efficient algorithm to calculate Eq. (2.8) using forward and backward procedures.

2.2.2 Forward-Backward algorithm

The forward-backward algorithm is generally used to calcurate $p(\boldsymbol{o} \mid \Lambda)$, which is the probability of the observation sequence $\boldsymbol{o}$ given the model Λ. If we directly calculate $p(\boldsymbol{o} \mid \Lambda)$, it requires on the order of $2T \cdot N^T$ calculation. The detail of the forward-backward algorithm is described in the following part.

The probability of a partial observation vector sequence from time 1 to t and the i-th state at time t, given the HMM Λ is defined as

$$\alpha_t(i) = p(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t, q_t = i \mid \Lambda). \tag{2.9}$$

$\alpha_t(i)$ is calculated recursively as follows:

1. Initialization
$$\alpha_1(i) = \pi_i b_i(\boldsymbol{o}_1), \quad 1 \leq i \leq N \tag{2.10}$$
2. Recursion
$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(\boldsymbol{o}_t), \quad \begin{array}{l} 1 \leq j \leq N \\ t = 2, \ldots, T \end{array} \tag{2.11}$$
3. Termination
$$p(\boldsymbol{o} \mid \Lambda) = \sum_{i=1}^{N} \alpha_T(i). \tag{2.12}$$

As the same way as the forward algorithm, backward variables $\beta_t(i)$ are defined as

$$\beta_t(i) = p(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T \mid s_t = i, \Lambda), \tag{2.13}$$

that is, the probability of a partial vector observation sequence from time t to T, given the i-th state at time t and the HMM Λ. The backward variables can also be calculated in a recursive manner as follows:

1. Initialization
$$\beta_T(i) = 1, \quad 1 \leq i \leq N \tag{2.14}$$
2. Recursion
$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(\boldsymbol{o}_{t+1}) \beta_{t+1}(j), \quad \begin{array}{l} 1 \leq i \leq N \\ t = T-1, \ldots, 1. \end{array} \tag{2.15}$$

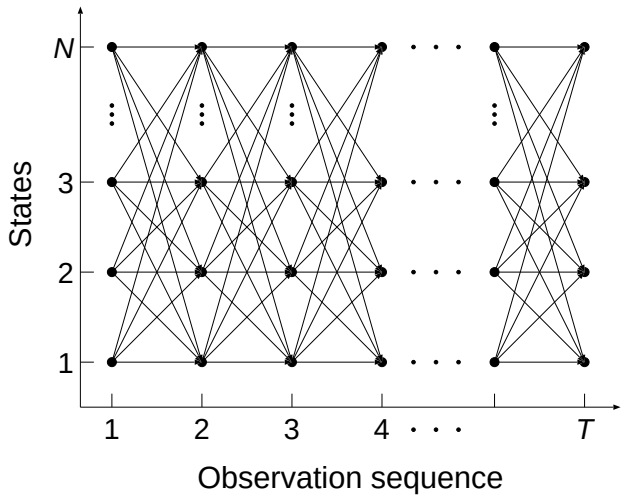


Figure 2.2: Implementation of the computation using forward-backward algorithm in terms of a trellis of observations and states.

3. Termination

$$p\left(\boldsymbol{o} \mid \Lambda\right) = \sum_{i=1}^{N} \beta_1(i). \tag{2.16}$$

The forward and backward variables can be used to compute the total output probability as follows:

$$p\left(\boldsymbol{o} \mid \Lambda\right) = \sum_{j=1}^{N} \alpha_t(j)\beta_t(j). \quad 1 \leq t \leq T \tag{2.17}$$

The forward-backward algorithm is based on the trellis structure shown in Figure 2.2. In this figure, the x-axis and y-axis represent observations and states of an HMM, respectively. On the trellis, all possible state sequences will re-merge into these N nodes no matter how long the observation sequence. In the case of the forward algorithm, at time $t = 1$, we need to calculate values of $\alpha_1(i)$, $1 \leq i \leq N$. At times $t = 2, 3, \ldots, T$, we need only calculate values of $\alpha_t(j)$, $1 \leq j \leq N$, where each calculation involves only the N previous values of $\alpha_{t-1}(i)$ because each of the N grid points can be reached from only the N grid points at the previous time slot. As a result, the forward-backward algorithm can reduce order of probability calculation.

2.3 Searching optimal state sequence

The single optimal state sequence $\hat{\boldsymbol{q}} = \{\hat{q_1}, \hat{q_2}, \ldots, \hat{q_T}\}$ for a given observation vector sequence $\boldsymbol{o} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$ is useful for various applications (e.g., decoding, initializing HMM parameters). By using a manner similar to the forward algorithm, which is often referred to as the Viterbi algorithm [19], we can obtain the optimal state sequence $\hat{\boldsymbol{q}}$. Let $\delta_t(i)$ be the likelihood of the most likely state sequence ending in the i-th state at time t

$$\delta_t(i) = \max_{q_1,\ldots,q_{t-1}} p\left(q_1, \ldots, q_{t-1}, q_t = i, \boldsymbol{o}_1, \ldots, \boldsymbol{o}_t \mid \Lambda\right), \tag{2.18}$$

and $\psi_t(i)$ be the array to keep track. The complete procedure for finding the optimal state sequence can be written as follows:

1. Initialization

$$\delta_1(i) = \pi_i b_i(\boldsymbol{o}_1), \qquad 1 \le i \le N \tag{2.19}$$
$$\psi_1(i) = 0, \qquad 1 \le i \le N \tag{2.20}$$

2. Recursion

$$\delta_t(j) = \max_i \left[\delta_{t-1}(i)\, a_{ij}\right] b_j(\boldsymbol{o}_t), \qquad \begin{array}{l} 1 \le i \le N \\ t = 2, 3, \ldots, T \end{array} \tag{2.21}$$
$$\psi_t(j) = \arg\max_i \left[\delta_{t-1}(i)\, a_{ij}\right], \qquad \begin{array}{l} 1 \le i \le N \\ t = 2, 3, \ldots, T \end{array} \tag{2.22}$$

3. Termination

$$\hat{P} = \max_i \left[\delta_T(i)\right], \tag{2.23}$$
$$\hat{q_T} = \arg\max_i \left[\delta_T(i)\right]. \tag{2.24}$$

4. Back tracking

$$\hat{q_t} = \psi_{t+1}\left(\hat{q_{t+1}}\right), \quad t = T-1, \ldots, 1. \tag{2.25}$$

It should be noted that the Viterbi algorithm is similar to the forward calculation of Eqs. (2.10)–(2.12). The major difference is the maximization in Eq. (2.21) over previous states, which is used in place of the summation in Eq. (2.11). It also should be clear that a trellis structure efficiently implements the computation of the Viterbi procedure.

2.4 Maximum likelihood estimation of HMM parameters

There is no known method to analytically obtain the model parameter set based on the maximum likelihood (ML) criterion to obtain Λ which maximizes its likelihood $p\left(\boldsymbol{o} \mid \Lambda\right)$ for a given observation sequence $\boldsymbol{o}$, in a closed form. Since this problem is a high dimensional nonlinear optimization problem, and there will be a number of local maxima, it is difficult to obtain Λ which globally maximizes $p\left(\boldsymbol{o} \mid \Lambda\right)$. However, the model parameter set Λ locally maximizes $p\left(\boldsymbol{o} \mid \Lambda\right)$ can be obtained using an iterative procedure such as the expectation-maximization (EM) algorithm [20], and the obtained parameter set will be appropriately estimated if a good initial estimate is provided.

In the following, the EM algorithm for the CD-HMM is described. The algorithm for the HMM with discrete output distributions can also be derived in a straightforward manner.

2.4.1 $\mathcal{Q}$-function

In the EM algorithm, an auxiliary function $\mathcal{Q}(\Lambda, \hat{\Lambda})$ of the current parameter set Λ and the new parameter set $\hat{\Lambda}$ is defined as follows:

$$\mathcal{Q}(\Lambda, \hat{\Lambda}) = \sum_{\text{all } \boldsymbol{q}} p(\boldsymbol{q} \mid \boldsymbol{o}, \Lambda) \log p(\boldsymbol{o}, \boldsymbol{q} \mid \hat{\Lambda}). \tag{2.26}$$

Each mixture of Gaussian components is decomposed into a substate, and $\boldsymbol{q}$ is redefined as a substate sequence,

$$\boldsymbol{q} = \left\{(q_1, s_1), (q_2, s_2), \ldots, (q_T, s_T)\right\}, \tag{2.27}$$

where (q_t, s_t) represents being in the s_t-th substate (Gaussian component) of the q_t-th state at time t.

At each iteration of the procedure, the current parameter set Λ is replaced by the new parameter set $\hat{\Lambda}$ which maximizes $\mathcal{Q}(\Lambda, \hat{\Lambda})$. This iterative procedure can be proved to

increase likelihood $p\left(\boldsymbol{o} \mid \Lambda\right)$ monotonically and converge to a certain critical point, since it can be proved that the $\mathcal{Q}$-function satisfies the following theorems:

- Theorem 1

$$\mathcal{Q}(\Lambda, \hat{\Lambda}) \geq \mathcal{Q}(\Lambda, \Lambda) \;\Rightarrow\; p(\boldsymbol{o} \mid \hat{\Lambda}) \geq p(\boldsymbol{o} \mid \Lambda) \tag{2.28}$$

- Theorem 2
  The auxiliary function $\mathcal{Q}(\Lambda, \hat{\Lambda})$ has the unique global maximum as a function of Λ, and this maximum is the one and only critical point.

- Theorem 3
  A parameter set Λ is a critical point of the likelihood $p(\boldsymbol{o} \mid \Lambda)$ if and only if it is a critical point of the $\mathcal{Q}$-function.

2.4.2 Maximization of $\mathcal{Q}$-function

According to Eqs. (2.2) and (2.7), $\log p\left(\boldsymbol{o}, \boldsymbol{q} \mid \Lambda\right)$ can be written as

$$\log p\left(\boldsymbol{o}, \boldsymbol{q} \mid \Lambda\right) = \log p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda\right) + \log P\left(\boldsymbol{q} \mid \Lambda\right), \tag{2.29}$$

$$\log p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda\right) = \sum_{t=1}^{T} \log \mathcal{N}\left(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{q_t s_t}, \boldsymbol{\Sigma}_{q_t s_t}\right), \tag{2.30}$$

$$\log P\left(\boldsymbol{q} \mid \Lambda\right) = \log \pi_{q_1} + \sum_{t=2}^{T} \log a_{q_{t-1} q_t} + \sum_{t=1}^{T} \log w_{q_t s_t}. \tag{2.31}$$

Hence, $\mathcal{Q}$-function (Eq. (2.26)) can be rewritten as

$$\begin{aligned}
\mathcal{Q}(\Lambda, \hat{\Lambda}) = & \sum_{i=1}^{N} p\left(\boldsymbol{o}, q_1 = i \mid \Lambda\right) \cdot \log \pi_i \\
& + \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{t=1}^{T-1} p\left(\boldsymbol{o}, q_t = i, q_{t+1} = j\right) \cdot \log a_{ij} \\
& + \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} p\left(\boldsymbol{o}, q_t = i, s_t = m \mid \Lambda\right) \cdot \log w_{im} \\
& + \sum_{i=1}^{N} \sum_{m=1}^{M} \sum_{t=1}^{T} p\left(\boldsymbol{o}, q_t = i, s_t = m \mid \Lambda\right) \cdot \log \mathcal{N}\left(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{im}, \boldsymbol{\Sigma}_{im}\right).
\end{aligned} \tag{2.32}$$

The parameter set Λ which maximizes the above equation subject to the stochastic constraints

$$\sum_{i=1}^{N} \pi_i = 1, \tag{2.33}$$

$$\sum_{j=1}^{N} a_{ij} = 1, \quad 1 \leq i \leq N \tag{2.34}$$

$$\sum_{m=1}^{M} w_{im} = 1, \quad 1 \leq i \leq N \tag{2.35}$$

can be derived by Lagrange multipliers or differential calculus as follows [21]:

$$\pi_i = \gamma_1(i), \qquad 1 \leq i \leq N \tag{2.36}$$

$$a_{ij} = \frac{\sum_{t=2}^{T} \xi_{t-1}(i,j)}{\sum_{t=2}^{T} \gamma_{t-1}(i)}, \qquad \begin{matrix} 1 \leq i \leq N \\ 1 \leq j \leq N \end{matrix} \tag{2.37}$$

$$w_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m)}{\sum_{t=1}^{T} \gamma_t(i)}, \qquad \begin{matrix} 1 \leq i \leq N \\ 1 \leq m \leq M \end{matrix} \tag{2.38}$$

$$\boldsymbol{\mu}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m) \cdot \boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(i,m)}, \qquad \begin{matrix} 1 \leq i \leq N \\ 1 \leq m \leq M \end{matrix} \tag{2.39}$$

$$\boldsymbol{\Sigma}_{im} = \frac{\sum_{t=1}^{T} \gamma_t(i,m) \cdot (\boldsymbol{o}_t - \boldsymbol{\mu}_{im}) (\boldsymbol{o}_t - \boldsymbol{\mu}_{im})^\top}{\sum_{t=1}^{T} \gamma_t(i,m)}, \qquad \begin{matrix} 1 \leq i \leq N \\ 1 \leq m \leq M \end{matrix} \tag{2.40}$$

where $\gamma_t(i)$, $\gamma_t(i,m)$, and $\xi_t(i,j)$ are the probability of being in the j-th state at time t,

the probability of being in the m-th substate of the i-th state at time t, and the probability of being in the i-th state at time t and j-th state at time $t+1$, respectively, that is

$$\begin{aligned}
\gamma_t(i) &= p(\boldsymbol{o}, q_t = i \mid \Lambda) \\
&= \frac{\alpha_t(i)\beta(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}, && \begin{array}{l} 1 \le i \le N \\ t = 1, \ldots, T \end{array} \qquad (2.41) \\
\gamma_t(i,m) &= p(\boldsymbol{o}, q_t = i, s_t = m \mid \Lambda) \\
&= \frac{\alpha_t(i)\beta(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \cdot \frac{w_{im}\mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{im}, \boldsymbol{\Sigma}_{im})}{\sum_{k=1}^{M} w_{ik}\mathcal{N}(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{ik}, \boldsymbol{\Sigma}_{ik})}, && \begin{array}{l} 1 \le i \le N \\ 1 \le m \le M \\ t = 1, \ldots, T \end{array} \qquad (2.42) \\
\xi_t(i,j) &= p(\boldsymbol{o}, q_t = i, q_{t+1} = j \mid \Lambda) \\
&= \frac{\alpha_t(i) a_{ij} b_j(\boldsymbol{o}_{t+1}) \beta_{t+1}(j)}{\sum_{l=1}^{N}\sum_{n=1}^{N} \alpha_t(l) a_{ln} b_n(\boldsymbol{o}_{t+1}) \beta_{t+1}(n)}. && \begin{array}{l} 1 \le i \le N \\ t = 1, \ldots, T \end{array} \qquad (2.43)
\end{aligned}$$

2.5 Summary

In this chapter, the basic theories of the hidden Markov models (HMMs), its algorithm for calculating the output probability (forward-backward algorithm), searching the optimal state sequence (Viterbi algorithm), and estimating its parameters (EM algorithm) are described. Following chapters show the HMMs for acoustic modeling in speech recognition.

Chapter 3

HMM-based speech recognition

Most of the current speech recognition systems uses HMMs as its acoustic model. In this chapter, statistical speech recognition framework based on the HMM is described. General speech recognition systems may be divided into five basic blocks: the front-end, acoustic models, language models, lexicon and search algorithm. These blocks are introduced in more detail in the following sections.

3.1 Statistical speech recognition

The goal of large vocabulary continuous speech recognition (LVCSR) systems is to take an acoustic waveform as its input and generate a transcription of the words being uttered. First, the speech waveform is recorded and sampled by a digital device. Next, processor converts the sampled waveform into an observation vector sequence $\boldsymbol{o} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\}$ by removing redundant or unimportant informations such as noises. There is a larage amount of variability in observation vector sequences even if the same words were uttered by the same speaker. Therefore, a statistical approach is adopted to map the observation vector sequence into the most likely word sequence. The speech recognition system usually choose the word sequence, $\boldsymbol{w} = \{w_1, \ldots, w_L\}$, with the maximum a posteriori (MAP) probability given the observation sequence as follows:

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} P\left(\boldsymbol{w} \mid \boldsymbol{o}\right) \tag{3.1}$$

Recently, discriminative models such as maximum entropy Markov models (MEMMs) [22] or conditional random fields (CRFs) [23] have been applied for modeling $P\left(\boldsymbol{w} \mid \boldsymbol{o}\right)$ directly [24, 25]. However, applying the discriminative models for LVCSR is still difficult

due to variabilities of the observation vector sequences and the vast number of possible word sequences. Therefore, most of the current speech recognition systems uses generative models rather than the discriminative ones. By using Bayes' rule, Eq. (3.1) can be written as

$$P(\boldsymbol{w} \mid \boldsymbol{o}) = \frac{p(\boldsymbol{o} \mid \boldsymbol{w}) P(\boldsymbol{w})}{p(\boldsymbol{o})}. \tag{3.2}$$

Since $p(\boldsymbol{o})$ is independent of the word sequence $\boldsymbol{w}$, the MAP decoding rule of Eq. (3.1) is

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} \; p(\boldsymbol{o} \mid \boldsymbol{w}) P(\boldsymbol{w}). \tag{3.3}$$

A general statistical speech recognition system may be described by the formulation in Eq. (3.3). The system consists of five main blocks: the front-end, acoustic models, language models, pronunciation lexicon and search algorithm.

The first term in Eq. (3.3), $p(\boldsymbol{o} \mid \boldsymbol{w})$, corresponds to the acoustic model, as it estimates the probability of an observation vector sequence $\boldsymbol{o}$, conditioned on the word sequence $\boldsymbol{w}$. For large vocabulary continuous speech recognition, the way of $p(\boldsymbol{o} \mid \boldsymbol{w})$ computation is to build statistical models for sub-word speech units, build up word models from these sub-word speech units using a pronunciation lexicon, and then postulate word sequences and evaluate the acoustic model probabilities of concatenated word models. It is possible to use any kind of models for $p(\boldsymbol{o} \mid \boldsymbol{w})$. Currently, context-dependent sub-word HMMs are used for most of speech recognition systems as its acoustic model.

The second term in Eq. (3.3), $P(\boldsymbol{w})$, corresponds to the language model, as it describes the probability associated with a postulated sequence of words. Generally language models are represented in a finite state network so as to be integrated into the acoustic model in a straightforward manner.

The final block, the search algorithm, implements the maximization in Eq. (3.3).

3.2 Front-ends

Comparing the sampled acoustic waveforms is difficult due to varying speaker and acoustic characteristics. However, the spectral shape of the speech signal have most of the important information [26]. Front-end of speech recognition systems generate observation vector sequences which represent the short-term spectrum of the speech signal. There

are many techniques for parameterizing speech spectra, i.e., linear prediction coefficients (LPC) [27,28], line spectral pair (LSP), cepstrum [29], mel-cepstrum [30], and so on. Mel filterbank cepstral coefficients (MFCC) [31] or perceptual linear prediction (PLP) [32] is generally used in most of the current speech recognition systems. In all cases the speech signal is assumed to be quasi-stationary so that it can be decided into short frames.In each frame period a new parameterized short-time spectra vector is produced by analyzing a speech segment. In a final step, delta and delta-delta coefficients are appended to the acoustic vector [33–36]. The delta and delta-delta coefficients are usually calculated as regression coefficients from their neighboring static features as follows:

$$\Delta \boldsymbol{c}_t = \sum_{\tau=-L_-^{(1)}}^{L_+^{(1)}} w^{(1)}(\tau)\boldsymbol{c}_{t+\tau}, \quad \Delta^2 \boldsymbol{c}_t = \sum_{\tau=-L_-^{(2)}}^{L_+^{(2)}} w^{(2)}(\tau)\boldsymbol{c}_{t+\tau}, \tag{3.4}$$

where $\boldsymbol{c}_t$, $\Delta \boldsymbol{c}_t$, and $\Delta^2 \boldsymbol{c}_t$ are static, delta, and delta-delta coefficients at time t, respectively, and $\left\{w^{(d)}(\tau)\right\}_{d=1,2\ \tau=-L_-^{(d)},\ldots,L_+^{(d)}}$ are regression window coefficients to calculate the d-th order dynamic feature. As a result, the observation vector at time t, $\boldsymbol{o}_t$, consists of static and dynamic features as

$$\boldsymbol{o}_t = \left[\boldsymbol{c}_t^\top, \Delta \boldsymbol{c}_t^\top, \Delta^2 \boldsymbol{c}_t^\top\right]^\top. \tag{3.5}$$

3.3 HMM-based acoustic modeling

The HMMs are used to provide the estimates of $p\left(\boldsymbol{o} \mid \boldsymbol{w}\right)$ in the speech recognition systems. For isolated word recognition with sufficient training data, an HMM can be trained for each word. However, for LVCSR tasks, it is unlikely that there are enough training examples of each word in the dictionary. Therefore, sub-word units such as phone or syllable is used. An HMM is generally trained for each phone. The HMMs corresponding to the phone sequence may then be concatenated to form a composite model representing words and sentences.

When the HMMs are trained for the set of phones, it is referred to as a monophone or context-independent system. However, there is a large amount of variation between realizations of the same phone depending on the previous and next phones. Triphones which take the previous and next phones into account are commonly used as context-dependent phones. The number of states and model parameters of a triphone system is significantly higher than a monophone system. However, it is unlikely that sufficient training data is

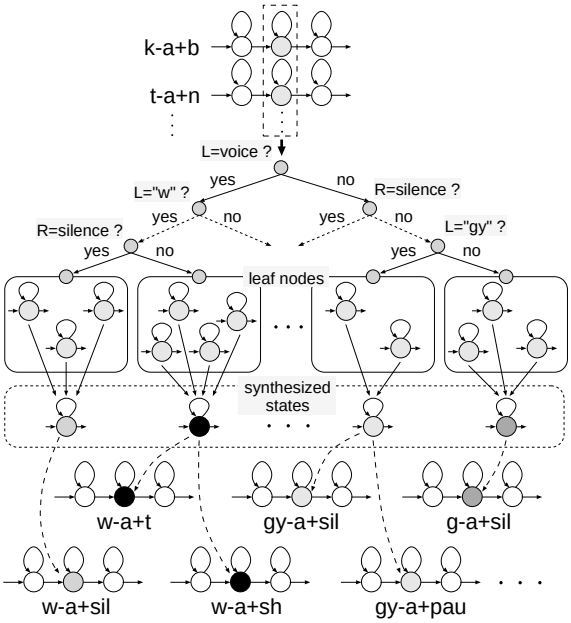


Figure 3.1: Example of a phonetic decision tree for triphone models.

available for parameter estimation. To avoid this problem, the state output probability distributions are generally shared.

A phonetic decision tree [37–39] is generally used to construct state tying structure in context-dependent systems (Figure 3.1). First, all phones are pooled in the root node. Next, the state clusters are split based on contextual questions. When the number of training data per state falls below a threshold, the splitting will terminate. A disadvantage of decision tree-based state clustering is that the splits maximize the likelihood of the training data locally [40, 41].

3.4 Word N-gram-based language modeling

The language model provides $P(\boldsymbol{w})$ in the speech recognition systems. Using chain rule, this can be expressed as

$$P(\boldsymbol{w}) = \prod_{l=1}^{L} P(w_l \mid w_{l-1}, \ldots, w_1). \tag{3.6}$$

To reduce the number of parameters, different histories can be divided into equivalence

class using a function $h(w_{l-1}, \ldots, w_1)$. In general, equivalence classes are defined by truncating the history to $N-1$ words. These word N-gram language models are defined as

$$P(\boldsymbol{w}) = \prod_{l=1}^{L} P(w_l \mid w_{l-1}, \ldots, w_{l-N+1}). \tag{3.7}$$

Standard values are $N = 2, 3$ which are called bi-gram or tri-gram models, respectively. The N-grams are estimated by counting relative frequencies from text corpus. For a vocabulary of V words, there are still V^N N-gram models. Word sequences can be assigned a zero probability for given a finite training data. Many smoothing technique such as discounting, backing off, and deleted interpolation have been proposed [42].

In the speech recognition systems, there is often a mismatch between the acoustic and language model. Dynamic ranges is different between the discrete probability, $P(\boldsymbol{w})$, estimated from a text corpus and the acoustic likelihood, $p(\boldsymbol{o} \mid \boldsymbol{w})$, obtained from high dimensional observation densities. For this mismatch, the language model probability is generally increased by a constant called the grammar scale factor. The speech recognition system also tend to output short words result in many insertion errors. To compensate this problem, an insertion penalty which reduce the total score $p(\boldsymbol{o} \mid \boldsymbol{w}) P(\boldsymbol{w})$ depending on the number of hypothesized words in the sequence is generally used. By taking these modifications into account in Eq. (3.3), a practical speech recognition system uses

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} \Big[\log\{p(\boldsymbol{o} \mid \boldsymbol{w})\} + \alpha \log\{P(\boldsymbol{w}) + \beta L\} \Big] \tag{3.8}$$

where α, β, and L are the grammar scale factor, the insertion penalty, and the total number of words, respectively. The α and β are empirically set.

3.5 Pronunciation lexicon

Each word is defined by a pronunciation obtained from a dictionary. The word HMM is the concatenation of the relevant sequence of sub-word HMMs. The lexicon is stored as a tree for computational efficiency. A tree-based lexicon have been used in various speech recognition system. Tree-based lexicon allows pronunciations with similar heads to share memory when being evaluated. Therefore, different pronunciations of the same word are stored as separate lexical items. The disadvantage of using the tree-based lexicon is that it is not an efficient approach to represent multiple pronunciations of the same word.

3.6 Search algorithms

To determine the word sequene yielding maximum combined probability from the acoustic and language model, the following problems must be resolved.

1. The number of words in given utterance is unknown.
2. Word boundaries in given utterance are also unknown.
3. The word boundaries are often fuzzy.
4. For a set of V word-reference patterns and L words in the utterance, there are $V\ L$ possible combinations of composite matching patterns.

To solve these problems, efficient search algorithm have been proposed. Most of these algorithms can categorized into two basic classes: Viterbi decoding [43] and stack decoding [44].

3.7 Summary

In this chapter, the statistical speech recognition framework and its main modules, front-ends, acoustic modeling, language modeling, and search algorithm, are described. Following chapter show the HMMs for acoustic modeling in speech synthesis.

Chapter 4

HMM-based speech synthesis

In the previous chapter, HMM-based speech recognition system was described. In this chapter, statistical speech synthesis framework and the HMM-based speech synthesis system are described.

4.1 Statistical speech synthesis

Text-to-speech synthesis system can be viewed as an inverse procedure of speech recognition system. The goal of a text-to-speech system is acoustic speech waveform generation from a word sequence. In general, given word sequence $\boldsymbol{w}$ is processed by a text analysis module. In this part, contextual factors (e.g., accent, lexical stress, part-of-speech, phrase boundary, etc.) are estimated. Next, a speech waveform is generated by a speech synthesis module.

The majority of state-of-the-art speech synthesis systems is trained by using a large amount of speech data. In general, this type of system is called as a corpus-based speech synthesis system [45]. Compared with the previous speech synthesis systems, corpus-based one especially improve the naturalness of synthesized speech.

One of the major approaches in the corpus-based speech synthesis is unit selection based one [46–48]. In this system, the speech waveform is segmented into the small units, phone, di-phone, syllable, etc.. Next, a unit sequence with minimum target and concatenation costs is selected [47] and connected.

Another major approach is statistical speech synthesis, such as HMM-based one [5]. This system generates speech parameter sequence $\boldsymbol{o} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$ with the maximum a posteriori (MAP) probability given the sub-word sequence $\boldsymbol{u}$ as follows:

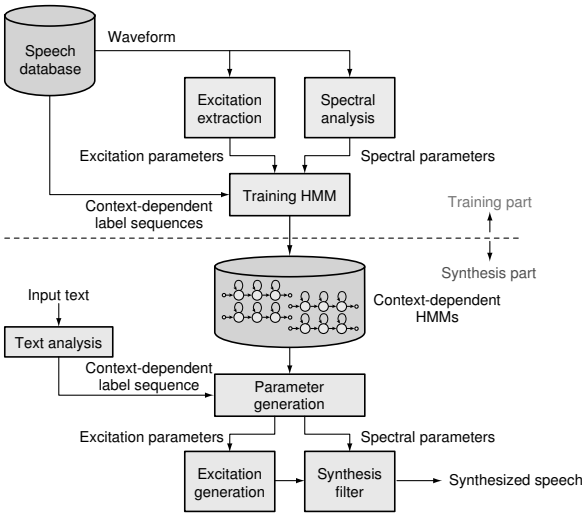


Figure 4.1: An overview of a typical HMM-based speech synthesis system.

$$\hat{\boldsymbol{o}} = \arg\max_{\boldsymbol{o}} P\left(\boldsymbol{o} \mid \boldsymbol{u}\right). \tag{4.1}$$

The term in Eq. (4.1) has the same form to the first term in Eq. (3.3). In speech recognition system, Bayes' rule is required to use generative models. On the other hand, generative models can directly be applied in speech synthesis system. The HMM is the most popular generative models.

4.2 HMM-based speech synthesis

4.2.1 Overview

Figure 4.1 shows the HMM-based speech synthesis system [5]. It consists of the training and synthesis part. In the training part, spectrum and excitation parameters are extracted from a speech database. These parameters are modeled by context-dependent HMMs. State duration models are also estimated. In the synthesis part, a sentence HMM is constructed by concatenating the context-dependent HMMs fro a given text to be synthesized. In synthesis part, the sequences of spectrum and excitation parameters are generated from the sentence HMM using speech parameter generation algorithm [49–51]. Finally, speech waveform is synthesized from a synthesis filter module. One of the advantage is that voice qualities of synthesized speech can be modified by transforming HMM parameters. It has been shown that its voice characteristics can be modified by speaker adaptation [52],

speaker interpolation [53], or eigenvoice technique [54].

4.2.2 Speech parameter generation algorithm

Problem

For a sentence HMM $\Lambda_{\boldsymbol{u}}$ corresponding to a given sub-word sequence $\boldsymbol{u}$, the speech synthesis problem is to obtain an output vector sequence consisted of spectral and excitation parameters.

$$\boldsymbol{o} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\} \tag{4.2}$$

which maximizes its posterior probability with respect to $\boldsymbol{o}$, that is

$$\begin{aligned}
\hat{\boldsymbol{o}} &= \arg\max_{\boldsymbol{o}} p\left(\boldsymbol{o} \mid \Lambda_{\boldsymbol{u}}\right) \\
&= \arg\max_{\boldsymbol{o}} \sum_{\text{all } \boldsymbol{q}} p\left(\boldsymbol{o}, \boldsymbol{q} \mid \Lambda_{\boldsymbol{u}}\right) \\
&= \arg\max_{\boldsymbol{o}} \sum_{\text{all } \boldsymbol{q}} p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda_{\boldsymbol{u}}\right) P\left(\boldsymbol{q} \mid \Lambda_{\boldsymbol{u}}\right) \qquad (4.3) \\
\boldsymbol{q} &= \{(q_1, s_1), (q_2, s_2), \ldots, (q_T, s_T)\} \qquad (4.4)
\end{aligned}$$

where, $\boldsymbol{q}$ and (q_t, s_t) represent a substate sequence and the s_t-th substate of the q_t-th state, respectively. This problem is approximated by a Viterbi approximation, because there is not method to analytically obtain $\boldsymbol{o}$ which maximizes $p\left(\boldsymbol{o} \mid \Lambda_{\boldsymbol{u}}\right)$ in a closed form. As a result, this maximization problem can be separated into two stages: finding the best substate sequence $\hat{\boldsymbol{q}}$ for given $\Lambda_{\boldsymbol{u}}$ and obtaining $\boldsymbol{o}$ which maximizes $p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda_{\boldsymbol{u}}\right)$ with respect to $\boldsymbol{o}$, i.e.,

$$\hat{\boldsymbol{q}} = \arg\max_{\boldsymbol{q}} P\left(\boldsymbol{q} \mid \Lambda_{\boldsymbol{u}}\right), \tag{4.5}$$

$$\hat{\boldsymbol{o}} = \arg\max_{\boldsymbol{o}} p\left(\boldsymbol{o} \mid \hat{\boldsymbol{q}}, \Lambda_{\boldsymbol{u}}\right). \tag{4.6}$$

The optimization of Eq. (4.5) is performed using explicit state duration models [55] in the HMM-based speech synthesis system. If the output vector $\boldsymbol{o}_t$ is independent from previous and next frames, the output vector sequence $\boldsymbol{o}$ which maximize $p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda_{\boldsymbol{u}}\right)$

is obtained as a sequence of mean vectors of substates. This causes discontinuity in the output vector sequence at transitions of substates. To avoid this problem, dynamic features have benn introduced. We assume that the output vector $\boldsymbol{o}_t$ consists of a static feature vector

$$\boldsymbol{c}_t = [c_t(1), \ldots, c_t(K)]^\top \tag{4.7}$$

and its dynamic features, that is

$$\boldsymbol{o}_t = \left[\boldsymbol{c}_t^\top, \Delta\boldsymbol{c}_t^\top, \Delta^2\boldsymbol{c}_t^\top\right]^\top, \tag{4.8}$$

where $\Delta\boldsymbol{c}_t$ and $\Delta^2\boldsymbol{c}_t$ are delta and delta-delta coefficients, respectively. They are calculated as follows:

$$\Delta\boldsymbol{c}_t = \sum_{\tau=-L_-^{(1)}}^{L_+^{(1)}} w^{(1)}(\tau)\boldsymbol{c}_{t+\tau}, \tag{4.9}$$

$$\Delta^2\boldsymbol{c}_t = \sum_{\tau=-L_-^{(2)}}^{L_+^{(2)}} w^{(2)}(\tau)\boldsymbol{c}_{t+\tau}. \tag{4.10}$$

Solution for the Problem

First, the output vector sequence $\boldsymbol{o}$ and the static feature vector sequence $\boldsymbol{c}$ can be rewritten as follows:

$$\boldsymbol{o} = \left[\boldsymbol{o}_1^\top, \boldsymbol{o}_2^\top, \ldots, \boldsymbol{o}_T^\top\right]^\top, \tag{4.11}$$

$$\boldsymbol{c} = \left[\boldsymbol{c}_1^\top, \boldsymbol{c}_2^\top, \ldots, \boldsymbol{c}_T^\top\right]^\top. \tag{4.12}$$

Then, the relationship between $\boldsymbol{c}$ and $\boldsymbol{o}$ can be expressed in a matrix form (Figure 4.2) as follows:

$$\boldsymbol{o} = \boldsymbol{W}\boldsymbol{c}, \tag{4.13}$$

where, $\boldsymbol{W}$ is a regression window matrix given by

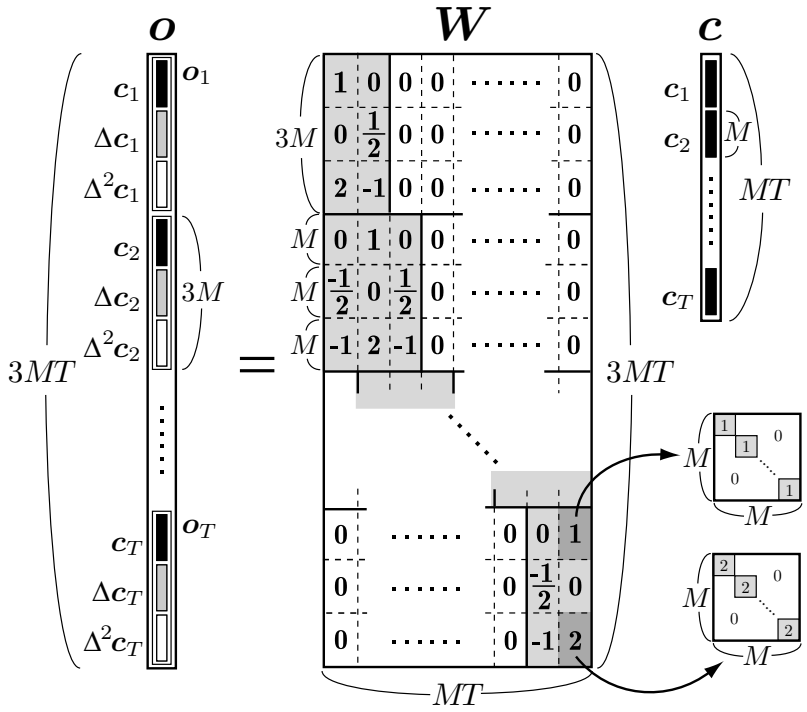


Figure 4.2: An example of the relationship between the static feature vector sequence $\boldsymbol{c}$ and the speech parameter vector sequence $\boldsymbol{o}$ in a matrix form (the dynamic features are calculated using $L_{-}^{(1)} = L_{+}^{(1)} = L_{-}^{(2)} = L_{+}^{(2)} = 1$, $w^{(1)}(-1) = -0.5$, $w^{(1)}(0) = 0.0$, $w^{(1)}(1) = 0.5$, $w^{(2)}(-1) = 1.0$, $w^{(2)}(0) = -2.0$, $w^{(2)}(1) = 1.0$).

$$\boldsymbol{W} = [\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_T]^\top \otimes \boldsymbol{I}_{M\times M}, \tag{4.14}$$

$$\boldsymbol{W}_t = \left[\boldsymbol{w}_t^{(0)}, \boldsymbol{w}_t^{(1)}, \boldsymbol{w}_t^{(2)}\right], \tag{4.15}$$

$$\boldsymbol{w}_t^{(0)} = \Big[\underbrace{0,\ldots,0}_{t-1}, 1, \underbrace{0,\ldots,0}_{T-t}\Big]^\top, \tag{4.16}$$

$$\boldsymbol{w}_t^{(1)} = \Big[\underbrace{0,\ldots,0}_{t-L_{-}^{(1)}-1}, w^{(1)}(-L_{-}^{(1)}), \ldots, w^{(1)}(0), \ldots, w^{(1)}(L_{+}^{(1)}), \underbrace{0,\ldots,0}_{T-\left(t+L_{+}^{(1)}\right)}\Big]^\top, \tag{4.17}$$

$$\boldsymbol{w}_t^{(2)} = \Big[\underbrace{0,\ldots,0}_{t-L_{-}^{(2)}-1}, w^{(2)}(-L_{-}^{(2)}), \ldots, w^{(2)}(0), \ldots, w^{(2)}(L_{+}^{(2)}), \underbrace{0,\ldots,0}_{T-\left(t+L_{+}^{(2)}\right)}\Big]^\top, \tag{4.18}$$

The output probability of $\boldsymbol{o}$ conditioned on $\boldsymbol{q}$ is calculated by multiplying the output probabilities of entire observation vectors,

$$p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda_{\boldsymbol{u}}\right) = \prod_{t=1}^{T} \mathcal{N}\left(\boldsymbol{o}_t \mid \boldsymbol{\mu}_{q_t s_t}, \boldsymbol{\Sigma}_{q_t s_t}\right), \tag{4.19}$$

where, $\boldsymbol{\mu}_{q_t s_t}$ and $\boldsymbol{\Sigma}_{q_t s_t}$ are the $3K \times 1$ mean vector and $3K \times 3K$ covariance matrix, respectively. Eq. (4.19) can be rewritten as an output probability of $\boldsymbol{o}$ from a single Gaussian component, that is

$$p\left(\boldsymbol{o} \mid \boldsymbol{q}, \Lambda_{\boldsymbol{u}}\right) = \mathcal{N}\left(\boldsymbol{o} \mid \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right), \tag{4.20}$$

where, $\boldsymbol{\mu}_{\boldsymbol{q}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{q}}$ are supervector and supermatrix corresponding to entire substate sequence $\boldsymbol{q}$, that is

$$\boldsymbol{\Sigma}_{\boldsymbol{q}} = \operatorname{diag}\left[\boldsymbol{\Sigma}_{q_1 s_1}, \boldsymbol{\Sigma}_{q_2 s_2}, \ldots, \boldsymbol{\Sigma}_{q_t s_t}\right], \tag{4.21}$$

$$\boldsymbol{\mu}_{\boldsymbol{q}} = \left[\boldsymbol{\mu}_{q_1 s_1}^{\top}, \boldsymbol{\mu}_{q_2 s_2}^{\top}, \ldots, \boldsymbol{\mu}_{q_t s_t}^{\top}\right]^{\top}. \tag{4.22}$$

Therefore, the logarithm of Eq. (4.19) can be written as

$$\log \mathcal{N}\left(\boldsymbol{o} \mid \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right) = -\frac{1}{2}\left\{3KT \log 2\pi + \log\left|\boldsymbol{\Sigma}_{\boldsymbol{q}}\right| + \left(\boldsymbol{o} - \boldsymbol{\mu}_{\boldsymbol{q}}\right)^{\top} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}\left(\boldsymbol{o} - \boldsymbol{\mu}_{\boldsymbol{q}}\right)\right\}. \tag{4.23}$$

Under the condition in Eq. (4.13), maximizing $\mathcal{N}\left(\boldsymbol{o} \mid \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right)$ with respect to $\boldsymbol{o}$ is equivalent to that with respect to $\boldsymbol{c}$. By setting

$$\frac{\partial \log \mathcal{N}\left(\boldsymbol{o} \mid \boldsymbol{\mu}_{\boldsymbol{q}}, \boldsymbol{\Sigma}_{\boldsymbol{q}}\right)}{\partial \boldsymbol{c}} = \mathbf{0}_{KT}, \tag{4.24}$$

we obtain a set of linear equations

$$\boldsymbol{R}_{\boldsymbol{q}} \boldsymbol{c} = \boldsymbol{r}_{\boldsymbol{q}}, \tag{4.25}$$

where, $\mathbf{0}_{KT}$ is a KT-dimensional zero vector, $\boldsymbol{R}_{\boldsymbol{q}}$ and $\boldsymbol{r}_{\boldsymbol{q}}$ are given as

$$\boldsymbol{R}_{\boldsymbol{q}} = \boldsymbol{W} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{W}, \tag{4.26}$$

$$\boldsymbol{r}_{\boldsymbol{q}} = \boldsymbol{W} \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1} \boldsymbol{\mu}_{\boldsymbol{q}}. \tag{4.27}$$

Since $\boldsymbol{R_q}$ is a $KT \times KT$ matrix, $O(K^3T^3)$ operations are required for solution of Eq. (4.25). Eq. (4.25) can be solved by the Cholesky with $O(K^3L^2T)$ operations by utilizing the special structure of $\boldsymbol{R_q}$. Eq. (4.25) can also be solved by an algorithm derived in [49–51], which can operate in a time-recursive manner [56].

4.3 Summary

In this chapter, a statistical speech synthesis framework and the speech parameter generation algorithm are described. Following chapter will derive a model which integrate state duration probability distribution explicity into the HMM for speech recognition.

Chapter 5

Hidden Semi-Markov Model-Based Speech Recognition

Hidden Markov models (HMMs) (Figure 5.1(a)) have formed the basis of many speech recognition systems since the 1970s. The advantages of using HMMs are: i) They can represent speech as probability distributions. ii) They are robust to temporal structure variations. iii) They provide efficient algorithms for estimating their model parameters. However, a number of limitations of HMMs for modeling speech have been reported [9]. One of their major limitations is in duration modeling. In HMMs, state duration probabilities are implicitly modeled by state transition probabilities; state duration probabilities decrease exponentially with time. Geometric distribution calculated by the state transition probabilities of HMMs would be inappropriate state duration probability distribution representation of the temporal structure of speech.

One of the solutions to this problem is to integrate state duration probability distributions explicitly into the HMM. This model is known as a hidden semi-Markov model (HSMM) [6–8], and is illustrated in Figure 5.1(b). Unlike HMMs, HSMMs have state duration probability distributions. Figure 5.2 shows the state duration probability distributions of an HMM and an HSMM. Geometric distribution in an HMM would be inappropriate representation of the temporal structure of speech. Although the gamma, Poisson and log Gaussian distributions have been applied to state duration modeling in HMM-based speech recognition, in this chapter, we assume that each state duration probability distribution is represented by a Gaussian distribution because there exists a simple clustering algorithm for Gaussian distributions. Although the clustering algorithm of the gamma distributions was reported in [57], we choose Gaussian distributions for simplicity in this chapter.

Although discrete probability distributions can represent any distributions, the lack of

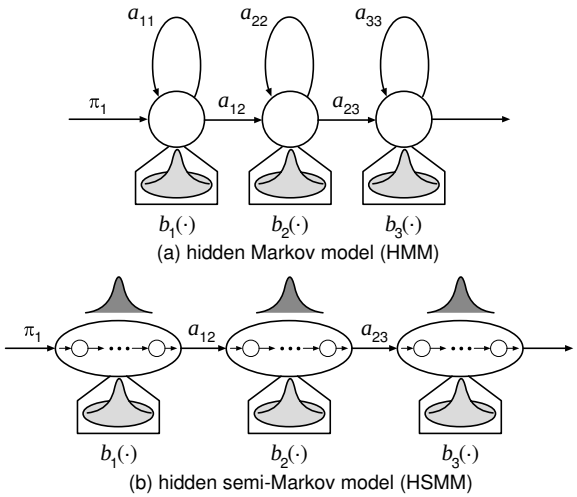


Figure 5.1: Examples of a 3-state left-to-right HMM and an HSMM with no skip.

training data could be an issue. In [8], a smoothing technique is used to prevent over training caused by the lack of training data. On the other hand, the use of continuous probability distributions may remedy such a over training problem. However, it is nessessary to choose an appropriate continuous probability distribution type previously. In this chapter, we use continuous probability distributions because it can avoid additional processing such as smoothing.

A variety of attempts to include explicit duration models in speech recognition systems have been reported [10–12]. However, they are not fully consistent because various approximations are used in both training and decoding:

1) State duration probability distributions were estimated from statistical variables calculated by the forward-backward algorithm of the HMM, not of the HSMM [5].

2) State duration probability estimation utilizes a context-independent model or context-dependent state tying structure of state output probability distributions [11].

3) State duration models were not applied directly in the decoding process. Instead, the N-best hypotheses generated by the HMMs were rescored [12].

We propose a fully consistent HSMM-based speech recognition system to overcome the above approximations. For approximation 1), we simultaneously estimate both state output and duration probability distributions based on the HSMM statistics calculated by the

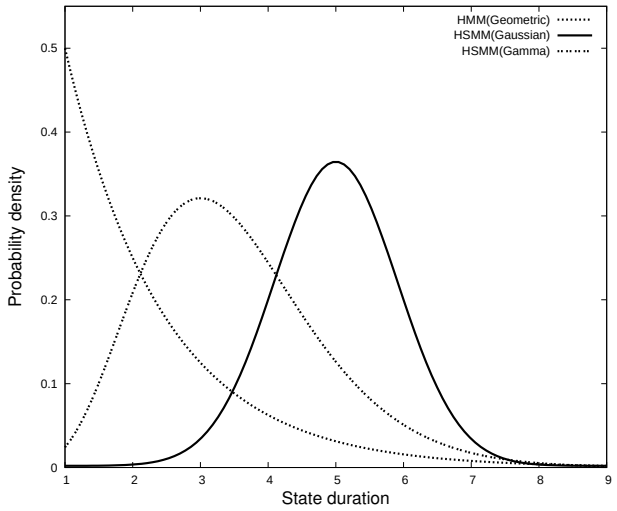


Figure 5.2: State duration probability distributions.

generalized forward-backward algorithm [6]. For approximation 2), state output and duration probability distributions are independently clustered using different phonetic decision trees [5] by a decision-tree-based state clustering technique [39]. For approximation 3), we design an HSMM-native speech decoder based on weighted finite-state transducers (WFSTs) to apply HSMM directly to the input speech.

5.1 A fully consistent HSMM-based speech recognition system

5.1.1 Training algorithms for HSMMs

We derived training algorithms for HSMMs based on the maximum likelihood (ML) criterion [5]. However, there is an inconsistency: state duration probability distributions have not been incorporated into the expectation step of the EM algorithm. In this section, the generalized forward-backward algorithm (expectation step) and parameter re-estimation formulas (maximization step) that are required to avoid the approximation in training [6, 8, 10], are described.

Generalized forward-backward algorithm

The output probability of an observation vector sequence $\boldsymbol{o}$ from an HSMM Λ can be computed efficiently using the generalized forward-backward algorithm. The partial forward probabilities $\alpha_t(\cdot)$ and partial backward probabilities $\beta_t(\cdot)$ are defined as follows:

$$\begin{aligned}
\alpha_0(j) &= \begin{cases} 1 & j = N \\ 0 & \text{otherwise} \end{cases}, && (5.1)\\
\alpha_t(j) &= P(\boldsymbol{o}_1, \ldots, \boldsymbol{o}_t, q_t = j \mid q_{t+1} \neq j, \Lambda) \\
&= \sum_{d=1}^{t} \sum_{\substack{i=1, \\ i \neq j}}^{N} \alpha_{t-d}(i)\, a_{ij} p_j(d) \prod_{s=t-d+1}^{t} b_j(\boldsymbol{o}_s), \quad \begin{matrix} t = 1, 2, \ldots, T \\ 1 \leq j \leq N \end{matrix}, && (5.2)\\
& && (5.3)\\
\beta_{T+1}(i) &= \begin{cases} 1 & i = N \\ 0 & \text{otherwise} \end{cases}, && (5.4)\\
\beta_T(i) &= a_{iN} \beta_{T+1}(N) \quad (1 \leq i \leq N), && (5.5)\\
\beta_t(i) &= P(\boldsymbol{o}_{t+1}, \ldots, \boldsymbol{o}_T, q_t = i \mid q_{t+1} \neq i, \Lambda) \\
&= \sum_{d=1}^{T-t} \sum_{\substack{j=1, \\ j \neq i}}^{N} a_{ij} p_j(d) \prod_{s=t+1}^{t+d} b_j(\boldsymbol{o}_s)\, \beta_{t+d}(j), \quad \begin{matrix} t = T-1, \ldots, 1 \\ 1 \leq i \leq N \end{matrix}, && (5.6)
\end{aligned}$$

where a_{ij}, $b_j(\boldsymbol{o}_t)$, N, and $p_j(d)$, are a state transition probability from the i-th state to the j-th state, a state output probability of an observation vector $\boldsymbol{o}_t$ from the j-th state, the total number of states, and a state duration probability of the j-th state, respectively. From the above equations, the output probability of the observation vector sequence $\boldsymbol{o} = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_T\}$ from the HSMM Λ is given by

$$P(\boldsymbol{o} \mid \Lambda) = \sum_{i=1}^{N} \sum_{\substack{j=1, \\ j \neq i}}^{N} \sum_{d=1}^{t} \alpha_{t-d}(i)\, a_{ij} p_{j(d)} \prod_{s=t-d+1}^{t} b_{j(\boldsymbol{o}_s)} \beta_{t(j)}, \quad t = 1, \ldots, T, \tag{5.7}$$

Parameter re-estimation formulas

In this chapter, we assume that each state output probability $b(\cdot)$ is represented by a mixture of Gaussian distributions. Parameter re-estimation formulas of the mixture weight

w_{jg}, mean vector $\boldsymbol{\mu}_{jg}$ and covariance matrix $\boldsymbol{\Sigma}_{jg}$ of the g-th mixture of the j-th state are given by

$$\overline{w}_{jg} = \frac{\sum_{t=1}^{T}\sum_{d=1}^{t}\gamma_t^d(j,g)}{\sum_{h=1}^{G}\sum_{t=1}^{T}\sum_{d=1}^{t}\gamma_t^d(j,h)}, \tag{5.8}$$

$$\overline{\boldsymbol{\mu}}_{jg} = \frac{\sum_{t=1}^{T}\sum_{d=1}^{t}\zeta_t^d(j,g)}{\sum_{t=1}^{T}\sum_{d=1}^{t}\gamma_t^d(j,g)}, \tag{5.9}$$

$$\overline{\boldsymbol{\Sigma}}_{jg} = \frac{\sum_{t=1}^{T}\sum_{d=1}^{t}\eta_t^d(j,g)}{\sum_{t=1}^{T}\sum_{d=1}^{t}\gamma_t^d(j,g)}, \tag{5.10}$$

respectively, where G is the number of Gaussian distributions, $\gamma_t^d(j,g)$, $\zeta_t^d(j,g)$ and $\eta_t^d(j,g)$ are occupancy probabilities, first, and second order statistics, respectively, given by

$$\gamma_t^d(j,g) = \frac{1}{P(\boldsymbol{o}\mid\Lambda)}\sum_{\substack{i=1,\\ i\neq j}}^{N}\alpha_{t-d}(i)\,a_{ij}p_j(d)\,\beta_t(j) \cdot \sum_{s=t-d+1}^{t} w_{jg}\mathcal{N}(\boldsymbol{o}_s\mid\boldsymbol{\mu}_{jg},\boldsymbol{\Sigma}_{jg})\cdot\prod_{\substack{k=t-d+1,\\ k\neq s}}^{t} b_j(\boldsymbol{o}_k), \tag{5.11}$$

$$\zeta_t^d(j,g) = \frac{1}{P(\boldsymbol{o}\mid\Lambda)}\sum_{\substack{i=1\\ i\neq j}}^{N}\alpha_{t-d}(i)\,a_{ij}p_j(d)\,\beta_t(j) \cdot \sum_{s=t-d+1}^{t} w_{jg}\mathcal{N}(\boldsymbol{o}_s\mid\boldsymbol{\mu}_{jg},\boldsymbol{\Sigma}_{jg})\cdot\prod_{\substack{k=t-d+1,\\ k\neq s}}^{t} b_j(\boldsymbol{o}_k)\,\boldsymbol{o}_s, \tag{5.12}$$

$$\eta_t^d(j,g) = \frac{1}{P(\boldsymbol{o} \mid \Lambda)} \sum_{i=1}^{N} \alpha_{t-d}(i)\, a_{ij} p_j(d)\, \beta_t(j) \cdot \sum_{s=t-d+1}^{t} w_{jg} \mathcal{N}(\boldsymbol{o}_s \mid \boldsymbol{\mu}_{jg}, \boldsymbol{\Sigma}_{jg}) \cdot \prod_{\substack{k=t-d+1, \\ k \neq s}}^{t} b_j(\boldsymbol{o}_k) \left[\boldsymbol{o}_s - \mu_{jg}\right] \left[\boldsymbol{o}_s - \mu_{jg}\right]^{\top}. \tag{5.13}$$

Let us assume that the state duration probability distribution of the j-th state of an HSMM Λ is modeled by a Gaussian distribution with mean ξ_j and variance σ_j^2. The re-estimation formulas of ξ_j and σ_j^2 are derived as follows:

$$p_j(d_j) = \mathcal{N}\left(d_j \mid \xi_j, \sigma_j^2\right), \tag{5.14}$$

$$\overline{\xi}_j = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j) \cdot (t_1 - t_0 + 1)}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j)}, \tag{5.15}$$

$$\overline{\sigma}_j^2 = \frac{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j) \cdot (t_1 - t_0 + 1)^2}{\sum_{t_0=1}^{T} \sum_{t_1=t_0}^{T} \chi_{t_0,t_1}(j)} - \left(\overline{\xi}_j\right)^2, \tag{5.16}$$

where $\chi_{t_0,t_1}(j)$ is the probability of occupying the j-th state of the HSMM Λ from time t_0 to t_1, which can be written as

$$\chi_{t_0,t_1}(j) = \frac{1}{P(\boldsymbol{o} \mid \Lambda)} \sum_{\substack{i=1 \\ i \neq j}}^{N} \alpha_{t_0-1}(i)\, a_{ij} \cdot \prod_{s=t_0}^{t_1} b_j(\boldsymbol{o}_s) \cdot p_j(t_1 - t_0 + 1) \cdot \beta_{t_1}(j). \tag{5.17}$$

5.1.2 Context-dependent duration modeling

There are a number of contextual factors that affect speech parameters. In HMM-based speech recognition systems, context-dependent models such as triphones have been used.

However, if context-dependent models are used, the number of possible models increases exponentially. To avoid this problem, a variety of parameter sharing techniques have been developed [58]. The use of phonetic decision trees is one good solution to this problem [39].

In the conventional HSMM-based speech recognition systems, either the context-independent duration model or the same parameter tying structure as of state output probability distributions was used [11] (Figure 5.3(a)). However, it is generally thought that state output and duration probability distributions have different context-dependencies. We adopted a context-dependent duration modeling technique used in HMM-based speech synthesis [5]. The state duration probabilities of each HSMM are modeled by single multivariate Gaussian distributions whose dimensionality is equal to the number of states of the HSMM. Thus, the Gaussian distribution of the i-th dimension has the mean and variance of the state duration probability distribution for the i-th state of the HSMM. In the proposed system, state output and duration probability distributions are clustered independently by phonetic decision trees [39] (Figure 5.3(b)). Constructed phonetic decision trees represent different context-dependencies for state duration and spectral features.

5.1.3 HSMM-native WFST decoder

Most conventional HSMM-based speech recognition systems have not used state duration models in their decoders [12]. Usually, the N-best hypotheses generated by the HMMs are rescored using the HSMM likelihood. We constructed an HSMM-based speech recognition system using weighted finite-state transducers (WFSTs) to incorporate state duration models into the decoding process.

Finite-state machines have been used in many areas of computational linguistics. These transducers appear as very interesting in speech processing. WFSTs associate weights, such as probabilities, duration, penalties, or any other quantity that accumulates linearly along paths to each pair of input and output symbol sequences. This offers a unified framework representing various models used in speech and language processing [59, 60]. An integrated WFST for speech recognition can be represented as

$$H \circ C \circ L \circ G, \tag{5.18}$$

where H, C, L, and G are WFSTs for a state transitions network, a context-dependent model mapping, a pronunciation lexicon, and a language model, respectively.

The advantages of using WFSTs for speech decoding are

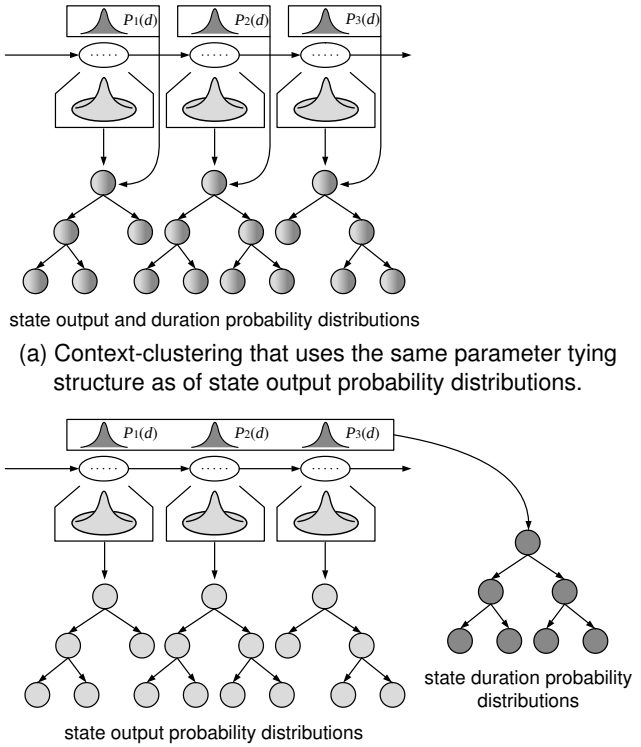


Figure 5.3: Context-clustering for state output and duration probability distributions.

- Individually designed components can be combined.
- Each component can be individually optimized.
- The decoder is easily managed, because the network and the decoder itself are constructed individually.

Furthermore, each component can be replaced easily (Figure 5.4). Using these advantages, we can easily design a speech decoder for HSMMs and fairly compare the performance of different acoustic models based on a common WFST decoding software.

Figure 5.5 shows the state transition of an HMM and an HSMM represented by WFSTs. All arcs of Figure 5.5(a), and Figure 5.5(b) are weighted by state transition probabilities, and state duration probabilities. The maximum state duration in Figure 5.5(b) is limited because we cannot represent infinite duration in the WFST framework. In this chapter, the normalization to satisfy the probability constraint $\sum_d p_j(d) = 1$ is not applied to state duration models because no major difference was found in speech recognition performances

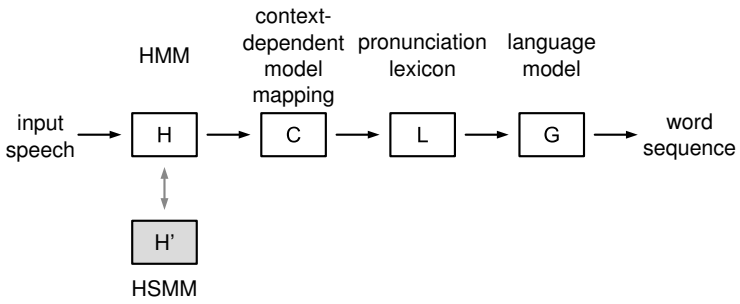


Figure 5.4: WFSTs for speech recognition.

for normalized and unnormalized state duration probabilities, respectively. It is noted that the normalization has similar effect to the duration weighting.

5.2 Experiments

To evaluate the performance of the proposed HSMM-based speech recognition system, speaker-dependent continuous phoneme recognition experiments were conducted on the ATR Japanese speech database B set (phonetically-balanced sentences). In all experiments, the speech data was down-sampled from 20 kHz to 16 kHz, windowed at a 5 ms frame rate using a 25 ms Blackman window, and parameterized into 25 mel-cepstral coefficients with a mel-cepstral analysis technique. Static coefficients including the zero-th coefficients and their first and second derivatives were used as feature parameters. 3-state left-to-right structures were used and 118 questions about left and right phonetic contexts were prepared for decision tree construction. Each state output distribution was modeled by a Gaussian distribution with a diagonal covariance matrix. A WFST for decoding was constructed from WFSTs representing chained triphone HMMs and a phoneme network (phoneme-pair grammar) based on the WFST composition and determinization. Maximum and minimum state duration of each HSMM state was limited to $\xi_i \pm \sqrt{\sigma_i^2} \times 2$.

5.2.1 Model size

Phonetic decision-tree-based context-clustering [39] was applied independently to state output and duration probability distributions. The MDL criterion was used to stop tree growth [61]. We changed the weight for the penalty term of c (Eq. (9) in [61]) to construct acoustic models with various numbers of parameters. The same weight c was used to cluster both state output and duration probability distributions. Thus, the number of

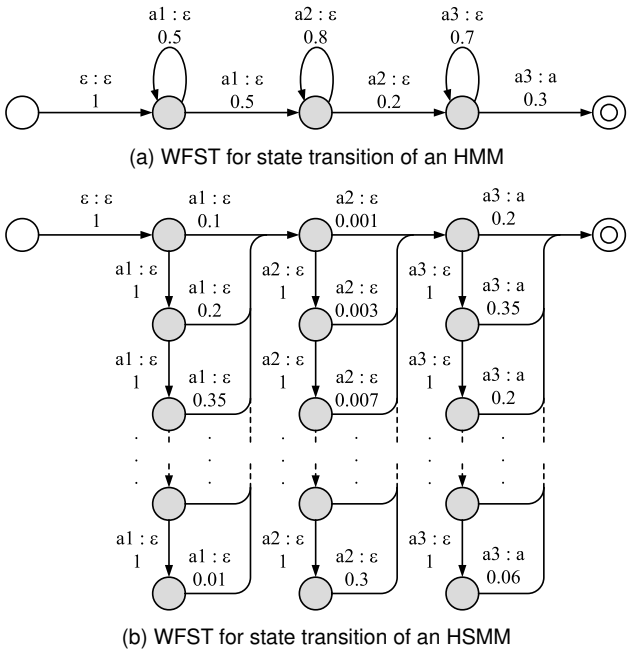


Figure 5.5: WFSTs for state transitions of an HMM and an HSMM.

state duration probability distributions changed according to the number of state output probability distributions.

Our WFST decoder for phoneme recognition can be represented as

$$\log\left(\boldsymbol{O} \mid \Lambda\right) \simeq \max_{\boldsymbol{Q}} \left\{\log P\left(\boldsymbol{O} \mid \boldsymbol{Q}, \Lambda\right) + w \log P\left(\boldsymbol{Q} \mid \Lambda\right)\right\} \tag{5.19}$$

where, $\boldsymbol{O}$, Λ, $\boldsymbol{Q}$, and w are an observation vector sequence, a sequence of phoneme HSMMs, an state sequence, and duration weight. In this experiment, we set $w = 1$ at all frames.

In the first experiment, phonetically balanced 450 sentences uttered by a speaker MHT were used for training HMMs and HSMMs. The remaining 53 sentences were used for evaluation. We fixed the beam width to 2000 and evaluated the effect of modeling state duration probability distributions.

Figure 5.6 shows the result. It can be seen from the figure that the proposed fully consistent HSMM-based system represents an improvement over the conventional HMM-based system in all settings.

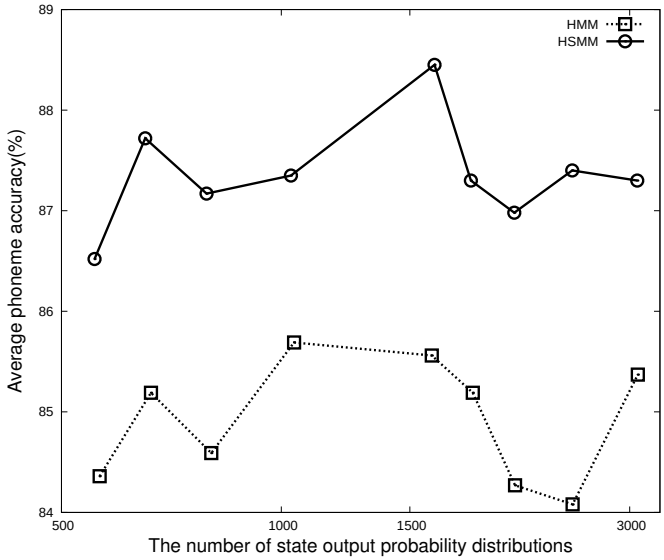


Figure 5.6: Average phoneme accuracy versus the number of state output probability distributions.

5.2.2 Search efficiency

The proposed HSMM-based system has a larger search space because the state transition WFST of HSMMs is more complex than that of HMMs. Therefore, we expected that its performance would depend strongly on beam width. To test this expectation, we fixed MDL weight c to 1 and examined the effective beam width of both the HMM- and HSMM-based systems. If the same beam width is used, the computational complexities of both systems are almost equal.

Figure 5.7 shows the results. It can be seen from the figure that if the beam width is lower than 200, the HSMM-based system does not perform as well as the HMM-based system. However, if the beam width is larger than 200, the HSMM-based system performs better.

5.2.3 Duration weight

In the third experiment, we fixed the beam width to 2000 and evaluated the effect of duration weight.

Figure 5.8 shows the results. As the duration weight increased, the performance improved, peaking when the weight reached 20. At this point, performance of the HSMM-based system achieved about 48% error reduction over the HMM-based system.

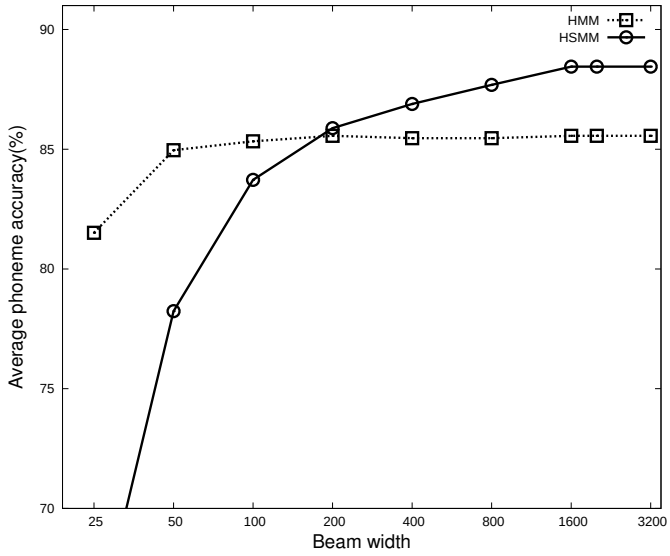


Figure 5.7: Average phoneme accuracy versus beam width.

5.2.4 Number of Mixture

To test the effect of the number of mixtures, we changed the number of mixtures of both the HMM- and HSMM-based systems[1]. The same MDL weight, beam width and duration weight were used.

Figure 5.9 shows the results. It can be seen from the figure that if the number of mixtures is higher than 4, the HSMM-based system does not perform as well as the HMM-based system. However, comparing the best performance for each of the 2-mix HSMM-based system and the 4-mix HMM-based system, we see that the 2-mix HSMM-based system performs better.

5.2.5 Comparative experiment

To investigate the effects of the three approximations mentioned in Section 5.1, we conducted a comparative experiment using 10 speakers (4 female speakers FKN, FKS, FTK, FYM, 6 male speakers MHO, MHT, MMY, MSH, MTK, MYI). In this experiment, we constructed the following 5 systems:

[1]In this experiment, we used fast forward-backward algorithm reported in [62] to reduce computational time of mixture.

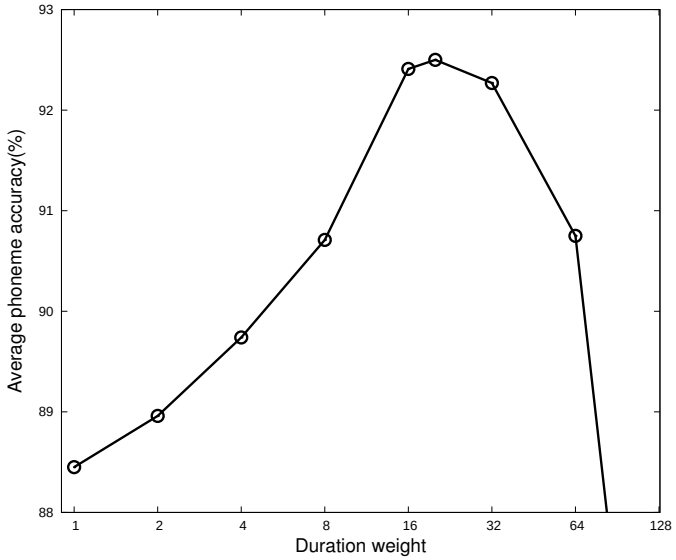


Figure 5.8: Average phoneme accuracy versus duration weight.

HMM An HMM-based system.

HSMM (train) An HSMM-based system with the approximation that state duration probability distributions were estimated based on statistics calculated by the forward-backward algorithm of the HMM [5].

HSMM (mono) An HSMM-based system with monophone state duration probability distributions.

HSMM (state) An HSMM-based system with a common state sharing structure for both state output and duration probability distributions[2]. In this experiment, context-clustering was applied using state output likelihood only.

HSMM (rescore) An HSMM-based system with the approximation that the 100 best hypotheses generated by the HMMs were rescored using the HSMM likelihood [12].

HSMM The proposed fully consistent HSMM-based system.

[2]It is effective in the sense of speech recognition performance that state output and duration probability distributions have independent state sharing structures. However, the system that state output and duration probability distributions have common state sharing structures performs a more effective search because their structures are combined by WFST optimization. Future work includes investigations of search efficiency with WFST optimization when state output and duration probability distributions have common state sharing structures.

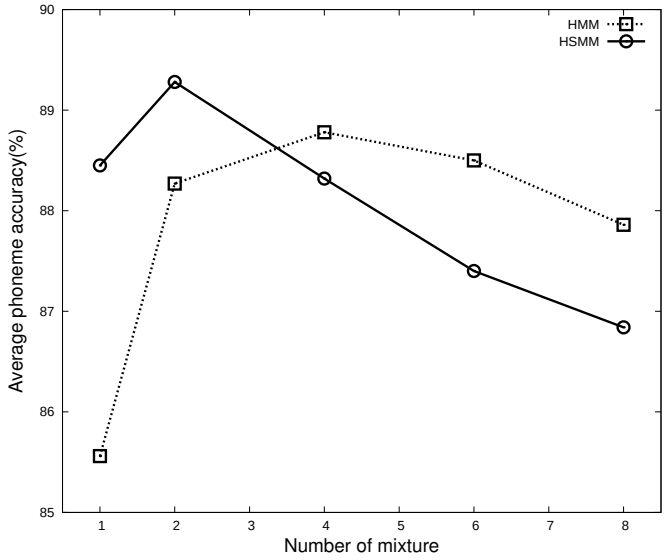


Figure 5.9: Average phoneme accuracy versus number of mixture.

The beam width were fixed to 2000. Phoneme insertion penalty and duration weight were optimized for each system.

Figure 5.10 shows the speech recognition performance of each system. Comparing "HSMM" with "HSMM (train)" and "HSMM (rescore)", we see that the approximation in training and decoding, respectively, degrade the speech recognition performance significantly. Although difference between "HSMM" and "HSMM (mono)" is small, "HSMM" performs better than "HSMM (mono)." It is seen from Figure 5.10 that "HSMM (state)" is worse than "HSMM (mono)". It seems that the lack of training data was caused by common state sharing structures of state output and duration probability distributions. By comparing "HMM" with "HSMM (mono)" and "HSMM", we found that the differences were statistically significant [63] at the 5% level. Finally, we can see that by avoiding all three approximations, the fully consistent HSMM-based system "HSMM" achieved about 9.1% error reduction over the standard HMM-based system "HMM".

5.2.6 Speaker-independent experiment

To test the effects of speaker-dependency, we conducted a speaker-independent continuous phoneme recognition experiment using 10 speakers. In this experiment, nine speakers data sets and one speaker data set are used as traing and testing, respectively. The beam width were fixed to 2000. Phoneme insertion penalty and duration weight were optimized

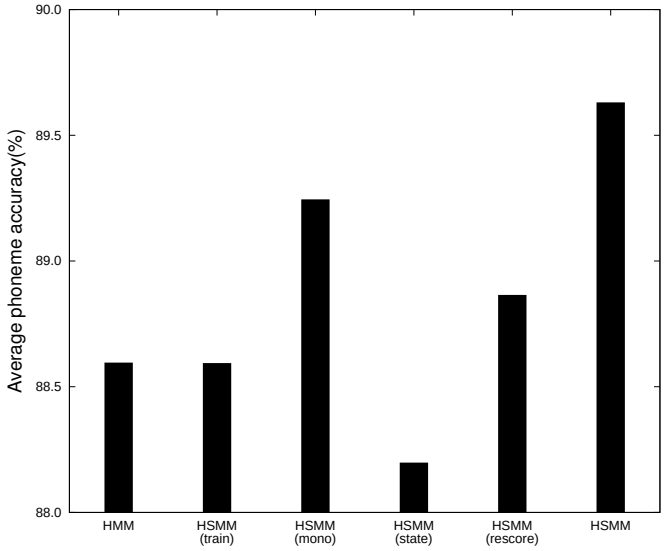


Figure 5.10: Comparative experiment: Average phoneme accuracy with insertion penalty and duration weight.

for each system.

Figure 5.11 shows the speech recognition performance of each system. It can be seen from the figure that the HMM-based system does not perform as well as the HSMM-based system for both speaker-dependent and speaker-independent tasks. This means that state duration modeling is effective not only for speaker-dependent tasks but speaker-independent tasks.

5.3 Summary

In this chapter, we constructed a fully consistent HSMM-based speech recognition system and evaluated its performance while avoiding approximations in training, context-clustering, and decoding. The result showed an obvious improvement in phoneme recognition accuracy. Future works include evaluation on speaker independent speech recognition tasks with multi-mixture state output probability distributions and investigation other kind of distributions such as gamma, Poisson and log Gaussian distributions for state duration modeling.

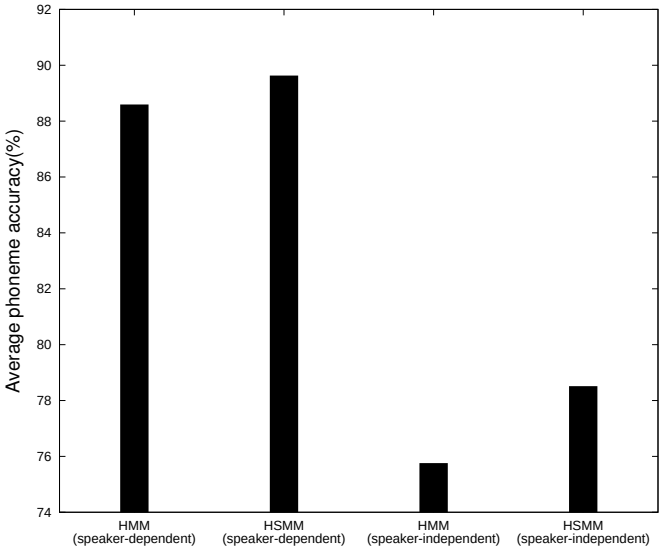


Figure 5.11: Speaker-independent experiment: Average phoneme accuracy with insertion penalty and duration weight.

Chapter 6

A covariance-tying technique for HMM-based speech synthesis

The most widely used speech synthesis technique is unit selection synthesis [3, 46, 47], in which appropriate sub-word units are selected from large speech databases. Although this technique can synthesize high-quality speech, it requires large databases of recorded speech. Furthermore, it usually requires excessively large footprints to put it on embedded devices such as mobile phones, PDAs, car navigation systems, and game machines.

Statistical parametric speech synthesis based on HMMs [5, 64] has grown in usage. Figure 4.1 gives an overview of a typical HMM-based speech synthesis system. In this system, the spectrum, excitation, and duration of speech are modeled simultaneously by context-dependent HMMs, and speech parameter trajectories are generated from the HMMs themselves under constraints between static and dynamic features [51]. One of the attractive points of HMM-based speech synthesis is its small footprint. HMM-based systems usually have smaller footprints than unit selection systems, because they store statistics rather than speech waveforms. However, further reduction is essential to put these systems on embedded devices that have little memory.

Speech parameters such as spectrum, excitation, and duration depend on a variety of contextual factors such as phoneme identities, accent types, and parts-of-speech. In the HMM-based speech synthesis system, context-dependent models are used to capture these contextual factors. If more combinations of these contextual factors are taken into account, we should be able to obtain more accurate models. However, as the number of contextual factors increases, the number of possible combinations also increases exponentially. As a result, it is difficult to robustly estimate model parameters due to the lack of training data. Furthermore, it is impossible for a finite set of training data to cover every possible combination of contextual factors. Various parameter-tying tech-

niques have been developed [58, 65–68] to avoid this problem. Among them, a decision tree-based context-clustering technique [69] has been widely used. In the HMM-based speech synthesis system, distributions of spectrum, excitation, and duration are clustered individually because they have their own contextual dependencies.

In this technique, top-down clustering is performed to maximize the likelihood of model parameters with respect to the training data by using questions about contexts. Then, HMM identifies which of those clustered into the same leaf node are tied. Unseen models can be generated by traversing the decision trees. Various criteria [69–73] have been proposed for selecting the questions to be used.

Conventionally, we construct an HMM stream-level tying structure in HMM-based speech synthesis, i.e., mean vectors and covariance matrices have exactly the same parameter-tying structure (Figure 6.1 (a)). However, we empirically know that covariance matrices have a smaller impact on the quality of synthesized speech than mean vectors. On the basis of this knowledge, a technique for context-clustering mean vectors while tying all covariance matrices (Figure 6.1 (b)) should be tested in HMM-based speech synthesis. If each parameter is stored in a single-precision floating-point number and the dimensionality of Gaussian distributions is 120, approximately 938 KBytes are required to store 1,000 Gaussian distributions with diagonal covariance matrices (statistics associated to the leaf nodes). However, tying all covariance matrices reduced it almost by half (469 KBytes).

Semi-tied covariance is one of the major covariance-tying techniques. This technique is a simple extension of the standard diagonal or full covariance matrices used with HMMs. Instead of having a distinct covariance matrix for every distribution, each covariance matrix consists of two elements, a component-specific diagonal covariance element and a tied transform. It is important to make the difference between the semi-tied covariance technique and the proposed technique clear.

The rest of this chapter is organized as follows. Section 6.1 describes the decision-tree-based context clustering technique. Context clustering for semi-tied covariance matrices are presented in Section 6.2. Section 6.3 describes the proposed decision tree-based context-clustering technique for mean vectors while tying all covariance matrices. Subjective listening test results are shown in Section 6.4. Finally, concluding remarks and future plans are presented in final section.

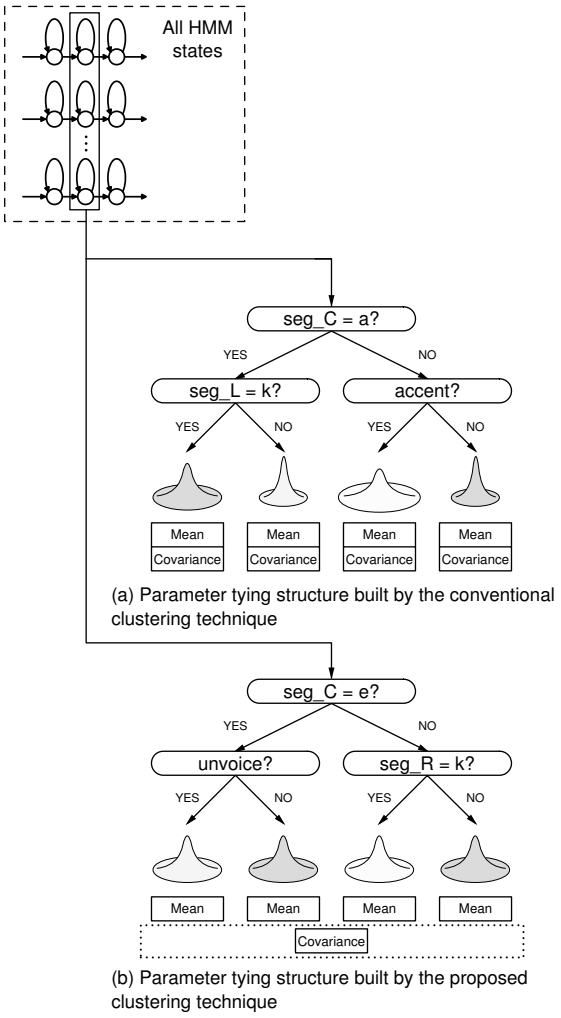


Figure 6.1: Context-dependent parameter-tying structure built by conventional and proposed clustering techniques.

6.1 Decision tree-based context clustering

In the decision-tree-based context-clustering technique, top-down clustering is performed to locally maximize the likelihood of model parameters with respect to the training data using pre-defined questions about contexts. Then, mean vectors and covariance matrices of HMM states clustered to the same leaf (terminal) node are tied. As a result, an HMM state-level tying structure can be constructed. The mean vector and the covariance matrix associated to the leaf node S, $\boldsymbol{\mu}_S$ and $\boldsymbol{\Sigma}_S$, can be estimated using the ML criterion as

$$\boldsymbol{\mu}_S = \frac{\sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t)\, \boldsymbol{o}_t}{\sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t)}, \tag{6.1}$$

$$\boldsymbol{\Sigma}_S = \frac{\sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t)\, (\boldsymbol{o}_t - \boldsymbol{\mu}_S)(\boldsymbol{o}_t - \boldsymbol{\mu}_S)^\top}{\sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t)}, \tag{6.2}$$

where T is the total number of frames in the training data, $\boldsymbol{M}_S$ is a set of HMM states clustered to the leaf node S, and $\gamma_m(t)$ is the posterior probability of an HMM state m for an observation vector at frame t, $\boldsymbol{o}_t$. The total log likelihood of the Gaussian distribution of node S to the associated training data is calculated as

$$\begin{aligned} \mathcal{L}(S) &= \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \log \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_S, \boldsymbol{\Sigma}_S) \\ &= -\frac{1}{2} \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \left\{ n + \log\left(2\pi \left|\boldsymbol{\Sigma}_S\right|\right) \right\}, \end{aligned} \tag{6.3}$$

where n is the dimensionality of $\boldsymbol{\mu}_S$.

The minimum description length (MDL) criterion [69] has been used in the HMM-based speech synthesis system to automatically control the size of decision trees. When cluster S is divided into S_{q+} and S_{q-} by a question q, the change of total description length by this split is calculated as follows:

$$\Delta_q = \mathcal{L}(S) - \left\{ \mathcal{L}(S_{q+}) + \mathcal{L}(S_{q-}) \right\} + \alpha \frac{N}{2} \log \Gamma(S_0), \tag{6.4}$$

where S_0 denotes a root node, α is a heuristic weight[1] for the penalty term of the MDL criterion, N is the number of parameters increased by this split, and

[1]The standard value of α is unity in the MDL criterion.

$$\Gamma(S) = \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t). \tag{6.5}$$

If all covariance matrices are diagonal covariance matrices, $N = n + n$. Note that the context-clustering based on the MDL criterion can be viewed as that based on the ML criterion with a threshold given by $\alpha \frac{N}{2} \log \Gamma(S_0)$.

6.2 Context clustering for semi-tied covariance matrices

In this section, we describe the semi-tied covariance technique since it will be evaluated as a conventional method in Section 6.4. In the HMM-based speech synthesis system, there is a choice of the form of the covariance matrices. When diagonal covariance matrices are used, elements of the feature vector are assumed to be independent. On the other hand, when full covariance matrices are used, all correlations are explicitly modeled. However, when full covariance matrices are used, the number of parameters per Gaussian component increases exponentially. Compared with a diagonal covariance matrix, the number of parameters per distribution increases to $n + \frac{n(n+1)}{2}$ from $n + n$. Furthermore, the number of training samples per distribution decreases. Due to this massive increase in the number of parameters, diagonal covariance matrices are generally used in HMM-based speech synthesis. Using the semi-tied covariance matrix is a good solution to this problem [74].

Semi-tied covariance matrices are a simple extension of the standard diagonal or full covariance matrices used with HMMs. Instead of having a distinct covariance matrix for every distribution, each covariance matrix consists of two elements, a component-specific diagonal covariance element, $\boldsymbol{\Sigma}^{(\mathrm{diag})}$, and a tied transform, $\mathbf{H}$. The form of the covariance matrix of state S is defined as

$$\boldsymbol{\Sigma}_S^{(\mathrm{stc})} = \mathbf{H} \boldsymbol{\Sigma}_S^{(\mathrm{diag})} \mathbf{H}^\top. \tag{6.6}$$

The number of parameters increased by a split, N, becomes $n + n$.

6.3 Context clustering while tying all covariance matrices

The decision-tree-based context-clustering techniques used in HMM-based speech synthesis systems construct an HMM state-level tying structure, i.e., the same tying structure is used for both mean vectors and covariance matrices. However, covariance matrices have less impact on the quality of synthesized speech than mean vectors. For example, even if we manually modify values of covariance matrices, the speech parameter trajectories generated from the original and modified models are often close to each other. In this chapter, we construct the tying structure of mean vectors using decision trees while tying all covariance matrices.

If all covariance matrices are tied, the total log likelihood of the leaf node S to the associated training data is calculated as follows:

$$
\begin{aligned}
\mathcal{L}'(S) &= \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \log \mathcal{N}(\boldsymbol{o}_t; \boldsymbol{\mu}_S, \boldsymbol{\Sigma}_g) \\
&= -\frac{1}{2} \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) (\boldsymbol{o}_t - \boldsymbol{\mu}_S)^\top \Sigma_g^{-1} (\boldsymbol{o}_t - \boldsymbol{\mu}_S) \\
&= -\frac{1}{2} \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \log\left(2\pi \left|\boldsymbol{\Sigma}_g\right|\right) \\
&= -\frac{1}{2} \mathrm{Tr}\left\{ \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) (\boldsymbol{o}_t - \boldsymbol{\mu}_S) (\boldsymbol{o}_t - \boldsymbol{\mu}_S)^\top \Sigma_g^{-1} \right\} \\
&\quad -\frac{1}{2} \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \log\left(2\pi \left|\boldsymbol{\Sigma}_g\right|\right) \\
&= -\frac{1}{2} \sum_{t=1}^{T} \sum_{m \in \boldsymbol{M}_S} \gamma_m(t) \left\{ \mathrm{Tr}\left(\boldsymbol{\Sigma}_S \boldsymbol{\Sigma}_g^{-1}\right) + \log\left(2\pi \left|\boldsymbol{\Sigma}_g\right|\right) \right\}, \qquad (6.7)
\end{aligned}
$$

where $\boldsymbol{\Sigma}_g$ is a globally tied covariance matrix, and $\boldsymbol{\Sigma}_S$ is defined in Eq. (6.2). Note that $\boldsymbol{\Sigma}_g$ is fixed in the context-clustering process because the computational cost is large.

When cluster S is divided into S_{q+} and S_{q-} by a question q, the change of total description length by this split is calculated as follows:

$$\Delta'_q = \mathcal{L}'(S) - \left\{ \mathcal{L}'(S_{q+}) + \mathcal{L}'(S_{q-}) \right\} + \alpha \frac{N}{2} \log \Gamma(S_0) . \tag{6.8}$$

Unlike Eq. (6.4), the number of parameters N increased by the split becomes n in this case because only mean vectors are split. We can expect that the proposed technique can efficiently reduce the footprints of HMM-based speech synthesis systems while retaining the quality of the synthesized speech.

6.4 Experiments

6.4.1 Experimental condition

To evaluate the effectiveness of the proposed technique, subjective listening tests were conducted. The first 450 sentences of the phonetically balanced 503 sentences from the ATR Japanese speech database B-set [75], uttered by male speaker MHT, were used for training. The remaining 53 sentences were used for evaluation. Speech signals were sampled at 16kHz and windowed with a 5-ms shift, and mel-cepstral coefficients [76] were obtained from STRAIGHT spectra [77]. Feature vectors consisted of spectrum and excitation parameters. The spectrum parameter vectors consisted of 39 STRAIGHT mel-cepstral coefficients including the zero coefficient and their delta and delta-delta coefficients. The excitation parameter vectors consisted of log F_0 and its delta and delta-delta. A seven-state (including the beginning and ending null states), left-to-right, no-skip structure was used for the hidden semi-Markov model [7]. The spectrum stream was modeled by single multi-variate Gaussian distributions. The excitation stream was modeled by multi-space probability distributions [78], each of which consists of a Gaussian distribution for voiced frames and a discrete distribution for unvoiced frames. State durations of each model were modeled by a five-dimensional (equal to the number of emitting states in each model) multi-variate Gaussian distribution. The decision tree-based context-clustering technique was separately applied to distributions for spectrum, excitation, and state duration. A speech parameter generation algorithm considering global variance (GV) [79] was used for parameter generation.

The MDL criterion [69] was used to control the size of decision trees. We changed the heuristic weight for the penalty term of α in Eq. (6.4) and Eq. (6.8) to construct acoustic models with various numbers of parameters. The weights used here were 8.0, 4.0, 2.0, 1.0, 0.5, and 0.25. Although the decision tree-based context-clustering technique was separately applied to distributions for spectrum and excitation, the same α was used.

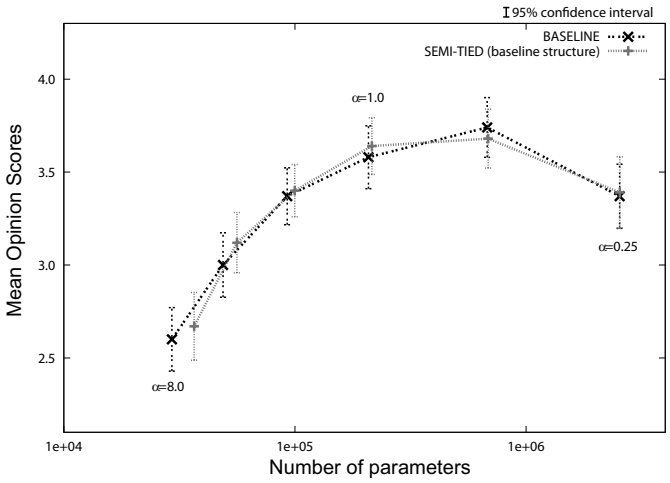


Figure 6.2: Subjective experimental results: Conventional method versus semi-tied covariance method. The same mean tying structures are constructed.

Ten subjects participated in these listening tests. Ten sentences were randomly selected from 53 sentences for each subject. The subjects were asked to rate the naturalness of the synthesized speech on a scale from 1 (completely unnatural) to 5 (natural). All experiments were carried out using headphones in a soundproof room.

6.4.2 Semi-tied covariance technique

To reduce the burden on listeners, the listening tests were split into five parts.

To confirm the effect of the covariance matrix type for naturalness and footprint, we evaluated the semi-tied covariance technique in the first experiment. The following two methods were evaluated.

BASELINE : The same structure tied by conventional context-clustering was used for mean vectors and diagonal covariance matrices.

SEMI-TIED (baseline structure) : Although the tying structure of mean vectors was exactly the same as the **BASELINE** system, all covariance matrices were semi-tied.

Figure 6.2 shows the subjective listening test results. The **SEMI-TIED (baseline structure)** system did not improve compared with **BASELINE** system. Although elements of the feature vector are assumed to be dependent in the semi-tied covariance technique, the assumption seems to have little impact on the quality of the synthesized speech.

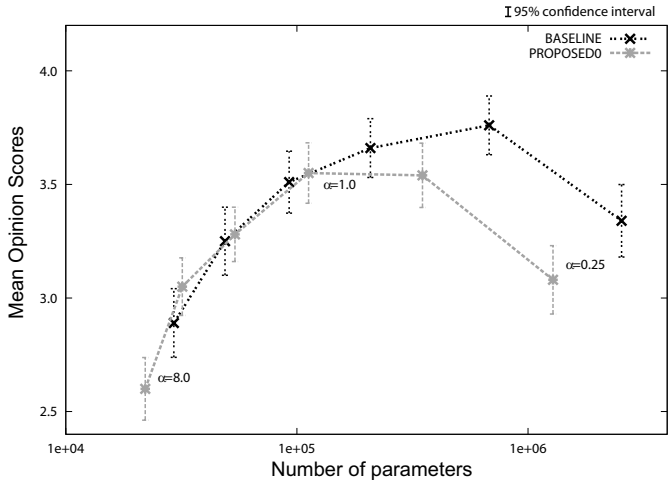


Figure 6.3: Subjective experimental results: Conventional method versus tied full covariance method. The same mean tying structures are constructed.

6.4.3 Tied covariance technique

Next, a listening test was designed to confirm the empirical knowledge that covariance matrices have little impact on the quality of synthesized speech. The following two methods were evaluated.

BASELINE : The same structure tied by conventional context-clustering was used for mean vectors and diagonal covariance matrices.

PROPOSED0 : Although the tying structure of mean vectors was exactly the same as the **BASELINE** system, all full covariance matrices were tied.

Figure 6.3 shows the subjective listening test results. The **PROPOSED0** system reduced scores slightly compared with the **BASELINE** system. A large amount of memory is required for full covariance matrices, even using the tying technique. Furthermore, covariance matrices without diagonal elements have little impact on the quality of synthesized speech. Therefore, it seems that the use of full covariance matrices is not appropriate for the embedded devices.

Next, we replaced full covariance matrices with diagonal covariance matrices. The following two methods were evaluated.

BASELINE : The same structure tied by conventional context-clustering was used for mean vectors and diagonal covariance matrices.

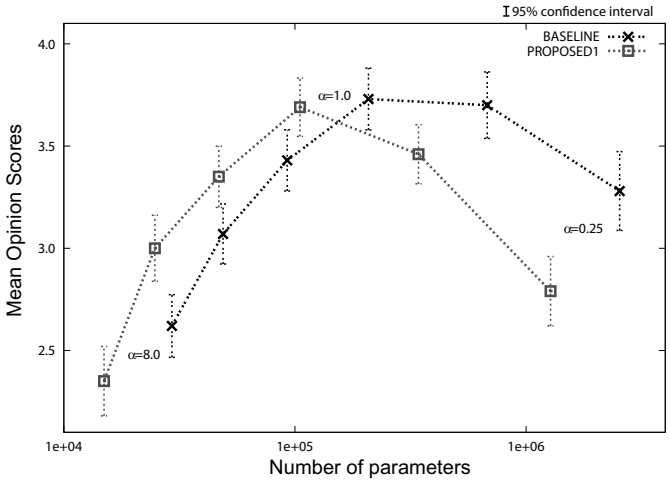


Figure 6.4: Subjective experimental results: Conventional method versus tied diagonal covariance method. The same mean tying structures are constructed.

PROPOSED1 : Although the tying structure of mean vectors was exactly the same as the **BASELINE** system, all diagonal covariance matrices were tied.

Figure 6.4 shows the subjective listening test results. The **PROPOSED1** system achieved almost the same subjective scores with almost half the number of parameters (footprints) when $\alpha = 1.0$. Tying diagonal covariance matrices seems to be more efficient than reducing the size of decision trees to achieve the same footprints.

The fourth listening test evaluated the performance of the proposed clustering technique while tying all diagonal covariance matrices. Note that the proposed clustering technique was not applied to full covariance matrices because of its large computational cost. The following two methods were compared.

BASELINE : The same structure tied by the conventional context-clustering was used for mean vectors and diagonal covariance matrices.

PROPOSED2 : Mean vectors were clustered by decision trees while tying all diagonal covariance matrices using the technique described in Section 6.3.

Figure 6.5 shows the experimental results. The **PROPOSED2** system significantly reduced the number of parameters. Furthermore, it achieved slightly better subjective scores than **BASELINE**. When each parameter was stored in a single-precision floating-point number, the footprint of the **BASELINE** system with $\alpha = 1.0$ was about 813 KBytes. On

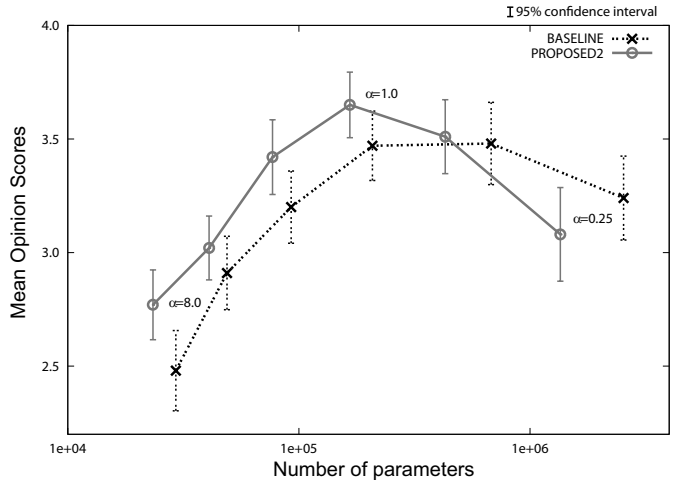


Figure 6.5: Subjective experimental results: Conventional method versus tied diagonal covariance method. Different mean tying structures are constructed.

the other hand, that of the **PROPOSED2** system with $\alpha = 1.0$ was 649KBytes. Furthermore, the **PROPOSED2** system with $\alpha = 2.0$ consumed only 300 KBytes while retaining the quality of synthesized speech close to the **BASELINE** system with $\alpha = 1.0$. Figure 6.6 shows the average log probabilities per frame of the **BASELINE**, **PROPOSED1**, and **PROPOSED2** systems. In terms of ML estimation of HMM parameters, tying all the covariance matrices decreased the likelihood function. The **PROPOSED2** system had a slightly higher probability than the **PROPOSED1** system because of the proposed context-clustering technique, which constructs appropriate mean vector structures while tying all covariance matrices.

The final listening test evaluated the performance of the two proposed systems compared with the baseline system. To guarantee the generalizability of the proposed method, the first 450 sentences of the phonetically balanced 503 sentences from the ATR Japanese speech database B-set [75], uttered by male speaker MHT and female speaker FKN, were used for training speaker-dependent models. The remaining 2 sets of 53 sentences were used for evaluation. Ten subjects participated in this listening test. Twenty sentences were randomly selected from 106 sentences for each subject. The **BASELINE** system with $\alpha = 1.0$, the **PROPOSED1** system with $\alpha = 1.0$, and the **PROPOSED2** systems with $\alpha = 1.0$ and 2.0 were compared. Figure 6.7 shows the subjective results. All proposed methods had a smaller footprint than the **BASELINE** system while maintaining the quality of the synthesized speech. The **PROPOSED2** system with $\alpha = 1.0$ had better subjective scores than the **PROPOSED1** system with $\alpha = 1.0$. Table 6.1 shows the number of leaf nodes of each system with $\alpha = 1.0$. In the **PROPOSED2** system, the number

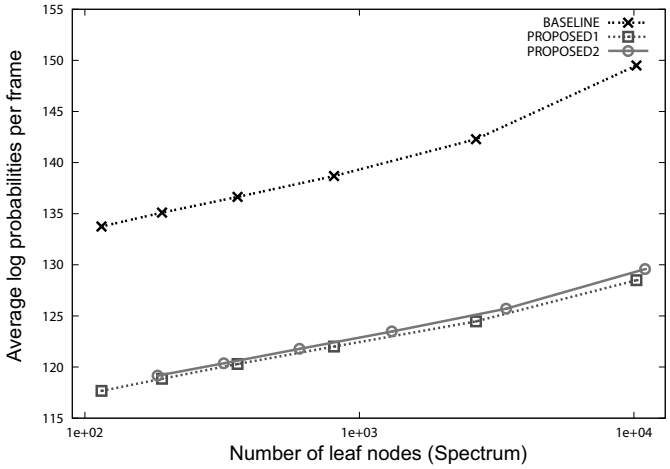


Figure 6.6: Objective experimental results: Conventional method versus two proposed methods.

of mean parameters can be increased even when the total number of parameters is decreased. It is supposed that the balance between model complexities of mean parameters and covariance parameters can be adjusted by using the proposed context-clustering technique, which constructs the appropriate mean vector structure while tying all covariance matrices.

6.5 Summary

A technique for reducing the footprints of HMM-based speech synthesis systems by tying all covariance matrices is described. Experimental results showed that the proposed technique efficiently reduced the footprints of an HMM-based speech synthesis system to less than half of its original size while retaining the quality of the synthesized speech. Future

Table 6.1: Comparison of the number of leaf nodes.

	Number of leaf nodes			
	Spectrum		F0	
	Mean	Covariance	Mean	Covariance
BASELINE	808	808	2015	2015
PROPOSED1	808	1	2015	1
PROPOSED2	1311	1	2210	1

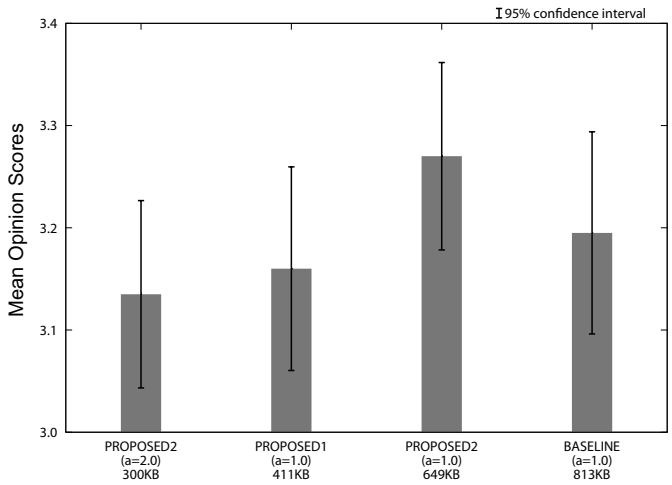


Figure 6.7: Subjective experimental results: Conventional method versus two proposed methods. Their footprints were calculated on the assumption that each parameter was stored in a single-precision floating-point number.

work includes using a separated clustering technique for mean vectors and covariance matrices.

Chapter 7

Simultaneous linguistic and acoustic model training for TTS conversion system

Standard text-to-speech (TTS) systems consist of two major modules: text analysis and speech synthesis modules. Conventionally, these two modules are constructed independently. The text analysis module is trained using text corpora. The module includes phrasing and prosodic models. On the other hand, the speech synthesis module is trained using a labeled speech database. The module includes acoustic models used for speech synthesis, which are based on the hidden Markov model (HMM). Therefore, if these two modules were combined and trained simultaneously as a unified model, we would expect improved overall performance of a TTS system.

In this chapter, we define a new integrated model for simultaneous linguistic and acoustic modeling. Two model parameter sets were simultaneously optimized by the proposed training algorithm. In this manner, we directly-formulate the TTS problem of synthesizing a speech waveform from a word sequence. Another advantage of the proposed approach is that hand-labeling of phrasing and prosodic events not required for neither linguistic nor acoustic model training because these labels are regarded as latent variables in the model.

7.1 Linguistic model

Text analysis modules in TTS systems consist of several parts (e.g., pronunciation, part-of-speech (POS) tagging, phrasing, and prosodic models), and we call the set of those

parts a "linguistic model." In this study, phrasing and prosodic models in particular are used as the "linguistic model" in accordance with tones and break indices "ToBI" [80].

We used two phrasing models. The first model is based on a 7-gram model, and the second model is based on a 4-gram POS model. Two types of pitch events are marked by the prosodic model: pitch events associated with accented syllables (pitch accents) and pitch events associated with intonational boundaries (phrasal tones). Therefore, we used two decision trees for the prosodic model in this chapter.

7.2 Integration of linguistic and acoustic models

In this section, we define a new integrated model to optimize linguistic and acoustic models simultaneously. First, a linguistic and an acoustic model are defined. The likelihood of the linguistic model λ_W, e.g., N-gram, decision tree model, is written as $P(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W)$, where $\boldsymbol{L}$ and $\boldsymbol{W}$ are label sequence and word sequence, respectively. On the other hand, the likelihood of the acoustic model λ_H is given by

$$P(\boldsymbol{O} \mid \boldsymbol{L}, \lambda_H) = \sum_{\boldsymbol{q}} P(\boldsymbol{O} \mid \boldsymbol{q}, \lambda_H) P(\boldsymbol{q} \mid \boldsymbol{L}, \lambda_H), \tag{7.1}$$

where $\boldsymbol{O} = (\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T)$ and $\boldsymbol{q} = (q_1, q_2, \ldots, q_T)$ are observation vector sequence and state sequence, respectively.

An integrated model λ that directly models the observation vector sequence $\boldsymbol{O}$ for the word sequence $\boldsymbol{W}$ is derived by combining the linguistic model λ_W and acoustic model λ_H, as follows:

$$P(\boldsymbol{O} \mid \boldsymbol{W}, \lambda) = \sum_{\boldsymbol{L}} \sum_{\boldsymbol{q}} P(\boldsymbol{O} \mid \boldsymbol{q}, \lambda_H) P(\boldsymbol{q} \mid \boldsymbol{L}, \lambda_H) P(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W), \tag{7.2}$$

where

$$\lambda = \{\lambda_H, \lambda_W\}. \tag{7.3}$$

We performed the linguistic model λ_W and acoustic model λ_H training simultaneously to optimize all parameters of the integrated model λ. The derivation of the algorithm is

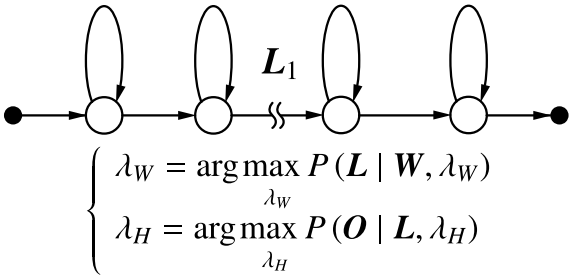


Figure 7.1: Conventional model optimization

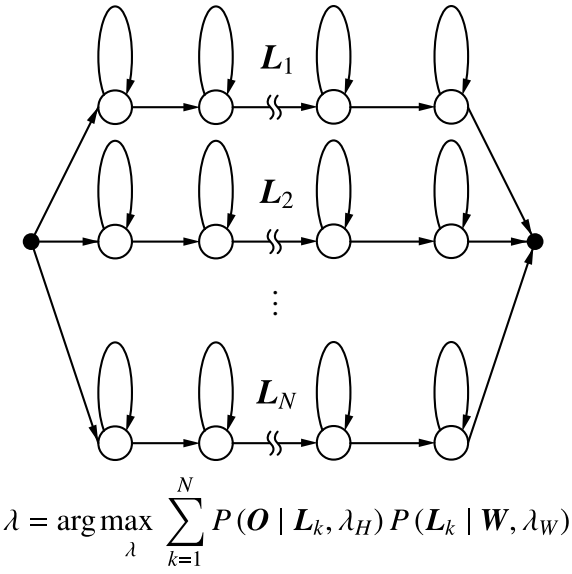


Figure 7.2: Proposed model optimization

shown in the next Section. Differences between the conventional and proposed training criteria are shown in Figure 7.1 and Figure 7.2, respectively. In the conventional model, training data has to be labeled by hand or an automatic labeling tool, which is time consuming or causes labeling errors, respectively. On the other hand, in the proposed model, the label sequence is regarded as a latent variable and marginalized like a state sequence in HMM. Labeling the training data accordingly in not necessary.

7.3 Parameter estimation formulas

7.3.1 EM algorithm

The expectation maximization (EM) algorithm [20] was used for training the proposed model. In the EM algorithm, the likelihood is maximized at each iteration using an auxiliary function called the Q-function:

$$\begin{aligned} Q\left(\lambda, \lambda^{'}\right) &= \sum_{\boldsymbol{L}} \sum_{\boldsymbol{q}} P\left(\boldsymbol{q}, \boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda_H, \lambda_W\right) \\ &\quad \log\left[P\left(\boldsymbol{O} \mid \boldsymbol{q}, \lambda_H^{'}\right) P\left(\boldsymbol{q} \mid \boldsymbol{L}, \lambda_H^{'}\right) P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W^{'}\right)\right], \end{aligned} \tag{7.4}$$

where λ, $\lambda^{'}$, and $P\left(\boldsymbol{q}, \boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda_H, \lambda_W\right)$ are the integrated model before updating, that after updating, and posterior probabilities of state sequence $\boldsymbol{q}$ and label $\boldsymbol{L}$, respectively. Posterior probabilities are calculated by Bayes' rule:

$$\begin{aligned} P\left(\boldsymbol{q}, \boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right) &= \frac{P\left(\boldsymbol{O}, \boldsymbol{q}, \boldsymbol{L} \mid \boldsymbol{W}, \lambda\right)}{\sum_{\boldsymbol{L}} \sum_{\boldsymbol{q}} P\left(\boldsymbol{O}, \boldsymbol{q}, \boldsymbol{L} \mid \boldsymbol{W}, \lambda\right)} \\ &= \frac{P\left(\boldsymbol{O} \mid \boldsymbol{q}, \lambda_H\right) P\left(\boldsymbol{q} \mid \boldsymbol{L}, \lambda_H\right) P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W\right)}{\sum_{\boldsymbol{L}} \sum_{\boldsymbol{q}} P\left(\boldsymbol{O} \mid \boldsymbol{q}, \lambda_H\right) P\left(\boldsymbol{q} \mid \boldsymbol{L}, \lambda_H\right) P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W\right)}. \end{aligned} \tag{7.5}$$

Increasing the value of the Q-function causes an increase in the likelihood of the training data:

$$Q\left(\lambda, \lambda^{'}\right) \geq Q\left(\lambda, \lambda\right) \Rightarrow P\left(\boldsymbol{O} \mid \lambda^{'}\right) \geq P\left(\boldsymbol{O} \mid \lambda\right). \tag{7.6}$$

Hence, maximization of the Q-function value at each iteration maximizes the likelihood of the training data. The EM algorithm starts with an initial model parameter λ^0, and iterates between the following two steps:

E-step : compute $Q\left(\lambda, \lambda^{(t)}\right)$
M-step : $\lambda^{(t+1)} = \arg\max_{\lambda} Q\left(\lambda, \lambda^{(t)}\right)$

where t denotes the number of the iteration. In this procedure, each step increases the value of the Q-function. Therefore, the likelihood of the training data either increases or remains unchanged at each iteration.

In the M-step of the integrated model λ, the linguistic model λ_W and the acoustic model λ_H were updated individually. The Q-function of the linguistic model λ_W is defined as

$$Q_W\left(\lambda, \lambda^{'}\right) = \sum_{\boldsymbol{L}} P\left(\boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right) \log P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda_W^{'}\right). \tag{7.7}$$

The acoustic model λ_H was optimized by setting derivatives of the Q-function to zero. As a result, the mean $\boldsymbol{\mu}_i$ and variance $\boldsymbol{\Sigma}_i$ of the i-th state output probability distribution (Gaussian distribution) were estimated as:

$$\boldsymbol{\mu}_i = \frac{\sum_t \sum_{\boldsymbol{L}} \gamma_i\left(t, \boldsymbol{L}\right) \boldsymbol{o}_t}{\sum_t \sum_{\boldsymbol{L}} \gamma_i\left(t, \boldsymbol{L}\right)} \tag{7.8}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_t \sum_{\boldsymbol{L}} \gamma_i\left(t, \boldsymbol{L}\right) \left(\boldsymbol{o}_t - \boldsymbol{\mu}_i\right) \left(\boldsymbol{o}_t - \boldsymbol{\mu}_i\right)^{\top}}{\sum_t \sum_{\boldsymbol{L}} \gamma_i\left(t, \boldsymbol{L}\right)}, \tag{7.9}$$

respectively, where

$$\begin{aligned} \gamma_i\left(t, \boldsymbol{L}\right) &= P\left(q_t = i, \boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right) \\ &= P\left(q_t = i \mid \boldsymbol{L}, \boldsymbol{O}, \lambda\right) P\left(\boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right). \end{aligned} \tag{7.10}$$

Posterior probabilities $P\left(q_t = i \mid \boldsymbol{L}, \boldsymbol{O}, \boldsymbol{W}, \lambda\right)$ of state q_t were computed by the forward-backward algorithm [1] using label sequence $\boldsymbol{L}$. On the other hand, posterior probabilities $P\left(\boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right)$ of label sequence $\boldsymbol{L}$ were written as follows:

$$P\left(\boldsymbol{L} \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right) = \frac{P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda\right) P\left(\boldsymbol{O} \mid \boldsymbol{L}, \lambda\right)}{\sum_{\boldsymbol{L}} P\left(\boldsymbol{L} \mid \boldsymbol{W}, \lambda\right) P\left(\boldsymbol{O} \mid \boldsymbol{L}, \lambda\right)}. \tag{7.11}$$

7.3.2 N-best approximation

Direct implementation of the EM algorithm is not feasible because the total number of possible combinations of label sequence $\boldsymbol{L}$ is too large. Thus, the N-best hypotheses generated by the text analysis module were used in this study[1]. The E-step was implemented accordingly as follows:

1 : generate N-best label sequences $\boldsymbol{L}_i, i = 1, \ldots, N$
2 : compute $P\left(\boldsymbol{O} \mid \boldsymbol{L}_i, \boldsymbol{W}, \lambda_H\right)$ for each label sequences $\boldsymbol{L}_i$
3 : compute $P\left(\boldsymbol{L}_i \mid \boldsymbol{O}, \boldsymbol{W}, \lambda\right)$ for each label sequences $\boldsymbol{L}_i$
4 : compute $Q\left(\lambda, \lambda'\right)$

In the M-step, model parameters were updated using the N-best label sequences. The above procedure optimizes the linguistic and acoustic models simultaneously. Furthermore, a state-sharing structure of HMM that matches the linguistic model was constructed by a context-clustering technique [69].

7.4 Experiment

7.4.1 Experimental conditions

Objective evaluations were conducted on the CMU-ARCTIC speech database to evaluate the performance of the proposed system. Training data, testing data, and speech analysis conditions are shown in Table 7.1. Each feature vector consisted of spectrum and F_0 parameter vectors. Each spectrum parameter vector consisted of the 0th - 39th STRAIGHT [77] mel-cepstral coefficients, their delta coefficients, and delta-delta coefficients. The F_0 parameter vector consisted of log F_0, its delta coefficient, and delta-delta coefficient. We used a 5-state left-to-right HMM structure with no-skip. Forty-one phonemes including the pause were used as speech units. Context-clustering based on a decision tree was applied to spectrum, F_0, and state duration models, individually. The minimum description length (MDL) criterion [69] was used to stop tree growth.

We trained linguistic models using the Boston University Radio Speech Corpus for the conventional automatic labeling technique. In the proposed system, these models were used as initial linguistic models.

[1] Although variational approximation is one of the methods for solving this problem, we chose the N-best approximation because label sequence $\boldsymbol{L}$ is strongly correlated with state sequence $\boldsymbol{q}$.

7.4.2 Evaluation

We calculated the root mean square error (RMSE) and correlation coefficient (Corr) of F_0 contour generation with respect to original speech in the voiced portions of the data. The RMSE and Corr are widely used to measure the accuracy of F_0 contour generation [81–84][2]. In this experiment, 4 systems were constructed as follows:

BASELINE: Only acoustic models were trained. The one-best label sequence was used.

PROSODIC: Both acoustic and prosodic models were trained simultaneously.

PHRASING: Both acoustic and phrasing models were trained simultaneously.

PROSODIC + PHRASING: Acoustic, prosodic, and phrasing models were trained simultaneously.

In the N-best approximation of simultaneous linguistic and acoustic model training, 100-best label hypotheses were used. About 20 days were taken to train the integrated models of the proposed system.

Calculations of RMSE and Corr are shown in Figure 7.3 and Figure 7.4, respectively. Three systems, PROSODIC, PHRASING, and PROSODIC + PHRASING, using simultaneous training of linguistic and acoustic models achieved a smaller RMSE and larger Corr, than those of "BASELINE." A graph of F_0 contours is shown in Figure 7.5. The F_0 contour generated by "PROSODIC" seems to exhibit a better goodness of fit with respect to original speech than that of "BASELINE."

Table 7.1: Experimental conditions

Database	CMU-ARCTIC speech database a female speaker SLT 1132 sentences train : 1000 sentences test : 132 sentences
Sampling rate	16kHz
Frame shift	5ms
Window length	25ms
Window function	Blackman window

[2]Although subjective evaluation experiments are also required, they have to be postponed because finding a sufficient number of native English speakers was not easy.

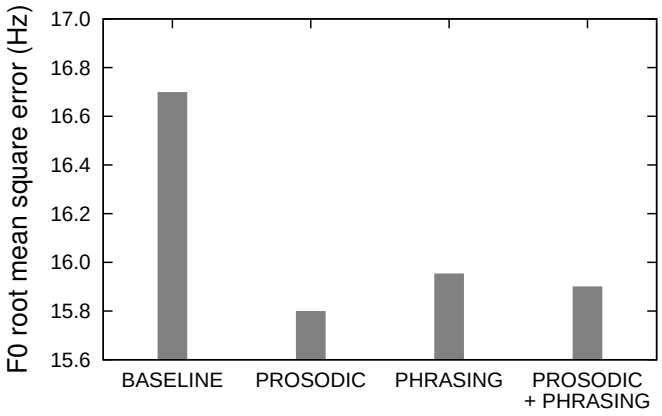


Figure 7.3: F_0 RMSE results

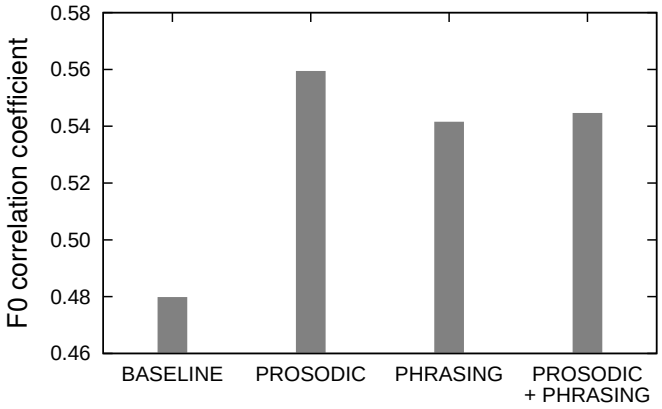


Figure 7.4: F_0 Corr results

7.5 Summary

In this chapter, we defined a new integrated model in which linguistic and acoustic models were combined into one model, and all model parameters were estimated simultaneously by the proposed training algorithm. We conducted objective evaluation experiments using phrasing and prosodic models as linguistic models to evaluate the effectiveness of the proposed system. The results demonstrate that the proposed system achieves better F_0 modeling accuracy than that of the conventional system. Future work will include simultaneous training of POS tagging modules and acoustic models. Subjective listening tests performed by native English speakers on a large database are also planned.

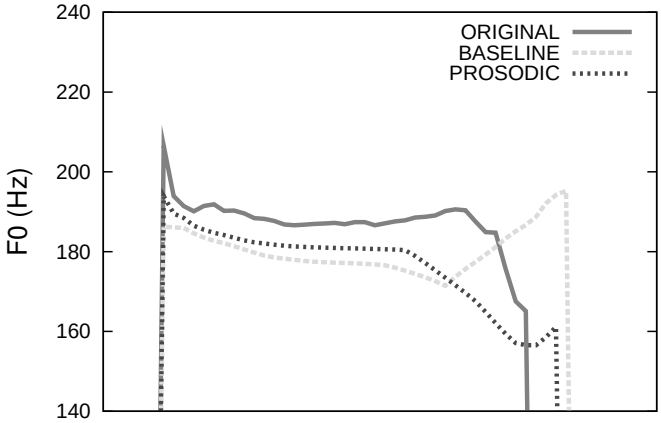


Figure 7.5: F_0 contours

Chapter 8

Unsupervised cross-lingual speaker adaptation

The goal of Speech-to-Speech Translation (S2ST) research is to "enable real-time, interpersonal communication via natural spoken language for people who do not share a common language" [85] and many large-scale projects (Verbmobil, Babylon, TC/LC-STAR, EU-Trans, ATR, etc.) have focused on this topic. In our EU FP7 project EMIME [86], we are developing a mobile device that performs personalized S2ST, such that a user's spoken input in one language is used to produce spoken output in another language, while continuing to sound like the user's voice.

Contrary to previous 'pipeline' S2ST systems that combined isolated automatic speech recognition (ASR), machine translation (MT), and text-to-speech (TTS) systems, or systems that coupled ASR with MT [87, 88], EMIME places the main emphasis on coupling ASR with TTS, specifically to enable cross-lingual speaker adaptation for HMM-based ASR and TTS [5, 89]. The principal modeling framework of speaker-adaptive HMM-based speech synthesis [89] is conceptually similar to conventional ASR systems (although without discriminative training) and it is therefore possible to share Gaussians, decision trees or linear transforms between the two [90].

In the EMIME project, we have conducted extensive experiments exploring the possibilities for combining ASR and TTS models. We have also developed unsupervised adaptation techniques for HMM-based TTS using either a phoneme recognizer [91] or a word-based large-vocabulary continuous speech recognizer (LVCSR) [13], and cross-lingual adaptation techniques for HMM-based TTS [14].

In this chapter, we integrate these developments into a single architecture which achieves unsupervised cross-lingual speaker adaptation for HMM-based speech synthesis. We

demonstrate an initial S2ST system built for four languages – American English, Mandarin, Japanese, and Finnish. Although all language pairs and directions are possible in our framework, only the English-to-Japanese adaptation was evaluated in the perceptual experiments presented here; these experiments focus on measuring the similarity between the output Japanese synthetic speech to the speech of the original English speaker. The following sections give an overview of the system built, the unsupervised cross-lingual speaker adaptation method and the TTS evaluation results.

8.1 Overview of the S2ST system using HMM-based ASR and TTS

All acoustic models, for both ASR and TTS, are trained on large conventional speech databases, comprising speech from hundreds of speakers, which were originally intended for ASR: WSJ0/1 (for English), Speecon Mandarin, JNAS (Japanese), and Speecon Finnish databases. Details of the front-end text processing used to derive phonetic-prosodic labels from the word transcriptions can be found in [92].

For each language, state-tied context-dependent speaker-independent HMMs (or multi-space distribution hidden semi-Markov models – MSD-HSMMs) are trained using speaker-adaptive training (SAT) [93]. For the state tying, minimum description length (MDL) automatic decision tree clustering is used [5]. The acoustic features for ASR are either the same as those for TTS or more typical ASR features such as MFCCs or PLPs. TTS acoustic features comprise the spectral and excitation features required for the STRAIGHT mel-cepstral vocoder with mixed excitation [89]. For unsupervised cross-lingual speaker adaptation and decoding, a multi-pass framework is used: in the first pass, initial transcriptions are obtained from speaker independent (SI) HMMs, and then CSMAPLR adaptation [94] is applied to SAT-HMMs (ASR) using these obtained transcriptions. In the second pass, using these adapted models, the transcriptions are refined. In the final pass, CSMAPLR transforms are estimated for SAT-HSMMs (TTS) with the refined transcriptions. These transforms can then be applied to the SAT-HSMMs for the output language, by employing a state-level mapping that has been constructed based on the Kullback-Leibler divergence (KLD) between pairs of states from the input and output TTS HMMs [14]. The ASR language models used for English, Mandarin and Japanese each contain about 20k bi-grams; the language model for Finnish is a word 10-gram plus a morph bi-gram [95]. For MT we simply used Google's AJAX language API[1]. In future work, this will be replaced by our own MT system based on one being developed for the AGILE project[2]. In the TTS mod-

[1] http://code.google.com/intl/ja/apis/ajaxlanguage/

[2] http://svr-www.eng.cam.ac.uk/research/projects/AGILE/

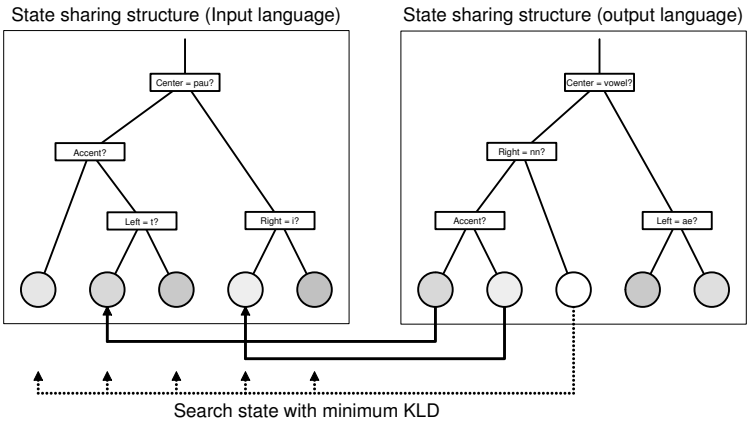


Figure 8.1: The state-mapping is learned by searching for pairs of states that have minimum KLD between input and output language HMMs. Linear transforms estimated with respect to the input language HMMs are applied to the output language HMMs, using the mapping to determine which transform to apply to which state in the output language HMMs.

ule, acoustic features are generated from the adapted HSMMs in the output language [89] and an MLSA filter is used to generate the speech waveform.

8.2 Unsupervised cross-lingual adaptation based on a state-level mapping learned using minimum KLD

A cross-lingual adaptation method based on a state-level mapping, learned using the KLD between pairs of states, was proposed by Wu *et al.* [14] and is summarized here. We call this approach "state-level transform mapping".

8.2.1 Learning the mapping between states

For each state ${}^{\forall}j \in [1, J]$ in the output language HMM $\lambda_{\mathrm{output}}$, we search for the state $\widehat{i}$ in the input language HMM λ_{input} with the minimum symmetrized KLD to state j in $\lambda_{\mathrm{output}}$:

$$\widehat{i} = \underset{1 \leq i \leq I}{\operatorname{argmin}} D_{\mathrm{KL}}(j, i), \tag{8.1}$$

where $\lambda_{\mathrm{output}}$ has J states and $D_{\mathrm{KL}}(j, i)$ represents the KLD between state i in λ_{input} and state j in $\lambda_{\mathrm{output}}$ (Figure 8.1). $D_{\mathrm{KL}}(j, i)$ is calculated as [96]:

$$D_{\mathrm{KL}}(j,i) \approx D_{\mathrm{KL}}(j\,||\,i) + D_{\mathrm{KL}}(i\,||\,j), \tag{8.2}$$

$$D_{\mathrm{KL}}(i\,||\,j) = \frac{1}{2}\ln\left(\frac{|\boldsymbol{\Sigma}_j|}{|\boldsymbol{\Sigma}_i|}\right) - \frac{D}{2} + \frac{1}{2}\mathrm{tr}\left(\boldsymbol{\Sigma}_j^{-1}\boldsymbol{\Sigma}_i\right) + \frac{1}{2}(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i), \tag{8.3}$$

where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ represent the mean vector and covariance matrix of the Gaussian pdf associated with state i.

8.2.2 Estimating the transforms for the input language HMM

Next, we estimate a set of *state-dependent* linear transforms $\widehat{\Lambda}$ for the input language HMM λ_{input} in the usual way:

$$\begin{aligned}\widehat{\Lambda} &= \left(\widehat{\boldsymbol{W}}_1, \cdots, \widehat{\boldsymbol{W}}_I\right) \\ &= \underset{\Lambda}{\mathrm{argmax}}\, P(\boldsymbol{O}|\lambda_{\mathrm{input}}, \Lambda)P(\Lambda),\end{aligned} \tag{8.4}$$

where $\boldsymbol{W}_i$ represents a linear transform for state i, I is the number of states in λ_{input}, and $\boldsymbol{O}$ represents the adaptation data. $P(\Lambda)$ represents the prior distribution of the linear transforms, which is a uniform distribution for MLLR and CMLLR and a matrix variate normal distribution for SMAPLR and CSMAPLR [94]. Note that the linear transforms will usually be tied (shared) between groups of states known as regression classes, to avoid over-fitting and to enable adaptation of all states, including those with no adaptation data.

8.2.3 Applying the transforms to the output language HMM

Finally, these transforms are mapped to the output language HMM. The Gaussian pdf in state j of $\lambda_{\mathrm{output}}$ is transformed using the linear transform for state $\widehat{i}$, which is transform $\widehat{\boldsymbol{W}}_{\widehat{i}}$. By transforming all Gaussian pdfs in $\lambda_{\mathrm{output}}$ in this way, cross-lingual speaker adaptation is achieved.

8.2.4 Unsupervised cross-lingual adaptation

We can extend this method to unsupervised adaptation simply by automatically transcribing the input data using ASR-HMMs. For supervised adaptation, λ_{input} and $\lambda_{\mathrm{output}}$ are both TTS-HMMs (for the input and output languages, respectively). For unsupervised adaptation of HMM-based speech synthesis, λ_{input} may be either a TTS-HMM, or an

ASR-HMM that utilizes the same acoustic features as TTS. No other constraints need to be placed on the ASR-HMM. In particular, it does not need to use prosodic-context-dependent-quinphones (which would be necessary for TTS models).

8.3 Experiments

8.3.1 Experimental conditions

We performed experiments on unsupervised English-to-Japanese speaker adaptation for HMM-based speech synthesis. An English speaker-independent model for ASR and average voice model for TTS were trained on the pre-defined training set "SI-84" comprising 7.2k sentences uttered by 84 speakers included in the "short term" subset of the WSJ0 database (15 hours of speech). A Japanese average voice model for TTS was trained on 10k sentences uttered by 86 speakers from the JNAS database (19 hours of speech). One male and one female American English speaker, not included in the training set, were chosen from the "long term" subset of the WSJ0 database as target speakers. The adaptation data comprised 5, 50, or 2000 sentences selected arbitrarily from the 2.3k sentences available for each of the target speakers.

Speech signals were sampled at a rate of 16 kHz and windowed by a 25ms Hamming window with a 10 ms shift for ASR and by an F_0-adaptive Gaussian window with a 5 ms shift for TTS. ASR feature vectors consisted of 39-dimensions: 13 PLP features and their dynamic and acceleration coefficients. TTS feature vectors comprised 138-dimensions: 39-dimension STRAIGHT mel-cepstral coefficients (plus the zeroth coefficient), $\log F_0$, 5 band-filtered aperiodicity measures, and their dynamic and acceleration coefficients. We used 3-state left-to-right triphone HMMs for ASR and 5-state left-to-right context-dependent multi-stream MSD-HSMMs for TTS. Each state had 16 Gaussian mixture components for ASR and a single Gaussian for TTS. For speaker adaptation, the linear transforms $\boldsymbol{W}_i$ had a tri-block diagonal structure, corresponding to the static, dynamic, and acceleration coefficients. Since automatically transcribed labels for unsupervised adaptation contain errors, we adjusted a hyperparameter (τ_b in [94]) of CSMAPLR to higher-than-usual value of 10000 in order to place more importance on the prior (which is a global transform that is less sensitive to transcription errors).

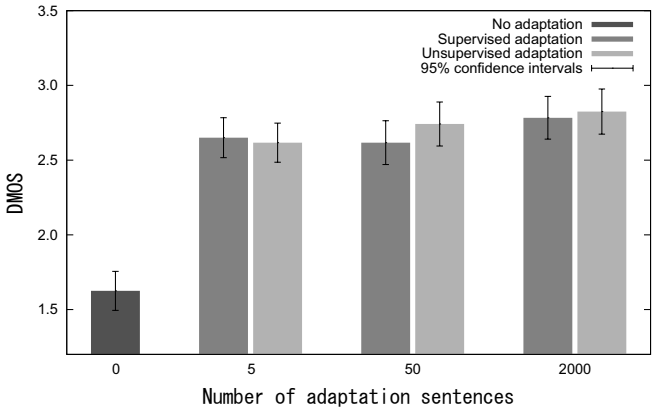


Figure 8.2: Experimental results: comparison of supervised and unsupervised speaker adaptation. "0 sentences" means the unadapted average voice model for the output language.

8.3.2 Listening tests

Synthetic stimuli were generated from 7 models: the average voice model and supervised or unsupervised adapted models each with 5, 50, or 2k sentences of adaptation data. 10 Japanese native listeners participated in the listening test. Each listener was presented with 12 pairs of synthetic Japanese speech samples in random order: the first sample in each pair was a reference original utterance from the database and the second was a synthetic speech utterance generated from one of the 7 models. For each pair, listeners were asked to give an opinion score for the second sample relative to the first (DMOS), expressing how similar the speaker identity was. Since there were no Japanese speech data available for the target English speakers, the reference utterances were English. The text for the 12 sentences in the listening test comprised 6 written Japanese news sentences randomly chosen from the Mainichi corpus and 6 spoken English news sentences from the English adaptation data that had been recognized using ASR then translated into Japanese text using MT.

Figure 8.2 shows the average DMOS and their 95% confidence intervals. First of all, we can see that the adapted voices are judged to sound more similar to target speaker than the average voice. Next, we can see that the differences between supervised and unsupervised adaptation are very small. This is a very pleasing result. However, the effect of the amount of adaptation data is also small, contrary to our expectations. This requires further investigation in future work.

Figure 8.3 shows the average scores using Japanese news texts from the corpus and En-

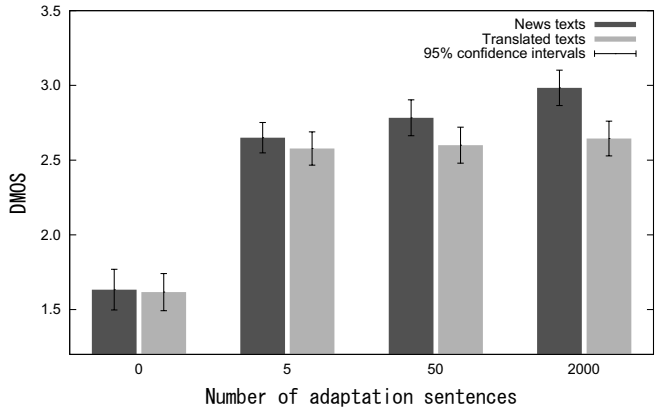


Figure 8.3: Experimental results: comparison of Japanese news texts chosen from the corpus and English news texts which were recognized by ASR then translated into Japanese by MT. "0 sentences" means the unadapted average voice model for the output language.

glish news texts recognized by ASR and translated by MT. It appears that the speaker similarity scores are affected by the text of the sentences. Interestingly the gap becomes larger as the number of adaptation sentences increases; this also deserves further investigation in future work.

8.4 Summary

In this chapter, we described the integration of several techniques we have developed for model adaptation into a single architecture which achieves unsupervised cross-lingual speaker adaptation for HMM-based speech synthesis. The listening tests show very promising results: it has been demonstrated that the adapted voices sound more similar to the target speaker than the average voice and that differences between supervised and unsupervised cross-lingual speaker adaptation are small. It appears that the speaker similarity scores are affected by the text of the sentences, which needs further investigation.

Although all language pairs and directions are possible in our system, only English-to-Japanese adaptation has been evaluated in the perceptual experiments presented here. Evaluation of other language pairs and directions is ongoing. Other future work includes unsupervised cross-lingual speaker adaptation using linear transform estimated directly by ASR-HMMs, which must then use the same acoustic features as TTS-HSMM.

Chapter 9

Conclusions

The present paper described improved acoustic modeling for HMM-based speech recognition and synthesis. Basic theories and fundamental algorithms of the HMM were reviewed in Chapter 2. Statistical speech recognition and synthesis frameworks based on the HMM were presented in Chapters 3 and 4, respectively. In Chapter 5, we constructed a fully consistent HSMM-based speech recognition system and evaluated its performance while avoiding approximations in training, context-clustering, and decoding. The result showed an obvious improvement in phoneme recognition accuracy. Future works include evaluation on speaker independent speech recognition tasks with multi-mixture state output probability distributions. In Chapter 6, we proposed a technique for reducing the footprints of HMM-based speech synthesis systems by tying all covariance matrices. The experimental results showed that the proposed technique efficiently shrinked the footprints of an HMM-based speech synthesis system to less than half of its original size while retaining the quality of synthesized speech. Future work includes applying the technique to full covariance matrices. In Chapter 7, we defined a new integrated model in which linguistic and acoustic models were combined into one model, and all model parameters were estimated simultaneously by the proposed training algorithm. We conducted objective evaluation experiments using phrasing and prosodic models as linguistic models to evaluate the effectiveness of the proposed system. The results demonstrate that the proposed system achieves better F_0 modeling accuracy than that of the conventional system. Future work includes simultaneous training of POS tagging modules and acoustic models. Subjective listening tests performed by native English speakers on a large database are also planned. In Chapter 8, we described the integration of several techniques we have developed for model adaptation into a single architecture which achieves unsupervised cross-lingual speaker adaptation for HMM-based speech synthesis. The listening tests show very promising results: it has been demonstrated that the adapted voices sound more similar to the target speaker than the average voice and that differences between

supervised and unsupervised cross-lingual speaker adaptation are small. It appears that the speaker similarity scores are affected by the text of the sentences, which needs further investigation. Although all language pairs and directions are possible in our system, only English-to-Japanese adaptation has been evaluated in the perceptual experiments presented here. Evaluation of other language pairs and directions is ongoing. Other future work includes unsupervised cross-lingual speaker adaptation using linear transform estimated directly by ASR-HMMs, which must then use the same acoustic features as TTS-HSMM.

Bibliography

[1] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of IEEE*, Vol. 77, pp. 257–285, 1989.

[2] A. Ljolje, J. Hirschberg, and J.P.H. van Santen. Automatic speech segmentation for concatenative inventory selection. pp. 305–311. Springer-Verlag, 1997.

[3] R.E. Donovan and P.C. Woodland. Automatic speech synthesizer parameter estimation using HMMs. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'95*, pp. 640–643, 1995.

[4] R.E. Donovan and E.M. Eide. The IBM trainable speech synthesis system. *Proceedings of International Conference on Spoken Language Processing'98*, Vol. 5, pp. 1703–1706, 1998.

[5] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis. *Proceedings of European Conference on Speech Communication and Technology'99*, Vol. 5, pp. 2347–2350, 1999.

[6] S.E. Levinson. Continuously variable duration hidden Markov models for automatic speech recognition. *Computer, Speech and Language*, Vol. 1, pp. 29–45, 1986.

[7] H. Zen, K. Tokuda, T. Masuko, and T. Kitamura. A hidden semi-Markov model-based speech synthesis system. *IEICE Transactions Information & System*, pp. 825–834, 2007.

[8] J.D. Ferguson. Variable duration models for speech. *Proceedings of the Symposium on the Application Hidden Markov Models to Text and Speech*, pp. 143–179, 1980.

[9] M. Ostendorf, V. Digalakis, and O.A. Kimball. From HMMs to segment models. *IEEE Transactions on Speech & Audio Processing*, Vol. 4, No. 5, pp. 360–378, 1996.

[10] M.J. Russell and A.E. Cook. Experimental evaluation of duration modeling techniques for automatic speech recognition. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'87*, Vol. 1, pp. 2376–2379, 1987.

[11] M.W. Koo, S.J. Park, and D.Y. Son. Context dependent phoneme duration modeling with tree-based state tying. *Proceedings of Interspeech'04*, Vol. 1, pp. 721–724, 2004.

[12] V.R.R. Gadde. Modeling word duration. *Proceedings of International Conference on Signal Processing'00*, Vol. 1, pp. 601–604, 2000.

[13] M. Gibson. Two-pass decision tree construction for unsupervised adaptation of HMM-based synthesis models. *Proceedings of Interspeech'09*, pp. 1791–1794, 2009.

[14] Y.J. Wu and K. Tokuda. State mapping based method for cross-lingual speaker adaptation in HMM-based speech synthesis. *Proceedings of Interspeech'09*, pp. 528–531, 2009.

[15] X.D. Huang, Y. Ariki, and M.A. Jack. *Hidden Markov models mor speech recognition*. Edinburgh University Press, 1990.

[16] Rabiner L. and B.H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, 1993.

[17] S. Young, G. Evermann, M.J.F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland. *The HTK Book (for HTK Version 3.3)*. Cambridge University, 2005.

[18] L. Rabiner, B.H. Juang, S.E. Levinson, and M.M. Sondhi. Recognition of isolated digits using hidden Markov models with continuous mixture densities. *AT&T Technical Journal*, Vol. 64, No. 6, pp. 1211–1234, 1985.

[19] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, Vol. 13, pp. 260–269, 1967.

[20] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistics Society*, Vol. 39, pp. 1–38, 1977.

[21] B.H. Juang. Maximum likelihood estimation for mixture of multivariate stochastic observations of Markov chains. *AT&T Technical Journal*, Vol. 64, No. 6, pp. 1235–1249, 1985.

[22] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. *Proceedings of International Conference Machine Learning*, pp. 591–598, 2000.

[23] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proceedings of International Conference on Machine Learning*, pp. 282–289, 2001.

[24] H.K.J. Kuo and Y. Gao. Maximum entropy dicrect models as a unified model for acoustic models in speech recognition. *Proceedings of International Conference on Spoken Language Processing*, pp. 681–684, 2004.

[25] A. Gunawardana, L. Mahajan, A. Acero, and J.C. Platt. Hidden conditional random fields for phone classification. *Proceedings of European Conference on Speech Communication and Technology*, pp. 1117–1120, 2005.

[26] J.R. Deller, J.H.L. Hansen, and J.G. Proakis. *Discrete-time processing of speech signals*. Macmillan, 1993.

[27] F. Itakura and S. Saito. A statistical method for estimation of speech spectral density and formant frequencies. *Transactions of Institute of Electronics, Information and Communication Engineering*, Vol. J53-A, pp. 35–42, 1970.

[28] J.D. Markel and A.H. Gray Jr. *Linear prediction of speech*. Springer-Verlag, 1976.

[29] A.V. Oppenheim and R.W. Schafer. *Digital signal processing*. Englewood Cliffs, 1975.

[30] T. Fukada, K. Tokuda, Kobayashi T., and S. Imai. An adaptive algorithm for mel-cepstral analysis of speech. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'92*, Vol. 1, pp. 137–140, 1992.

[31] S.B. Gavis and P. Mermelstein. Comparison of parameteric representations for monosylabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, & Signal Processing*, Vol. 28, pp. 357–366, 1980.

[32] H. Hermansky. Perceptual linear prediction (PLP) of speech. *Journal of the Acoustic Society of America*, Vol. 87, No. 4, pp. 1738–1752, 1990.

[33] S. Sagayama and F. Itakura. On individuality in a dynamic measure of speech. *Proceedings of Spring Conference of Acoustic Society of Japan*, pp. 589–590, 1979. (in Japanese).

[34] S. Furui. Speaker independent isolated word recognition using dynamic features of speech spectrum. *IEEE Transactions Acoustics, Speech, & Signal Processing*, Vol. 34, pp. 52–59, 1986.

[35] C.H. Lee and E. Giachin. Improved acoustic modeling for speaker independent large vocabulary continuous speech recognition. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, pp. 161–164, 1991.

[36] J.G. Wilpon, C.H. Lee, and L. Rabiner. Improvements in connected digit recognition using higher order spectral and energy features. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, Vol. 1, pp. 349–352, 1991.

[37] S. Sagayama. Phoneme environment clustering for speech recognition. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 397–400, 1989.

[38] K.F. Lee, S. Hayamizu, H.W. Hon, C. Huang, J. Swartz, and R. Weide. Allophone clustering for continuous speech recognition. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing*, pp. 749–752, 1990.

[39] J.J. Odell. *The use of context in large vocabulary speech recognition*. PhD thesis, Cambridge University, 1995.

[40] H.J. Nock, M.J.F. Gales, and S.J. Young. A comparative study of methods for phonetic decision-tree state clustering. *Proceedings of European Conference on Speech Communication and Technology*, Vol. 1, pp. 111–114, 1997.

[41] S. Gao, J.S. Zhang, S. Nakamura, C.H. Lee, and T.S. Chu. Weighted graph based decision tree optimization for high accuracy acoustic modeling. *Proceedings of International Conference on Spoken Language Processing*, pp. 1233–1236, 2002.

[42] S.M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech, & Signal Processing*, Vol. 35, No. 3, pp. 400–411, 1987.

[43] H. Ney, D. Mergel, A. Noll, and A. Paelser. Data-driven search organisation for continuous speech recognition. *IEEE Transactions on Signal Processing*, Vol. 40, pp. 272–281, 1992.

[44] F. Jelinek. Fast sequential decoding algorithm using a stack. *IBM Journal of Research and Development*, Vol. 13, pp. 675–685, 1969.

[45] R. Sproat, J. Hirschberg, and D. Yarowsky. A corpus-based synthesizer. *Proceedings of International Conference on Spoken Language Processing*, pp. 563–566, 1992.

[46] A.W. Black and P. Taylor. CHATR: a generic speech synthesis system. *Proceedings of COLING94*, pp. 983–986, 1994.

[47] A. Hunt and A.W. Black. Unit selection in a concatenative speech synthesis system using a large speech database. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'96*, Vol. 1, pp. 373–376, 1996.

[48] A.W. Black and P. Taylor. The Festival speech synthesis system: system documentation. Technical Report HCRC/TR-83, University of Edinburgh, 1997.

[49] K. Tokuda, T. Kobayashi, and S. Imai. Speech parameter generation from HMM using dynamic features. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'95*, pp. 660–663, 1995.

[50] K. Tokuda, T. Masuko, Y. Yamada, T. Kobayashi, and S. Imai. An algorithm for speech parameter generation from continuous mixture HMMs with dynamic features. *Proceedings of European Conference on Speech Communication and Technology'95*, pp. 757–760, 1995.

[51] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'00*, Vol. 3, pp. 1315–1318, 2000.

[52] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi. Adaptation of pitch and spectrum for HMM-based speech synthesis using MLLR. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'01*, Vol. 2, pp. 805–808, 2001.

[53] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Speaker interpolation in HMM-based speech synthesis system. *Proceedings of European Conference on Speech Communication and Technology'97*, Vol. 5, pp. 2523–2526, 1997.

[54] K. Shichiri, A. Sawabe, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Eigenvoices for HMM-based speech synthesis. *Proceedings of International Conference on Spoken Language Processing'02*, pp. 1269–1272, 2002.

[55] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Duration modeling for HMM-based speech synthesis. *Proceedings of International Conference on Spoken Language Processing'98*, Vol. 2, pp. 29–32, 1998.

[56] K. Koishida, K. Tokuda, T. Masuko, and T. Kobayashi. Vector quantization of speech spectral parameters using statistics of dynamic features. *Proceedings of International Conference on Signal Processing'97*, pp. 247–252, 1997.

[57] Y. Ishimatsu, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Investigation of state duration model based gamma distribution for HMM-based speech synthesis. *IEICE technical report (J)*, Vol. 101, No. 352, pp. 57–62, 2001.

[58] K.F. Lee. Context-dependent phonetic hidden Markov models for speaker-independent continuous speech recognition. *IEEE Transactions on Acoustics, Speech, & Signal Processing*, Vol. 38, No. 4, pp. 599–609, 1990.

[59] M. Mohri, F. Pereira, and M. Riley. Weighted finite-state transducers in speech recognition. *Proceedings of ASR'00*, pp. 97–106, 2000.

[60] C. Allauzen and M. Mohri. Generalized optimization algorithm for speech recognition transducers. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'03*, Vol. 1, pp. 352–355, 2003.

[61] K. Shinoda and T. Watanabe. Acoustic modeling based on the MDL criterion for speech recognition. *Proceedings of European Conference on Speech Communication and Technology'97*, pp. 99–102, 1997.

[62] S.Z. Yu and H. Kobayashi. An efficient forward-backward algorithm for an explicit-duration hidden Markov model. *IEEE Signal Processing Letters*, Vol. 10 (1), pp. 11–14, 2003.

[63] Nakagawa. S. and H. Takagi. Statistical methods for comparing pattern recognition algorithms and comments on evaluating speech recognition performance. *The Journal of Acoustical Society of Japan (J)*, Vol. 50, No. 10, pp. 849–854, 1994.

[64] T. Masuko, K. Tokuda, T. Kobayashi, and S. Imai. Speech synthesis from HMMs using dynamic features. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'96*, pp. 389–392, 1996.

[65] J. Takami and S. Sagayama. A successive state splitting algorithm for efficient allophone modeling. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'92*, pp. 573–576, 1992.

[66] P. Woodland and S. Young. Benchmark darpa rm results with the HTK portable HMM toolkit. *Proceeding of DARPA Continuous Speech Recognition Workshop*, pp. 71–76, 1992.

[67] M.Y. Hwang, X. Huang, and F. Alleva. Predicting unseen triphones with senones. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'93*, pp. 311–314, 1993.

[68] M. Ostendorf and H. Singer. HMM topology design using maximum likelihood successive state splitting. *Computer Speech & Language*, Vol. 1, No. 1, pp. 17–41, 1997.

[69] K. SHINODA and T. Watanabe. MDL-based context-dependent subword modeling for speech recognition. *The Journal of the Acoustical Society of Japan (E)*, Vol. 21, No. 2, pp. 79–86, 2000.

[70] W. Chou and W. Reichl. Decision tree state tying based on penalized bayesain information criterion. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'99*, pp. 345–348, 1999.

[71] S. Watanabe, Y. Minami, A. Nakamura, and N. Ueda. Training of shared states in hidden Markov model based on bayesian approach. *IEICE technical report. Speech*, Vol. 102, No. 35, pp. 43–48, 2002.

[72] T. Kato, S. Kuroiwa, T. Shimizu, and N. Higuchi. Tree-based clustering for gaussian mixture HMMs. *Transactions of Institute of Electronics, Information and Communication Engineering*, Vol. J83-D-II, No. 11, pp. 2128–2136, 2000.

[73] H.J. Nock. Context clustering for triphone-based speech recognition. *Master Thesis, Cambridge University*, 1996.

[74] M.J.F. Gales. Semi-tied covariance matrices for hidden Markov models. *IEEE Transactions on Speech & Audio Processing*, Vol. 7, No. 3, pp. 272–281, 1999.

[75] A. Kurematsu, K. Takeda, Y. Sagisaka, S. Katagiri, H. Kuwabara, and K. Shikano. ATR Japanese speech database as a tool of speech recognition and synthesis. *Speech Communication*, Vol. 9, pp. 357–363, 1990.

[76] K. Tokuda, T. Kobayashi, T. Chiba, and S. Imai. Spectral estimation of speech by mel-generalized cepstral analysis. *Transactions of Institute of Electronics, Information and Communication Engineering*, Vol. 75-A, No. 7, pp. 1124–1134, 1992.

[77] H. Kawahara. Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based f0 extraction : Possible role of a repetitive structure in sounds. *Speech Communication*, Vol. 27, pp. 187–207, 1999.

[78] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi. Hidden Markov models based on multi-space probability distribution for pitch pattern modeling. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'99*, pp. 229–232, 1999.

[79] T. Toda and K. Tokuda. Speech parameter generation algorithm considering global variance for HMM-based speech synthesis. *Proceedings of Interspeech'05*, pp. 2801–2804, 2005.

[80] K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg. ToBI: A standard for labeling English prosody. *Proceedings of the 1992 International Conference on Spoken Language Processing*, pp. 867–870, 1992.

[81] A.W. Black and A.J. Hunt. Generating F_0 contours from ToBI labels using linear regression. *Proceedings of International Conference on Signal Processing'96*, pp. 1385–1388, 1996.

[82] K. Dusterhoff, A.W. Black, and P. Taylor. Using decision trees within the tilt intonation model to predict F_0 contours. *Proceedings of European Conference on Speech Communication and Technology'99*, pp. 1627–1630, 1999.

[83] X. Sun. F_0 generation for speech synthesis using a multi-tier approach. *Proceedings of International Conference on Signal Processing'02*, pp. 2077–2080, 2002.

[84] S. Sakai. Additive modeling of English F_0 contour for speech synthesis. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'05*, Vol. 1, pp. 277–280, 2005.

[85] F.H. Liu, L. Gu, Y. Gao, and M. Picheny. Use of statistical N-gram models in natural language generation for machine translation. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'03*, pp. 636–639, 2003.

[86] Effective Multilingual Interaction in Mobile Environments (The FP7 EMIME project). `http://www.emime.org`.

[87] Y. Gao. Coupling vs. unifying: Modeling techniques for speech-to-speech translation. *Proceedings of European Conference on Speech Communication and Technology'03*, pp. 365–368, 2003.

[88] H. Ney. Speech translation: coupling of recognition and translation. *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'99*, pp. 517–520, 1999.

[89] J. Yamagishi, T. Nose, H. Zen, L. Zhen-Hua, T. Toda, K. Tokuda, S. King, and S. Renals. A robust speaker-adaptive HMM-based text-to-speech synthesis. *IEEE TSALP*, Vol. 17, No. 6, pp. 1208–1230, 2009.

[90] J. Dines, J. Yamagishi, and S. King. Measuring the gap between HMM-based ASR and TTS. *Proceedings of Interspeech'09*, pp. 1391–1394, 2009.

[91] S. King, K. Tokuda, H. Zen, and J. Yamagishi. Unsupervised adaptation for HMM-based speech synthesis. *Proceedings of Interspeech'08*, pp. 1869–1872, 2008.

[92] J. Yamagishi, B. Usabaev, S. King, O. Watts, J. Dines, J. Tian, R. Hu, K. Oura, K. Tokuda, R. Karhila, and M. Kurimo. Thousands of voices for HMM-based speech synthesis. *Proceedings of Interspeech'09*, pp. 420–423, 2009.

[93] M.J.F. Gales. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech & Language*, Vol. 12, No. 2, pp. 75–98, 1998.

[94] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, and J. Isogai. Analysis of speaker adaptation algorithms for HMM-based speech synthesis and a constrained smaplr adaptation algorithm. *IEEE Transactions on Speech, Audio & Language Processing*, Vol. 17, No. 1, pp. 66–83, 2009.

[95] T. Hirsimaki, J. Pylkkonen, and M. Kurimo. Importance of high-order N-gram models in morph-based speech recognition. *IEEE Transactions on Speech, Audio & Language Processing*, Vol. 17, No. 4, pp. 724–732, 2009.

[96] Y. Qian, H. Lang, and F.K. Soong. A cross-language state sharing and mapping approach to bilingual (Mandarin – English) TTS. *IEEE Transactions on Speech, Audio & Language Processing*, Vol. 17, No. 6, pp. 1231–1239, 2009.

List of Publications

Journal papers

[1] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "A Fully Consistent Hidden Semi-Markov Model-Based Speech Recognition System," *IEICE Transactions on Information & Systems*, vol. E91-D, no. 11, pp. 2693–2700, Nov. 2008.

[2] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "A Covariance Tying Technique for HMM-based Speech Synthesis," *IEICE Transactions on Information & Systems*, vol. E93-D, no. 3, May 2010 (to be published).

[3] Junichi Yamagishi, Bela Usabaev, Simon King, Oliver Watts, John Dines, Jilei Tian, Young Guan, Rile Hu, **Keiichiro Oura**, Yi-Jian Wu, Keiichi Tokuda, Reima Karhila, and Mikko Kurimo, "Thousands of Voices for HMM-based Speech Synthesis - Analysis and Application of TTS Systems Build on Various ASR Corpora," *IEEE Transactions on Audio, Speech & Language Processing* (conditional acceptance).

International conference proceedings

[4] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "Hidden Semi-Markov Model Based Speech Recognition System Using Weighted Finite-State Transducer," *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'06*, vol. 1, pp. 33–36, May 2006.

[5] **Keiichiro Oura**, Yoshihiko Nankaku, Tomoki Toda, Keiichi Tokuda, Rannierry Maia, Shinsuke Sakai, and Satoshi Nakamura, "Simultaneous Acoustic, Prosodic, and Phrasing Model Training for TTS Conversion Systems," *Proceedings of International Symbosium on Chinise Spoken Language Processing'08*, SPE1.1, pp. 1–4, Dec. 2008.

[6] **Keiichiro Oura**, Yi-Jian Wu, and Keiichi Tokuda, "Overview of NIT HMM-based speech synthesis system for Blizzard Challenge 2009," *Proceedings of Blizzard Challenge 2009*, Sep. 2009.

[7] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "Tying covariance matrices to reduce the footprint of HMM-based speech synthesis systems," *Proceedings of Interspeech 2009*, pp. 1759–1761, Sep. 2009.

[8] Junichi Yamagishi, Bela Usabev, Simon King, Oliver Watts, John Dines, Jilei Tian, Rile Hu, **Keiichiro Oura**, Keiichi Tokuda, Reima Karhila, and Mikko Kurimo, "Thousands of voices for HMM-based speech synthesis," *Proceedings of Interspeech 2009*, pp. 420–423, Sep. 2009.

[9] Heiga Zen, **Keiichiro Oura**, Takashi Nose, Junichi Yamagishi, Shinji Sako, Tomoki Toda, Takashi Masuko, Alan W. Black, and Keiichi Tokuda, "Recent development of the HMM-based speech synthesis system (HTS)," *Proceedings of Asia-Pacific Signal and Information Processing Association'09*, pp. 121–130, Oct. 2009.

[10] **Keiichiro Oura**, Junichi Yamagishi, Simon King, Mirjam Wester, and Keiichi Tokuda, "Unsupervised cross-lingual speaker adaptation for HMM-based speech synthesis," *Proceedings of International Conference on Acoustics, Speech, and Signal Processing'10*, May 2010 (to be published).

Technical reports

[11] Heiga Zen, **Keiichiro Oura**, Takashi Nose, Junichi Yamagishi, Shinji Sako, Tomoki Toda, Takashi Masuko, Alan W. Black, and Keiichi Tokuda, "Recent developments

of the HMM-based speech synthesis system (HTS)," *Technical Report of IEICE*, vol. 2007, no. 129, pp. 301–306, Dec. 2007.

[**12**] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "Tying covariance for HMM-based speech synthesis," *Technical Report of IEICE*, vol. 108, no. 337, pp. 215–220, Dec. 2008.

[**13**] **Keiichiro Oura**, Junichi Yamagishi, Mirjam Wester, Simon King, and Keiichi Tokuda, "Unsupervised speaker adaptation for Speech-to-Speech Translation system," *Technical Report of IEICE*, vol. 109, no. 356, pp. 13–18, Dec. 2009.

Domestic conference proceedings

[**14**] **Keiichiro Oura**, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "HSMM Based Speech Recognition Using WFST," *Proceedings of Autumn Meeting of the ASJ*, vol. I, 2-7-15, pp. 87–88, Sep. 2005.

[**15**] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "Postfiltering for HMM-based speech synthesis using Mel-LSPs," *Proceedings of Autumn Meeting of the ASJ*, vol. I, 3-4-9, pp. 367–368, Sep. 2007.

[**16**] **Keiichiro Oura**, Yoshihiko Nankaku, Tomoki Toda, Keiichi Tokuda, Rannierry Maia, Shinsuke Sakai, and Satoshi Nakamura, "Simultaneous prosody and acoustic model training for english speech synthesis," *Proceedings of Spring Meeting of the ASJ*, vol. I, 2-11-1, pp. 335-336, Mar. 2008.

[**17**] **Keiichiro Oura**, Heiga Zen, Yoshihiko Nankaku, Akinobu Lee, and Keiichi Tokuda, "Tying variance for HMM-based speech synthesis," *Proceedings of Autumn Meeting of the ASJ*, vol. I, 2-P-29, pp. 421–422, Sep. 2008.

[**18**] **Keiichiro Oura**, Junichi Yamagishi, Simon King, Mirjam Wester, and Keiichi Tokuda, "Unsupervised English-to-Japanese speaker adaptation for HMM-based

speech synthesis," *Proceedings of Autumn Meeting of the ASJ*, vol. I, 3-P-18, pp. 401–402, Oct. 2009.

[**19**] Xianglin Peng, **Keiichiro Oura**, Yoshihiko Nankaku, and Keiichi Tokuda, "Cross-lingual speaker adaptation considering differences between language-dependent average voices," *Proceedings of Spring Meeting of the ASJ*, vol. I, 1-7-21, May 2010 (to be published).

[**20**] Makoto Yamada, Nobuyuki Nishizawa, Tuneo Kato, **Keiichiro Oura**, and Keiichi Tokuda, "Subjective evaluations of speech rate control methods for HMM-based speech synthesis," *Proceedings of Spring Meeting of the ASJ*, vol. I, 1-P-2, May 2010 (to be published).

[**21**] **Keiichi Oura**, Shinji Sako, and Keiichi Tokuda, "Japanese Text-to-Speech Synthesis System: Open JTalk," *Proceedings of Spring Meeting of the ASJ*, vol. I, 2-7-6, May 2010 (to be published).

[**22**] Ayami Mase, **Keiichiro Oura**, Yoshihiko Nankaku, and Keiichi Tokuda, "HMM-based singing voice synthesis system trained by using pitch-shifted pseudo data," *Proceedings of Spring Meeting of the ASJ*, vol. I, 2-7-7, May 2010 (to be published).

[**23**] Satoru Muto, **Keiichiro Oura**, Yoshihiko Nankaku, and Keiichi Tokuda, "Reducing computational cost of training for HMM-based singing voice synthesis using note boundaries," *Proceedings of Spring Meeting of the ASJ*, vol. I, 2-7-8, May 2010 (to be published).

Other

[**24**] Koichiro Kawashima, Tatsuya Tomioka, Osamu Terakura, Tomohiro Hirano, Yohei Yamaguchi, Taizo Umezaki, **Keiichiro Oura**, and Yukio Niiri, "Nitech future based on inauguration of business," *GOKISO*, no. 432, pp. 12–23, Nov. 2009.

[**25**] **Keiichiro Oura**, Heiga Zen, Shinji Sako, and Keiichi Tokuda, “Construction of speech synthesis systems using HTS,” *Journal of Human Interface*, Feb. 2010 (to be published).

Appendix A

Coverage

2008年05月02日 中日新聞 朝刊1面

非公表

Figure A.1: News paper of CHUNICHI (Jun 2th, 2008)

2008年09月22日 日刊工業新聞 9面

非公表

Figure A.2: News paper of NIKKAN KOGYO (Sep. 22th, 2008)

2008年09月22日 日経産業新聞 3面

非公表

Figure A.3: News paper of NIKKEN SANGYO (Sep. 22th, 2008)

Figure A.4: Yahoo Japan! (Sep. 22th, 2008)

Appendix B

Software

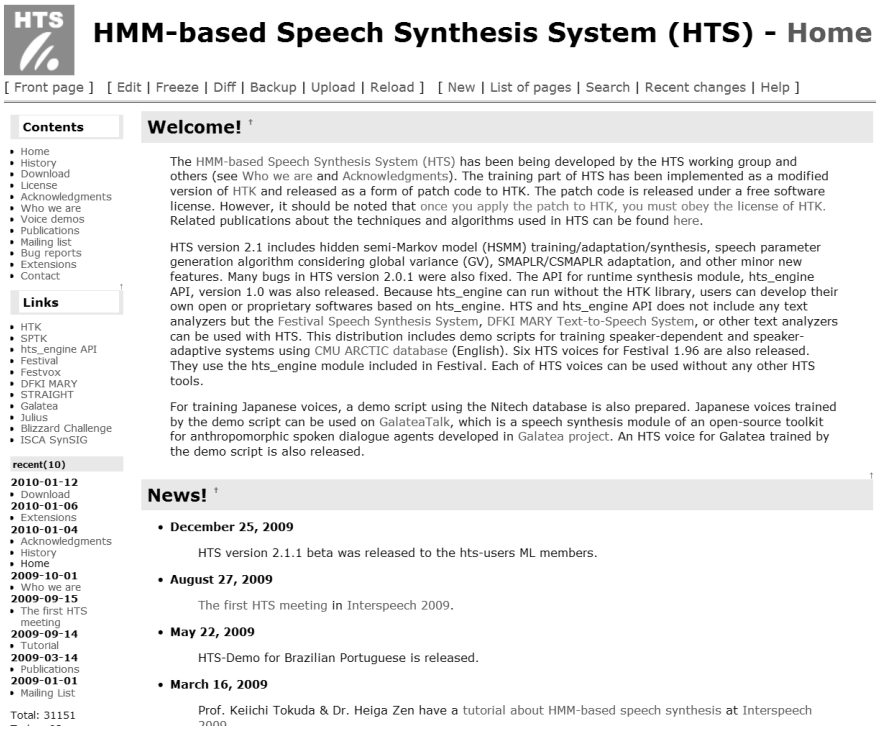


Figure B.5: HTS: http://hts.sp.nitech.ac.jp/

hts_engine API

What is hts_engine API?

The hts_engine is software to synthesize speech waveform from HMMs trained by the HMM-based speech synthesis system (HTS).
This software is released under the New and Simplified BSD license.

Getting hts_engine API

hts_engine API version 1.02 (December 25, 2009)

Runtime HMM-based speech synthesis engine API and its applications.

- Source Code: hts_engine_API-1.02.tar.gz
- Documentation: hts_engine_API_readme-1.02.txt
- Reference Manual: hts_engine_API_reference-1.02.pdf

Flite+hts_engine version 1.00 (December 25, 2009)

English TTS system "Flite+hts_engine" consists of Flite and hts_engine API.

- Source Code: flite+hts_engine-1.00.tar.gz
- Documentation: flite+hts_engine_readme-1.00.txt

HTS voice version 1.01 (December 25, 2009)

HTS voice for hts_engine API trained by using CMU ARCTIC database.

- Binary Package: hts_voice_cmu_us_arctic_slt-1.01.tar.gz
- Documentation: hts_voice_cmu_us_arctic_slt_readme-1.01.txt

Mailing Lists

hts-engine-users@lists.sourceforge.net

Attention!!
"HTS" and "HTS demo script" are not supported in this mailing list.

Figure B.6: hts_engine API: http://hts-engine.sourceforge.net/

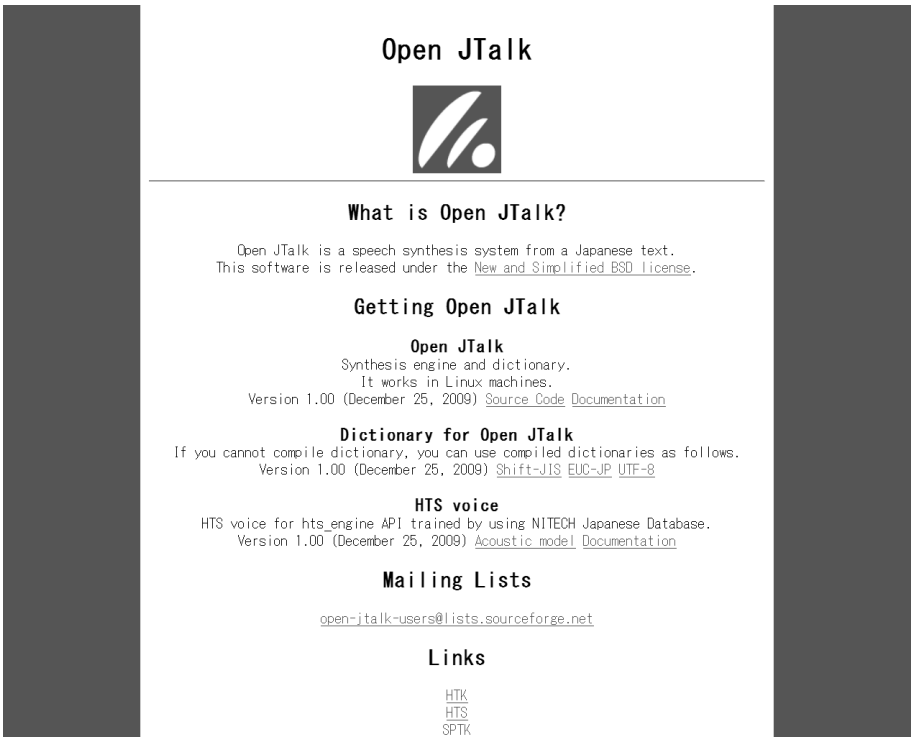


Figure B.7: Open JTalk: http://open-jtalk.sourceforge.net/