DOCTORAL DISSERTATION

A STATISTICAL APPROACH TO SPEECH SYNTHESIS AND IMAGE RECOGNITION BASED ON HIDDEN MARKOV MODELS

(隠れマルコフモデルに基づく
音声合成と画像認識のための統計的アプローチ)

DOCTOR OF ENGINEERING

JANUARY 2018

Kei SAWADA

Supervisor : Dr. Yoshihiko NANKAKU
Dr. Keiichi TOKUDA

Department of Scientific and Engineering Simulation
Nagoya Institute of Technology

Abstract

We human beings communicate with others by transmitting auditory and visual information. Recently, since information technology has steadily improved, not only communication between humans, but also communication between humans and computers has become feasible. In consequence, new services such as spoken dialogue system and biometrics authentication system have been developed. In order to realize communication between humans and computers, the computer need to interpret auditory and visual information. Speech recognition and speech synthesis are used as techniques for processing auditory information, and image recognition and image synthesis are used as techniques for processing visual information. Improvements in these techniques are necessary for smooth communication between humans and computers.

Hidden Markov models (HMM)-based speech recognition and speech synthesis have been proposed as a standard framework. HMMs are one of widely used statistical models for representing time series by well-defined algorithms. Additionally, two-dimensional data such as pixel values of image can be modeled by extending the HMM to two dimensions. In this paper, I propose speech synthesis and image recognition based on HMMs to realize smooth communication between humans and computers. Especially, for widening the communication, I investigate highly versatile construction methods from low-resource data.

For speech synthesis, I propose a method for constructing text-to-speech (TTS) systems for languages with unknown pronunciations. There are thousands of active written languages in the world. However, conventional methods of constructing corpus-based TTS systems for a new language not only require preparation of training corpus but also require language-specific knowledge. Especially, to marshal language-specific knowledge about pronunciation for each new language requires high cost. Therefore, a goal of the speech synthesis research is to establish a language-independent framework that can be used to construct TTS systems for any written language. To address this problem, I investigate a framework for automatically constructing a TTS system from a target language database consisting of only speech data and corresponding Unicode texts. In the proposed method, pseudo phonetic information of the target language with unknown pronunciation is obtained by a speech recognizer of a rich-resource proxy language. Then, a grapheme-to-phoneme converter and a statistical parametric speech synthesizer are constructed based on the obtained pseudo phonetic information. With these processes, it becomes possible to construct a TTS system automatically without specific knowledge on the target language.

For image recognition, I propose an image recognition method based on hidden Markov eigen-image models (HMEMs) using a Bayesian framework. The geometric variations of the object to be recognized, e.g., size, location, and rotation, are an essential problem in image recognition. Separable lattice hidden Markov models (SL-HMMs), which have been proposed to reduce the effect of geometric variations, can perform elastic matching both horizontally and vertically. However, SL-HMMs still have a limitation in that the images are assumed to be generated independently from corresponding HMM states. It is insufficient to represent variations in images, e.g., lighting conditions and object deformation. To overcome this problem, HMEMs have been proposed in which the structure of factor analysis (FA) or probabilistic principal component analysis (PPCA) is integrated into SL-HMMs. HMEMs have good properties of both SL-HMMs and FA/PPCA: invariances to the size and location of objects to be recognized and a linear feature extraction. In some image recognition tasks, it is difficult to acquire sufficient training data. Additionally, models with a complex structure such as HMEMs suffer from the over-fitting problem, especially when there is insufficient training data. This study aims to accurately estimate HMEMs using the variational Bayesian (VB) method. The VB method can utilize prior distributions representing useful prior information and is expected to have a high generalization ability due to the marginalization of model parameters. Furthermore, to relax the local maximum problem in the VB method, the deterministic annealing expectation maximization algorithm is applied to train HMEMs. Experiments on face recognition indicated that the proposed method offers a significantly improved image recognition performance.

As described above, in this paper, I propose a statistical approach to speech synthesis and image recognition based on HMMs, and they are evaluated in experiments.

Keywords: Speech synthesis, image recognition, hidden Markov model

Abstract in Japanese

我々人間は，聴覚・視覚情報の伝達により他者とのコミュニケーションを行う．近年では情報技術の発達により，人と人のコミュニケーションのみでなく，人とコンピュータによるコミュニケーションが実現可能となってきた．これにより，音声対話システムや認証システムなど新たなサービスが開発されている．人とコンピュータのコミュニケーションを実現するためには，コンピュータが聴覚・視覚情報を解釈する必要がある．聴覚情報を処理する技術として音声認識や音声合成が，視覚情報を処理する技術として画像認識や画像合成が挙げられる．人とコンピュータの円滑なコミュニュケーションのために，これらの技術の改善が求められている．

音声認識や音声合成の代表的な枠組みとして，隠れマルコフモデル (hidden Markov model; HMM) を用いた手法が提案されている．HMM は，音声データのような時系列のデータをモデル化するのに適しており，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，トポロジーを対象に応じて設計可能，現実的な計算量で学習可能などの特徴がある．さらに，HMM を 2 次元に拡張することで画像データのような 2 次元データもモデル化することができる．本論文では，円滑な人とコンピュータのコニュニケーションを実現するために，HMM に基づく音声合成と画像認識についての検討を行う．特に，少量の学習データから汎用性が高い手法の構築することにより，コミュニケーションの幅を広げる．

音声合成においては，発音情報が未知の言語におけるテキスト音声合成システムの構築法を提案する．世界には数千におよぶ書記言語が存在すると考えられており，あらゆる書記言語のテキスト音声合成 (text-to-speech; TTS) システムを構築することは，音声合成研究の 1 つのゴールである．しかし，一般的な TTS システム構築法は，目的とする言語に関する専門的な知識を用いた人手による作業を必要とし，言語ごとに高い構築コストがかかる．そこで，本論文では発音情報が未知である言語の音声データと Unicode テキストのみから構成されるデータベースから，言語に関する専門的な知識を利用せずに TTS システムを自動構築する手法について検討する．提案法では，発音情報が未知であるターゲット言語の発音情報を代理言語の音声認識により獲得する．そして，疑似発音情報に基づき書記素音素変換器と統計的音声合成器を構築する．これにより，ターゲット言語固有の知識を利用することなく TTS

システムを構築することが可能となる．

画像認識においては，ベイズ基準に基づく可変固有画像モデル (hidden Markov eigen-image models; HMEM) を提案する．画像認識において，認識対象の位置や大きさなどの幾何学的変動に対応可能な分離型格子 HMM に固有画像のような主成分分析の構造を組み込んだ HMEM が提案されている．従来，HMEM の学習には尤度最大化基準が用いられてきた．しかし，画像認識では十分な量の学習データを用いることが困難である場合も多く，このような場合に，尤度最大化基準により HMEM のような複雑なモデル構造を学習すると過学習を起こす恐れがある．そこで，本論文では，ベイズ基準に基づく高精度な HMEM の学習を提案する．ベイズ基準は，事前情報を事前分布として用いて事後分布を推定することにより過学習の緩和が期待できる．さらに，確定的アニーリング期待値最大化アルゴリズムを導入することで，初期値に依存した局所最適解の問題を克服する．

以上のように，本論文では，人とコンピュータのコミュニケーションの実現のために，HMM に基づく高性能な音声合成システムと画像認識システムを構築するための統計的アプローチを提案し，評価実験により有効性を検証する．

Acknowledgement

I would like to thank Yoshihiko Nankaku and Keiichi Tokuda for his support, encouragement, and guidance.

I am grateful to Akinobu Lee, Keiichiro Oura, and Kei Hashimoto for their technical supports and helpful discussions. Special thanks go to all the members of Tokuda and Nankaku laboratory and Lee laboratory for creating such a friendly and stimulating place to work. I would be remiss if I did not thank Natsuki Kuromiya, a secretary of the laboratory, for their kind assistance.

This research and development work was partly supported by Grant-in-aid for JSPS Fellows Grant Number 15J08391 and the Hori Sciences and Arts Foundation.

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

Contents

List of Tables

List of Figures

Chapter 1

Introduction

We human beings communicate with others by transmitting auditory and visual information. Recently, since computer hardware and information technology have steadily improved, not only communication between humans, but also communication between humans and computers has become feasible. In consequence, new services such as spoken dialogue system and biometrics authentication system have been developed. The emergence of these services will enrich our lives.

In order to realize communication between humans and computers, the computer need to interpret auditory and visual information. Speech recognition and speech synthesis are used as techniques for processing auditory information, and image recognition and image synthesis are used as techniques for processing visual information. Therefore, improvements in these techniques are necessary for smooth communication between humans and computers.

Hidden Markov models (HMM)-based speech recognition and speech synthesis have been proposed as a standard framework for computers to process auditory information. HMMs are one of widely used statistical models for representing time series by well-defined algorithms. Additionally, two-dimensional data such as pixel values of image can be modeled by extending the HMM to two dimensions. In this paper, I propose speech synthesis and image recognition based on HMMs to realize smooth communication between humans and computers. Especially, for widening the communication, I investigate highly versatile construction methods from low-resource data.

For speech synthesis, I propose a method for constructing text-to-speech (TTS) systems for languages with unknown pronunciations. A number of studies on TTS systems have been conducted. Consequently, the quality of synthetic speech has improved, and TTS systems are now used in various applications, such as in-car navigation, spoken dialogue, and speech translation systems. Accordingly, the demand for TTS systems offering high-quality synthetic speech, various speaking styles, and various languages is increasing. There are thousands of active written languages in the world [1]. Construction of a TTS system for a new language leads to increased use of applications. TTS systems for low-resource languages are in great demand because speech translation systems are very useful applications for low-resource languages. However, conventional methods of constructing corpus-based TTS systems for a new language not only require preparation of training corpus but also require language-specific knowledge. Especially, to marshal language-specific knowledge about pronunciation for each new language requires high cost. Therefore, a goal of the speech synthesis research is to establish a language-independent framework that can be used to construct TTS systems for any written language.

To construct a TTS system for a new language, it is necessary to marshal language-dependent elements, e.g., to define a phoneset and linguistic features, such as accents and parts-of-speech, for each language. However, doing so requires language-specific knowledge. Therefore, a low language-dependency framework is needed in order to construct TTS systems for new languages. In this study, I focus on automatic construction of a TTS system without knowledge specific to the language with the unknown pronunciation. I construct a TTS system from a database consisting of the only speech data and Unicode [2] texts corresponding to speech data. The problem in this situation is that a phoneset, phonetic information corresponding to speech data, and a lexicon do not exist. To solve these phoneset and phonetic information problems, speech recognition is carried out by using the speech recognizer of a rich-resource proxy language. Pseudo phoneme sequences of the target language speech data are obtained from the speech recognition results. An statistical parametric speech synthesis (SPSS)-based speech synthesizer of the target language is then trained from speech data and pseudo phoneme sequence pairs. To solve the lexicon problem, I train a grapheme-to-phoneme converter based on joint-sequence models [3] from text and pseudo phoneme sequence pairs. The joint-sequence model is a N-gram model that models a joint-sequence in which grapheme and phoneme sequences are aligned. The model can estimate a phoneme sequence with the highest likelihood from a grapheme sequence. In addition, in order to improve quality of synthesized speech, I propose improvement of the speech recognizer and estimation of the phoneme sequence considering phoneme duration. With these processes, it becomes possible to construct a TTS system automatically without specific knowledge on the target language.

For image recognition, I propose an image recognition method based on hidden Markov eigen-image models (HMEMs) using a Bayesian framework. Image recognition is a technique for identifying objects in an image. Typical applications include biometrics authentication, e.g., fingerprint and face, optical character recognition (OCR), and general object recognition. As computer processing power increases, machine learning approaches based on statistical learning theory have been successfully applied in the field of image recognition. Moreover, not only applying general statistical classifier, approaches considering the specific problems of image recognition, e.g., geometric variations such as size, location, and rotation, image size variations, lighting conditions, object deformation, and occlusion, have been actively studied.

Among the specific problems of image recognition, geometric variations of an object to be recognized are a serious problem in image recognition. Therefore, much research work has been conducted on this problem. These can broadly be divided into three approaches: 1) task-dependent normalization techniques, 2) local features, and 3) the integration of geometric invariants into model structures. For approach 3), HMM based techniques, which integrate geometric invariants into model structures, have been proposed [4, 5]. Geometric matching between input images and model parameters is represented by discrete hidden variables and the normalization process is included in the calculation of probabilities. However, the extension of HMMs to two dimensions for two-dimensional data, e.g., pixel values of an image, generally leads to an exponential increase in the amount of computation needed for training. To overcome this problem, several low computational complexity HMM structures have been proposed [6–12]. Among them, separable lattice HMMs (SL-HMMs) have been proposed to reduce computational complexity while retaining outstanding properties that model two-dimensional data [12]. SL-HMMs can perform elastic matching in both the horizontal and vertical directions, which makes it possible to model not only invariances to the size and location of an object but also nonlinear warping in both dimensions. Furthermore, some extensions to structures representing typical geometric variations that are seen in many image recognition tasks have already been proposed, e.g., a structure for rotational variations [13], a structure with multiple horizontal and vertical Markov chains [14], and explicit state duration modeling [15]. By selecting an appropriate model structure reflecting the data generation process for a target task, human knowledge can effectively be utilized as prior information, and this makes it possible to construct models with a small amount of training data. However, SL-HMMs still have a limitation in their application to image recognition: observations are assumed to be generated independently from corresponding HMM states. It is insufficient to represent variations in images, e.g., lighting conditions and object deformation. To overcome this limitation, hidden Markov eigen-image models (HMEMs) have been proposed [16]. The basic idea of the HMEMs is that eigen-images [17, 18] are generated from an SL-HMM. In the HMEM, the eigen-images are represented by probabilistic hidden variable models, such as factor analysis (FA) [19–21] or probabilistic principal component analysis (PPCA) [22]. Therefore, HMEMs have the good properties of both SL-HMMs and FA/PPCA: size and location invariant image recognition and a linear feature extraction based on statistical analysis.

In some image recognition tasks, only a small amount of training data is available, therefore efforts to achieve a high generalization ability are required. However, the training of HMEMs easily falls into the over-fitting problem because HMEMs have a complex model structure. Also, the maximum likelihood (ML) criterion has typically been used in training HMEMs [16]. Since the ML criterion produces a point estimate of model parameters, the estimation accuracy may be degraded, especially when there is insufficient training data. In this study, I focus on estimating HMEMs with a high generalization ability by using the Bayesian criterion. The Bayesian criterion assumes that model parameters are random variables, and a high generalization ability can be obtained by marginalizing all model parameters in estimating predictive distributions. Moreover, the Bayesian criterion can utilize prior distributions representing useful prior information on model parameters. However, the Bayesian criterion requires complicated integral and expectation computations to obtain posterior distributions when models have hidden variables. To overcome this problem, the variational Bayesian (VB) method [23] has been proposed as an approximation method. Additionally, to alleviate the local maximum problem dependent on the initial parameters, I apply the deterministic annealing expectation maximization (DAEM) algorithm [24, 25] to the training of HMEMs using the VB method.

The rest of the paper is organized as follows. In Chapter 2, I explain basic theories of HMMs. Chapter 3 shows constructing TTS systems for languages with unknown pronunciations. Chapter 4 presents image recognition method based on HMEMs using a Bayesian framework. Concluding remarks and future plans are presented in the final chapter.

Chapter 2

Hidden Markov models

2.1 Definition of hidden Markov models

An hidden Markov model (HMM) is a finite state machine which generates a sequence of discrete time observations [26–28]. At each frame it changes states according to its state transition probability distributions, and then generates an observation $\boldsymbol{o}_t$ at frame t, according to its output probability distribution of the current state. Therefore, the HMMs are a doubly stochastic random process model.

The joint likelihood of observations $\boldsymbol{o} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_T\}$ and hidden variables $\boldsymbol{z} = \{z_1, z_2, \ldots, z_T\}$ can be written as:

$$P(\boldsymbol{o}, \boldsymbol{z} \,|\, \boldsymbol{\Lambda}) = P(\boldsymbol{z} \,|\, \boldsymbol{\Lambda}) P(\boldsymbol{o} \,|\, \boldsymbol{z}, \boldsymbol{\Lambda}) \tag{2.1}$$

$$= P(z_1 \,|\, \boldsymbol{\Lambda}) \prod_{t=1}^{T} P(z_t \,|\, z_{t-1}, \boldsymbol{\Lambda}) \prod_{t=1}^{T} P(\boldsymbol{o} \,|\, z_t, \boldsymbol{\Lambda}), \tag{2.2}$$

where $\boldsymbol{\Lambda}$ is a set of model parameters. The model parameters of HMMs are summarized as follows:

$$\boldsymbol{\Lambda} = \{\boldsymbol{\pi}, \boldsymbol{a}, \boldsymbol{b}\}. \tag{2.3}$$

1) $\boldsymbol{\pi} = \{\pi_k \,|\, 1 \leq k \leq K\}$: an initial state probability distribution. Where K is maximum HMM state. The probability of state k at $t = 1$ is represented by:

$$\pi_k = P(z_1 = k \,|\, \boldsymbol{\Lambda}). \tag{2.4}$$

2) $\boldsymbol{a} = \{a_{k,\bar{k}} \,|\, 1 \leq k, \bar{k} \leq K\}$: a state transition probability matrix. The probability of moving from state k to state $\bar{k}$ is represented by:

$$a_{k,\bar{k}} = P(z_t = \bar{k} \,|\, z_{t-1} = k, \boldsymbol{\Lambda}). \tag{2.5}$$

3) $\boldsymbol{b} = \{b_k(\boldsymbol{o}_t) \,|\, 1 \leq k \leq K\}$: an output probability distribution. The probability of an observation $\boldsymbol{o}_t$ being generated from a state k is represented by $b_k(\boldsymbol{o}_t) = P(\boldsymbol{o}_t \,|\, \boldsymbol{z}_t = k, \boldsymbol{\Lambda})$. In continuous distribution HMMs, each output probability distribution is modeled by a gaussian mixture model [29] as follows:

$$b_k\left(\boldsymbol{o}_t\right) = \sum_{m=1}^{M} w_{k,m} \mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{k,m}, \Sigma_{k,m}\right), \tag{2.6}$$

where M is the number of Gaussian component and $w_{k,m}$, $\boldsymbol{\mu}_{k,m}$, and $\Sigma_{k,m}$ respectively denote the Gaussian component (mixture) weight, mean vector, and covariance matrix of the m-th mixture of the k-th state. Each Gaussian component is defined by

$$\begin{aligned}&\mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{k,m}, \boldsymbol{\Sigma}_{k,m}\right)\\ &\quad= \frac{1}{\sqrt{(2\pi)^D \left|\boldsymbol{\Sigma}_{k,m}\right|}} \exp\left[-\frac{1}{2}\left(\boldsymbol{o}_t - \boldsymbol{\mu}_{k,m}\right)^\top \boldsymbol{\Sigma}_{k,m}^{-1}\left(\boldsymbol{o}_t - \boldsymbol{\mu}_{k,m}\right)\right],\end{aligned} \tag{2.7}$$

where symbol $\top$ means transpose of vector or matrix and D is the dimensionality of an observation vector $\boldsymbol{o}_t$. For each state, mixture weight $\{w_{k,m}\}_{m=1}^{M}$ should satisfy that:

$$\sum_{m=1}^{M} w_{k,m} = 1, \quad 1 \leq k \leq K \tag{2.8}$$

$$w_{k,m} \geq 0, \quad \begin{matrix} 1 \leq k \leq K \\ 1 \leq m \leq M \end{matrix} \tag{2.9}$$

so that $\{b_k\left(\cdot\right)\}_{k=1}^{K}$ are properly normalized, i.e.:

$$\int_{\mathrm{R}^D} b_k\left(\boldsymbol{o}_t\right) \mathrm{d}\boldsymbol{o}_t = 1. \quad 1 \leq k \leq K \tag{2.10}$$

Figures 2.1 and 2.2 show the model structure and graphical model representation of HMMs. Figure 2.1 shows a 3-state left-to-right model, in which the state increases or stays the same state as frame increases. In Figure 2.2, circles represent random variables, clear means hidden, and shaded means observed. The left-to-right HMMs are generally used to model speech parameter sequences, since they can appropriately model signals.

The total output probability of the observation vector sequence from the HMM is calculated by marginalizing Eq. (2.2) over all possible state sequences,

$$P\left(\boldsymbol{o} \,|\, \boldsymbol{\Lambda}\right) = \sum_{\boldsymbol{z}} \left[P(z_1 \,|\, \boldsymbol{\Lambda}) \prod_{t=1}^{T} P(z_t \,|\, z_{t-1}, \boldsymbol{\Lambda}) \prod_{t=1}^{T} P(\boldsymbol{o} \,|\, z_t, \boldsymbol{\Lambda})\right]. \tag{2.11}$$

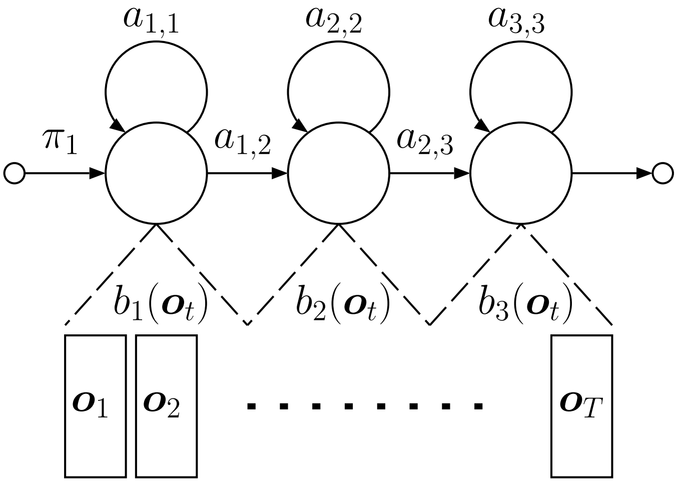


Figure 2.1: Model structure of HMMs.

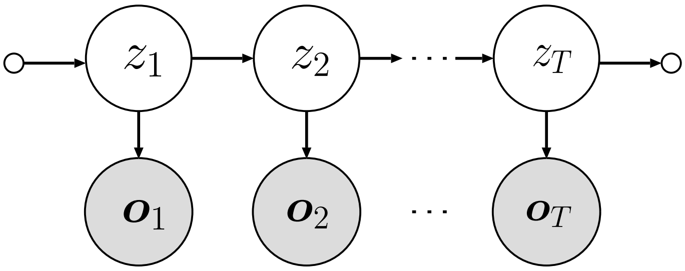


Figure 2.2: Graphical model representation of HMMs.

The order of $2T \cdot K^T$ calculation is required, since at every $t = 1, 2, \ldots, T$ there are K possible states that can be reached (i.e., there are K^T possible state sequences). This calculation is computationally infeasible, even for small values of K and T; e.g., for $K = 5, T = 100$, 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. (2.11) using forward and backward procedures.

2.2 Forward-backward algorithm

The forward-backward algorithm is generally used to calculate $P\left(\boldsymbol{o} \,|\, \boldsymbol{\Lambda}\right)$, which is the probability of the observation sequence $\boldsymbol{o}$ given the model $\boldsymbol{\Lambda}$. If I directly calculate $P\left(\boldsymbol{o} \,|\, \boldsymbol{\Lambda}\right)$, it requires on the order of $2T \cdot K^T$ calculation. The detail of the forward-backward algorithm is described in the following part.

The probability of a partial observation vector sequence from frame 1 to t and the k-th state at frame t, given the HMM $\mathbf{\Lambda}$ is defined as:

$$\alpha_t(k) = P\left(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t, z_t = k \,|\, \mathbf{\Lambda}\right). \tag{2.12}$$

$\alpha_t\left(k\right)$ is calculated recursively as follows:

1. Initialization

$$\alpha_1(k) = \pi_k b_k\left(\boldsymbol{o}_1\right), \quad 1 \leq k \leq K \tag{2.13}$$

2. Recursion

$$\alpha_t(\bar{k}) = \sum_{k=1}^{K} \alpha_{t-1}(k) a_{k,\bar{k}} b_k\left(\boldsymbol{o}_t\right), \quad \begin{array}{l} 1 \leq \bar{k} \leq K \\ 2 \leq t \leq T \end{array} \tag{2.14}$$

3. Termination

$$P\left(\boldsymbol{o} \,|\, \mathbf{\Lambda}\right) = \sum_{k=1}^{K} \alpha_T(k). \tag{2.15}$$

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

$$\beta_t(k) = P\left(\boldsymbol{o}_{t+1}, \boldsymbol{o}_{t+2}, \ldots, \boldsymbol{o}_T \,|\, z_t = k, \mathbf{\Lambda}\right), \tag{2.16}$$

that is, the probability of a partial vector observation sequence from frame t to T, given the k-th state at frame t and the HMM $\mathbf{\Lambda}$. The backward variables can also be calculated in a recursive manner as follows:

1. Initialization

$$\beta_T(k) = 1, \quad 1 \leq k \leq K \tag{2.17}$$

2. Recursion

$$\beta_t(k) = \sum_{\bar{k}=1}^{K} a_{k,\bar{k}} b_k\left(\boldsymbol{o}_{t+1}\right) \beta_{t+1}(\bar{k}), \quad \begin{array}{l} 1 \leq k \leq K \\ 1 \leq t \leq T-1 \end{array} \tag{2.18}$$

3. Termination

$$P\left(\boldsymbol{o} \,|\, \mathbf{\Lambda}\right) = \sum_{k=1}^{K} \beta_1(k). \tag{2.19}$$

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

$$P\left(\boldsymbol{o} \,|\, \mathbf{\Lambda}\right) = \sum_{\bar{k}=1}^{K} \alpha_t(\bar{k}) \beta_t(\bar{k}). \quad 1 \leq t \leq T \tag{2.20}$$

In the case of the forward algorithm, at frame $t = 1$, We need to calculate values of $\alpha_1(k)$, $1 \leq k \leq K$. At frames $2 \leq t \leq T$, We need only calculate values of $\alpha_t(\bar{k})$, $1 \leq \bar{k} \leq K$, where each calculation involves only the K previous values of $\alpha_{t-1}(k)$ because each of the K grid points can be reached from only the K grid points at the previous frame slot. As a result, the forward-backward algorithm can reduce order of probability calculation.

2.3 Viterbi algorithm

The single optimal state sequence $\hat{\boldsymbol{z}} = \{\hat{z_1}, \hat{z_2}, \ldots, \hat{z_T}\}$ 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 [30], the optimal state sequence $\hat{\boldsymbol{z}}$ can be obtained. Let $\delta_t\left(k\right)$ be the likelihood of the most likely state sequence ending in the k-th state at frame t

$$\delta_t(k) = \max_{z_1,\ldots,z_{t-1}} P\left(\boldsymbol{o}_1, \boldsymbol{o}_2, \ldots, \boldsymbol{o}_t, z_1, \ldots, z_{t-1}, z_t = k \,|\, \boldsymbol{\Lambda}\right), \tag{2.21}$$

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

1. Initialization

$$\delta_1\left(k\right) = \pi_k b_k\left(\boldsymbol{o}_1\right), \qquad 1 \leq k \leq K \tag{2.22}$$

$$\psi_1\left(k\right) = 0, \qquad 1 \leq k \leq K \tag{2.23}$$

2. Recursion

$$\delta_t\left(\bar{k}\right) = \max_k \left[\delta_{t-1}\left(k\right) a_{k,\bar{k}} b_k\left(\boldsymbol{o}_t\right)\right], \qquad \begin{matrix} 1 \leq k \leq K \\ 2 \leq t \leq T \end{matrix} \tag{2.24}$$

$$\psi_t\left(\bar{k}\right) = \arg\max_k \left[\delta_{t-1}\left(k\right) a_{k,\bar{k}} b_k\left(\boldsymbol{o}_t\right)\right], \qquad \begin{matrix} 1 \leq k \leq K \\ 2 \leq t \leq T \end{matrix} \tag{2.25}$$

3. Termination

$$\delta_T\left(K\right) = \max_k \left[\delta_T\left(k\right)\right], \tag{2.26}$$

$$\hat{z}_T = \arg\max_k \left[\delta_T\left(k\right)\right], \tag{2.27}$$

4. Back tracking

$$\hat{z}_t = \psi_{t+1}\left(\hat{z}_{t+1}\right). \quad 1 \le t \le T-1 \tag{2.28}$$

It should be noted that the Viterbi algorithm is similar to the forward calculation of Eqs. (2.13)–(2.15). The major difference is the maximization in Eq. (2.24) over previous states, which is used in place of the summation in Eq. (2.14). It also should be clear that a trellis structure efficiently implements the computation of the Viterbi procedure.

2.4 Expectation-maximization algorithm

There is no known method to analytically obtain the model parameter set based on the maximum likelihood (ML) criterion to obtain $\boldsymbol{\Lambda}$ which maximizes its likelihood $P\left(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}\right)$ 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 $\boldsymbol{\Lambda}$ which globally maximizes $P\left(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}\right)$. However, the model parameter set $\boldsymbol{\Lambda}$ locally maximizes $P\left(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}\right)$ can be obtained using an iterative procedure such as the expectation-maximization (EM) algorithm [31], and the obtained parameter set will be appropriately estimated if a good initial estimate is provided.

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

2.4.1 EM algorithm for HMMs

Since HMMs have hidden variables $\boldsymbol{z}$, it is difficult to obtain an analytic solution to likelihood $P\left(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}\right)$. The parameters of HMMs can be estimated via the EM algorithm [31], which is an iterative procedure. This procedure maximizes the expectation of the complete-data log-likelihood so-called $\mathcal{Q}$-function:

$$\mathcal{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Lambda}^{(\mathrm{old})}) = \sum_{\boldsymbol{z}} P(\boldsymbol{z}\,|\,\boldsymbol{o}, \boldsymbol{\Lambda}^{(\mathrm{old})}) \ln P(\boldsymbol{o}, \boldsymbol{z}\,|\,\boldsymbol{\Lambda}), \tag{2.29}$$

where $\boldsymbol{\Lambda}^{(\mathrm{old})}$ denotes the current parameters. Each Gaussian component is decomposed into a sub-state, and $\boldsymbol{z}$ is redefined as a sub-state sequence,

$$\boldsymbol{z} = \left\{\left(z_1, s_1\right), \left(z_2, s_2\right), \ldots, \left(z_T, s_T\right)\right\}, \tag{2.30}$$

where (z_t, s_t) represents being in the s_t-th sub-state of the z_t-th state at frame t.

The likelihood of the training data is guaranteed to increase by increasing the value of the $\mathcal{Q}$-function. The EM algorithm starts with some initial model parameters $\mathbf{\Lambda}^{(\mathrm{old})}$ and iterates between the following two steps.

$$\begin{aligned}\text{(E-step):}\quad & \text{compute } \mathcal{Q}(\mathbf{\Lambda}, \mathbf{\Lambda}^{(\mathrm{old})})\\ \text{(M-step):}\quad & \mathbf{\Lambda}^{(\mathrm{new})} = \arg\max_{\mathbf{\Lambda}} \mathcal{Q}(\mathbf{\Lambda}, \mathbf{\Lambda}^{(\mathrm{old})})\end{aligned}$$

The E-step computes the posterior probabilities of the hidden variables $P(\boldsymbol{z} \,|\, \boldsymbol{o}, \mathbf{\Lambda}^{(\mathrm{old})})$ while keeping model parameters $\mathbf{\Lambda}^{(\mathrm{old})}$ fixed to current values. Then, the $\mathcal{Q}$-function is computed by using $P(\boldsymbol{z} \,|\, \boldsymbol{o}, \mathbf{\Lambda}^{(\mathrm{old})})$. The M-step estimates the re-estimation parameters $\mathbf{\Lambda}^{(\mathrm{new})}$ by maximizing the $\mathcal{Q}$-function. These steps are iterated until convergence of the log-likelihood by replacing $\mathbf{\Lambda}^{(\mathrm{old})} \leftarrow \mathbf{\Lambda}^{(\mathrm{new})}$. By maximizing the $\mathcal{Q}$-function with respect to model parameter $\mathbf{\Lambda}$, the re-estimation parameters $\mathbf{\Lambda}^{(\mathrm{new})}$ in the M-step can be easily derived. By contrast, the calculation of the posterior probabilities $P(\boldsymbol{z} \,|\, \boldsymbol{o}, \mathbf{\Lambda}^{(\mathrm{old})})$ in the E-step is computationally intractable due to the combination of hidden variables.

2.4.2 Update model parameters

According to Eqs. (2.2) and (2.6), the joint likelihood of observations and hidden variables $\ln P(\boldsymbol{o}, \boldsymbol{z} \,|\, \mathbf{\Lambda})$ can be written as:

$$\ln P(\boldsymbol{o}, \boldsymbol{z} \,|\, \mathbf{\Lambda}) = \ln P(\boldsymbol{z} \,|\, \mathbf{\Lambda}) + \ln P(\boldsymbol{o} \,|\, \boldsymbol{z}, \mathbf{\Lambda}), \tag{2.31}$$

$$\ln P(\boldsymbol{z} \,|\, \mathbf{\Lambda}) = \ln \pi_{z_1} + \sum_{t=2}^{T} \ln a_{z_{t-1}, z_t}, \tag{2.32}$$

$$\ln P(\boldsymbol{o} \,|\, \boldsymbol{z}, \mathbf{\Lambda}) = \sum_{t=1}^{T} \ln w_{z_t, s_t} + \sum_{t=1}^{T} \ln \mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{z_t, s_t}, \Sigma_{z_t, s_t}\right). \tag{2.33}$$

Therefore, $\mathcal{Q}$-function (Eq. (2.29)) can be rewritten as:

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\Lambda}, \boldsymbol{\Lambda}^{(\mathrm{old})}) = &\sum_{k=1}^{K} P\left(\boldsymbol{o}, z_1 = k \,|\, \boldsymbol{\Lambda}\right) \ln \pi_k \\ &+ \sum_{k=1}^{K} \sum_{\bar{k}=1}^{K} \sum_{t=1}^{T-1} P\left(\boldsymbol{o}, z_{t-1} = k, z_t = \bar{k}\right) \ln a_{k,\bar{k}} \\ &+ \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{t=1}^{T} P\left(\boldsymbol{o}, z_t = k, s_t = m \,|\, \boldsymbol{\Lambda}\right) \ln w_{k,m} \\ &+ \sum_{k=1}^{K} \sum_{m=1}^{M} \sum_{t=1}^{T} P\left(\boldsymbol{o}, z_t = k, s_t = m \,|\, \boldsymbol{\Lambda}\right) \ln \mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{k,m}, \boldsymbol{\Sigma}_{k,m}\right).\end{aligned} \tag{2.34}$$

The set of HMM model parameters $\boldsymbol{\Lambda}$ which maximizes the above equation subject to the stochastic constraints

$$\sum_{k=1}^{K} \pi_i = 1, \tag{2.35}$$

$$\sum_{\bar{k}=1}^{K} a_{k,\bar{k}} = 1, \quad 1 \leq k \leq K \tag{2.36}$$

$$\sum_{m=1}^{M} w_{k,m} = 1, \quad 1 \leq k \leq K \tag{2.37}$$

$$w_{k,m} \geq 0, \quad \begin{matrix} 1 \leq k \leq K \\ 1 \leq m \leq M \end{matrix} \tag{2.38}$$

can be derived by using Lagrange multipliers as follows [32]:

$$\pi_k = \gamma_1(k), \qquad 1 \le k \le K \tag{2.39}$$

$$a_{k,\bar{k}} = \frac{\displaystyle\sum_{t=2}^{T} \xi_t(k,\bar{k})}{\displaystyle\sum_{t=2}^{T} \gamma_t(k)}, \qquad \begin{array}{c} 1 \le k \le K \\ 1 \le \bar{k} \le K \end{array} \tag{2.40}$$

$$w_{k,m} = \frac{\displaystyle\sum_{t=1}^{T} \gamma_t(k,m)}{\displaystyle\sum_{t=1}^{T} \gamma_t(k)}, \qquad \begin{array}{c} 1 \le k \le K \\ 1 \le m \le M \end{array} \tag{2.41}$$

$$\boldsymbol{\mu}_{k,m} = \frac{\displaystyle\sum_{t=1}^{T} \gamma_t(k,m)\boldsymbol{o}_t}{\displaystyle\sum_{t=1}^{T} \gamma_t(k,m)}, \qquad \begin{array}{c} 1 \le k \le K \\ 1 \le m \le M \end{array} \tag{2.42}$$

$$\boldsymbol{\Sigma}_{k,m} = \frac{\displaystyle\sum_{t=1}^{T} \gamma_t(k,m) \left(\boldsymbol{o}_t - \boldsymbol{\mu}_{k,m}\right) \left(\boldsymbol{o}_t - \boldsymbol{\mu}_{k,m}\right)^{\top}}{\displaystyle\sum_{t=1}^{T} \gamma_t(k,m)}, \quad \begin{array}{c} 1 \le k \le K \\ 1 \le m \le M \end{array} \tag{2.43}$$

where $\gamma_t(k)$ denotes the probability of being in the k-th state at frame t, $\gamma_t(k,m)$ denotes the probability of being in the m-th sub-state of the k-th state at frame t, and $\xi_t\left(k,\bar{k}\right)$ denotes the probability of being in the k-th state at frame $t-1$ and $\bar{k}$-th state at frame t

that is

$$\begin{aligned}
\gamma_t\left(k\right) &= P\left(\boldsymbol{o}, z_t = k \,|\, \boldsymbol{\Lambda}\right) \\
&= \frac{\alpha_t(k)\beta(k)}{\sum\limits_{k'=1}^{K}\alpha_t(k')\beta_t(k')}, && \begin{matrix} 1 \leq k \leq K \\ 1 \leq t \leq T \end{matrix} \qquad (2.44)\\
\gamma_t\left(k, m\right) &= P\left(\boldsymbol{o}, z_t = k, s_t = m \,|\, \boldsymbol{\Lambda}\right) \\
&= \frac{\alpha_t(k)\beta(k)}{\sum\limits_{k'=1}^{K}\alpha_t(k')\beta_t(k')} \frac{w_{k,m}\mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{k,m}, \boldsymbol{\Sigma}_{k,m}\right)}{\sum\limits_{m'=1}^{M} w_{k,m'}\mathcal{N}\left(\boldsymbol{o}_t \,|\, \boldsymbol{\mu}_{k,m'}, \boldsymbol{\Sigma}_{k,m'}\right)}, && \begin{matrix} 1 \leq k \leq K \\ 1 \leq m \leq M \\ 1 \leq t \leq T \end{matrix} \qquad (2.45)\\
\xi_t(k, \bar{k}) &= P\left(\boldsymbol{o}, z_{t-1} = k, z_t = \bar{k} \,|\, \boldsymbol{\Lambda}\right) \\
&= \frac{\alpha_t(k) a_{k,\bar{k}} b_{\bar{k}}\left(\boldsymbol{o}_{t+1}\right)\beta_{t+1}(\bar{k})}{\sum\limits_{k'=1}^{K}\sum\limits_{\bar{k}'=1}^{K}\alpha_t(k') a_{k',\bar{k}'} b_{\bar{k}'}\left(\boldsymbol{o}_{t+1}\right)\beta_{t+1}(\bar{k}')}. && \begin{matrix} 1 \leq k, \bar{k} \leq K \\ 1 \leq t \leq T \end{matrix} \qquad (2.46)
\end{aligned}$$

Chapter 3

Constructing text-to-speech systems for languages with unknown pronunciations

3.1 Background

A number of studies on text-to-speech (TTS) systems have been conducted. Consequently, the quality of synthetic speech has improved, and TTS systems are now used in various applications, such as in-car navigation, spoken dialogue, and speech translation systems. Accordingly, the demand for TTS systems offering high-quality synthetic speech, various speaking styles, and various languages is increasing. There are thousands of active written languages in the world [1]. Construction of a TTS system for a new language leads to increased use of applications. TTS systems for low-resource languages are in great demand because speech translation systems are very useful applications for low-resource languages. However, conventional methods of constructing corpus-based TTS systems for a new language not only require preparation of training corpus but also require language-specific knowledge. Especially, to marshal language-specific knowledge about pronunciation for each new language requires high cost. Therefore, a goal of the speech synthesis research is to establish a language-independent framework that can be used to construct TTS systems for any written language.

TTS systems can be examined as a text-to-speech mapping problem. Phoneme, the simplest abstract class of speech sounds, is a widely used intermediate representation for mapping. Thus, TTS systems have two main components: text analysis (text-to-phoneme) and speech waveform generation (phoneme-to-speech). In the text analysis part, a phoneme of an input text is estimated by using a lexicon which contains phonetic information. Additionally, some phonetic contextual factors, e.g., accents and parts-of-speech, are also estimated. These phoneme and phonetic contextual factors are used linguistic features. Since this part is highly dependent on the target language, it is costly to construct a TTS system for someone not familiar with the target language. In the speech waveform generation part, a speech waveform is generated from the linguistic features estimated by the text analysis part. Corpus-based speech synthesis approaches such as unit-selection [33] and statistical parametric speech synthesis (SPSS) have been proposed for the speech waveform generation part. SPSS, e.g., hidden Markov model (HMM)- and deep neural network (DNN)-based speech synthesis [34, 35], has been actively researched and the quality of synthetic speech has greatly improved. An SPSS system has several advantages: 1) within its statistical training framework, it can train the statistical properties of speakers, speaking styles, emotions, etc. from a training corpus; 2) many techniques that were developed for HMM/DNN-based speech recognition can be applied to speech synthesis; and 3) multiple languages can easily be supported because the language-dependent element is the only set of linguistic features to be used.

To construct a TTS system for a new language, it is necessary to marshal language-dependent elements, e.g., to define a phoneset and linguistic features, such as accents and parts-of-speech, for each language. However, doing so requires language-specific knowledge. Therefore, a low language-dependency framework is needed in order to construct TTS systems for new languages. In this study, I focus on automatic construction of a TTS system without knowledge specific to the language with the unknown pronunciation. I construct a TTS system from a database consisting of the only speech data and Unicode [2] texts corresponding to speech data. The problem in this situation is that a phoneset, phonetic information corresponding to speech data, and a lexicon do not exist. To solve these phoneset and phonetic information problems, speech recognition is carried out by using the speech recognizer of a rich-resource proxy language. Pseudo phoneme sequences of the target language speech data are obtained from the speech recognition results. An SPSS-based speech synthesizer of the target language is then trained from speech data and pseudo phoneme sequence pairs. To solve the lexicon problem, I train a grapheme-to-phoneme converter based on joint-sequence models [3] from text and pseudo phoneme sequence pairs. The joint-sequence model is a N-gram model that models a joint-sequence in which grapheme and phoneme sequences are aligned. The model can estimate a phoneme sequence with the highest likelihood from a grapheme sequence. In addition, in order to improve quality of synthesized speech, I propose improvement of the speech recognizer and estimation of the phoneme sequence considering phoneme duration. With these processes, it becomes possible to construct a TTS system automatically without specific knowledge on the target language.

In another way to address language-dependency, several low language-dependency frameworks have been proposed [36–38]. Grapheme-based speech synthesis treat every single graphemes as separate phoneme [36, 37]. Methods of constructing a TTS system based on UniTran [37], a transliteration framework to convert Unicode text into a guessed phoneme [39], and vector space models (VSMs) [38] have also been proposed. Unlike these low language-dependency methods, the proposed method can utilize the obtained pseudo phonetic information by the speech recognizer of a proxy language. Therefore, not only grapheme information but also phonetic information can be utilized to construct TTS systems in the proposed method.

I applied the proposed method to Japanese. The results of objective and subjective experiments are discussed and the impacts of components are analyzed. Comparing the proposed and grapheme-based TTS system, a subjective preference test was conducted. Additionally, I challenged the construction of TTS systems for nine Indian languages (Assamese, Bengali, Gujarati, Hindi, Malayalam, Marathi, Rajasthani, Tamil, and Telugu) using the proposed method in the Blizzard Challenge 2014 and 2015 [40, 41]. The results of the Blizzard Challenge 2015 were shown that the proposed TTS system was more natural sounding than the baseline TTS system for many languages.

3.2 HMM-based speech synthesis

A text-to-speech (TTS) system generates intelligible, natural-sounding artificial speech for a given input text. One of the major approaches in the TTS system is statistical parametric speech synthesis (SPSS) [42]. SPSS is a means of "mapping" (i.e., representing a map) of speech waveforms from text on the basis of a statistical model. However, a statistical model for directly predicting a speech waveform from text is difficult to construct. Accordingly, mapping a speech waveform from text can be divided into two steps: text analysis and speech waveform generation parts. Good examples of architectures suitable for modeling time-series data are available for acoustic modeling, and efficient training algorithms have been developed. For those reasons, HMMs are widely utilized, and SPSS based on HMMs (called "HMM-based speech synthesis") have become widely used as a standard speech-synthesis technique [34].

Figure 3.1 shows overview of the HMM-based speech synthesis system [43]. HMM-based speech synthesis system consists of training and synthesis parts. In the training part, context-dependent label sequences are estimated from text. Additionally, spectrum and excitation parameters are extracted from speech waveforms. These parameters are modeled by context-dependent HMMs. In the synthesis part, a sentence HMM is constructed by concatenating the context-dependent HMMs from a given text to be synthesized. The

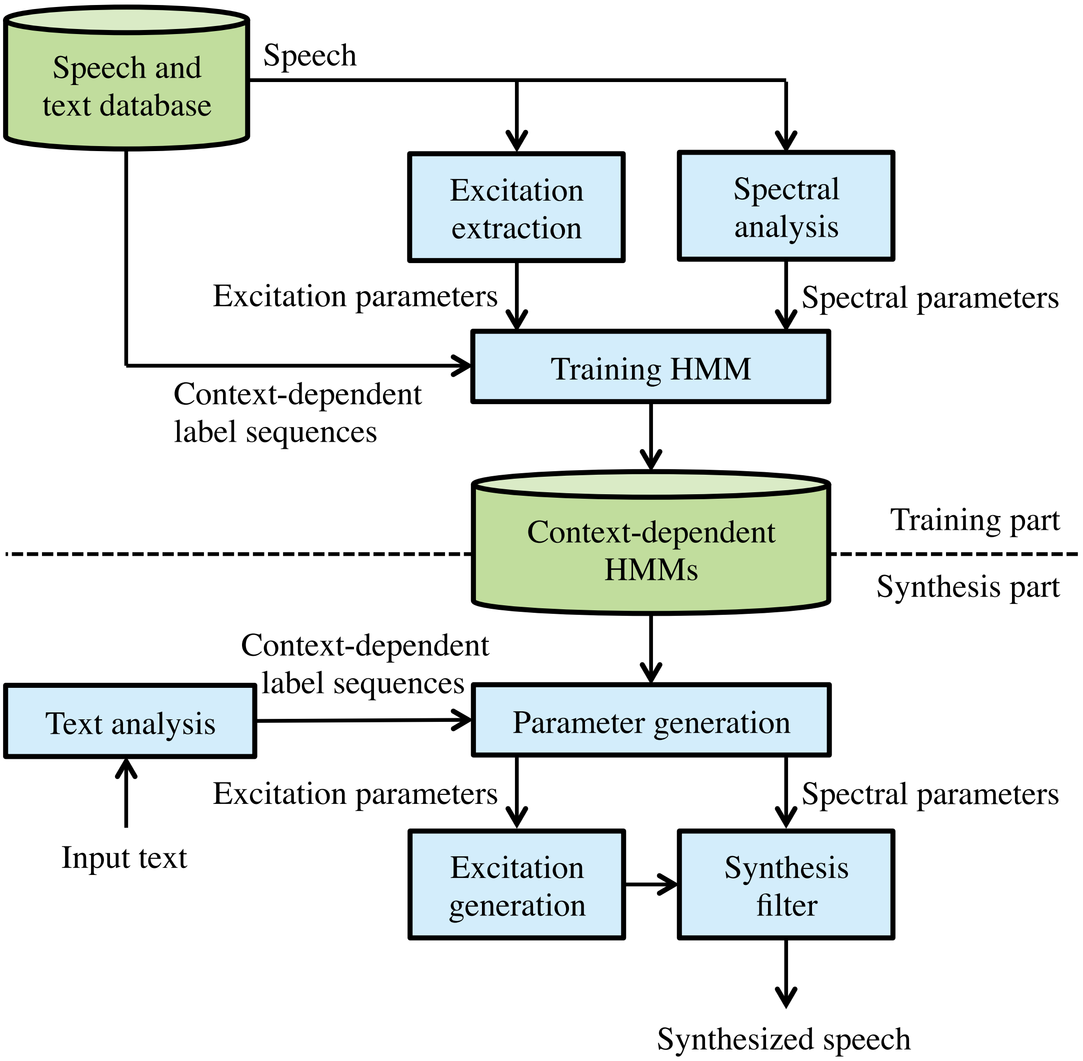


Figure 3.1: An overview of a typical HMM-based speech synthesis system.

sequences of spectrum and excitation parameters are generated from the sentence HMM using speech parameter generation algorithm [44–46]. Finally, speech waveform is synthesized from a synthesis filter module.

3.3 Text-to-speech system construction

3.3.1 Language-dependent text-to-speech system construction

Phonemes are widely used by text-to-speech (TTS) systems as intermediate representations for mapping a text to speech. The following language-dependent operations are needed in order to construct a TTS system of a new language.

- Define a phoneset and linguistic features.
- Construct a lexicon or grapheme-to-phoneme converter for the text analysis part.

Normally, the phoneset is defined based on the phonology of the target language. Linguistic features are designed based on pronunciation information obtained from texts of each language. Additionally, the lexicon for converting from graphemes to phonemes is manually created.

3.3.2 Constructing a text-to-speech system for a language with an unknown pronunciation

In this paper, I propose a method for constructing TTS systems that uses a target language database consisting of speech data and Unicode texts corresponding to speech data. In the case of an unknown-pronunciation language, it is difficult to define a phoneset and even more difficult to construct a hand-made lexicon, because they require manual operations used language-specific knowledge. Furthermore, it is hard to obtain a phoneme sequence corresponding to the speech data. To solve these problems, a speech recognizer of a rich-resource proxy language, e.g., English, for the target language can be used for automatic acquisition of phoneme sequences. The phoneset of the proxy language speech recognizer is then used as the phoneset of the target language. Although the phoneset is different from the appropriate phoneset of the target language, similar phonemes are assigned to speech data in this approach. To overcome the lexicon problem, a grapheme-to-phoneme converter based on a statistical model is used instead of a hand-made lexicon. In this way, entire TTS systems can be constructed within a statistical framework.

Figure 3.2 shows an overview of the proposed TTS system construction method for a language with an unknown pronunciation. This method consists of a speech recognizer (SR), word aligner (WA), grapheme-to-phoneme converter (G2P), and speech synthesizer (SS). The details of each component are described in the following sections.

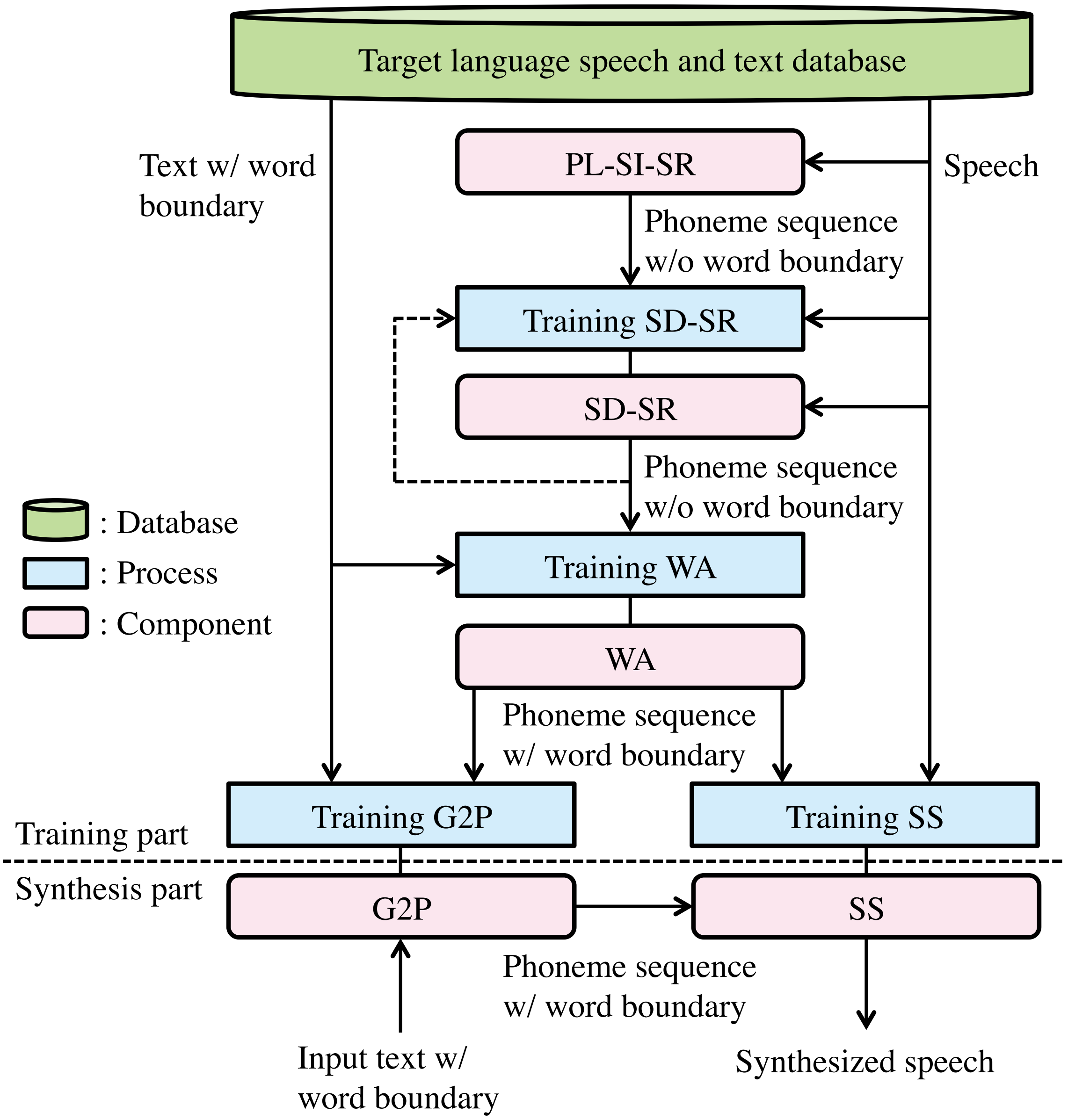

Target language speech and text database
Text w/ word boundary
Speech
PL-SI-SR
Phoneme sequence w/o word boundary
Training SD-SR
SD-SR
Phoneme sequence w/o word boundary
: Database
: Process
: Component
Training WA
WA
Phoneme sequence w/ word boundary
Training G2P
Training SS
Training part
Synthesis part
G2P
SS
Phoneme sequence w/ word boundary
Input text w/ word boundary
Synthesized speech

Figure 3.2: Overview of the proposed TTS system construction method for a language with an unknown pronunciation (PL-SI-SR: proxy language speaker-independent speech recognizer, SD-SR: speaker-dependent speech recognizer, WA: word aligner, G2P: grapheme-to-phoneme converter, SS: speech synthesizer).

Speech recognizer (SR)

In the case of SPSS, phoneme sequences corresponding to the speech data are necessary for acoustic modeling. To obtain phoneme sequences, speech recognition is carried out by using a proxy language speaker-independent SR (PL-SI-SR). For the target language recognition, a lexicon and language model are not used, and a phoneme network is designed so that each phoneme connects to every phoneme. In this way, the PL-SI-SR can work without being affected by a proxy-language-dependent phoneme sequence.

Since the accuracy of the phoneme sequences affects the latter components, i.e., the WA, G2P, and SS, it is important to estimate phoneme sequences accurately. To do so, a speaker-dependent SR (SD-SR) is constructed from initial phoneme sequences obtained by the PL-SI-SR. Furthermore, the phoneme sequence estimation and SD-SR training are iterated in order to adapt an acoustic model to training data. These iterations are acoustic-driven unsupervised training of speech units that uses the phoneset of the proxy language as an initial value.

Modeling of phoneme durations is important component for the SS. It is expected that phoneme sequences that are suitable for the SS can be obtained by taking account of phoneme duration. However, a hidden Markov model (HMM)-based SR has trouble accounting for phoneme duration because an HMM does not have explicit state duration information. Therefore, phoneme sequences are rescored using an alignment likelihood of a hidden semi-Markov model (HSMM) that has explicit state duration probability distributions. The phoneme sequence with the highest HSMM alignment likelihood in the N-best hypotheses of the HMM speech recognition result is selected as the pseudo phoneme sequence corresponding to the speech data.

Word aligner (WA)

Since many languages, e.g., English and Spanish, are written with spaces between words, a word-level G2P is suitable for the text analysis part. Furthermore, word boundary information is useful as linguistic features of the SS. However, a phoneme sequence obtained by the SR does not include word boundaries. Therefore, I construct a WA based on a joint-sequence model [3] for estimating word boundaries.

The optimal grapheme and phoneme pair alignment $\hat{\boldsymbol{w}}$ is estimated as follows:

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w} \in \boldsymbol{W}} P(\boldsymbol{w}). \tag{3.1}$$

Here, $\boldsymbol{w}$ is a alignment of grapheme and phoneme pairs and $\boldsymbol{W}$ denotes the set of alignments of all possibly different grapheme and phoneme pairs. The parameters of the joint-sequence models are estimated by using the expectation-maximization (EM) algorithm. Pairs of texts with word boundaries and phoneme sequences obtained by the speech recognition are used for training. The WA is trained by providing a constraint condition such that a pause in the recognition results must be a word boundary. The Viterbi algorithm is used to align the grapheme and phoneme pairs. The word boundary of the phoneme sequence are estimated by the phoneme corresponding to the grapheme with the word boundary.

Grapheme-to-phoneme converter (G2P)

To synthesize an arbitrary text, an input text needs to be converted into a phoneme sequence. However, in a language with an unknown pronunciation, it is difficult to construct a hand-made lexicon for converting input texts into phonemes. To overcome this problem, a G2P based on a joint-sequence model [3] is used instead of a hand-made lexicon. The G2P is trained from word-level pairs of text and phoneme sequences obtained by the SR and WA.

Insertion of appropriate pauses is important for natural synthesized speech. To estimate pauses by the G2P, word-level phoneme sequences of training data contain pauses in the speech recognition results. This makes it possible to estimate pauses when converting a phoneme sequence by the G2P.

Speech synthesizer (SS)

In the case of SPSS, context-dependent models are used to capture a variety of phonetic contextual factors. To generate naturally sounding synthesized speech, appropriate phonetic contextual factors (linguistic features) need to be defined. Here, I can use linguistic features of phoneme, syllable, word, phrase, and utterance. The details of these hierarchical linguistic features are as follows.

- Phoneme:
  - the current phoneme;
  - preceding and succeeding two phonemes;
  - the position of the current phoneme within the current syllable.
- Syllable:
  - the number of phonemes within preceding, current, and succeeding syllables;
  - the position of the current syllable within the current word and phrase;
  - the vowel identity within the current syllable.
- Word:

  - the number of syllables within preceding, current, and succeeding words;
  - the position of the current word within the current phrase.

- Phrase:
  - the number of syllables and words within preceding, current, and succeeding phrases;
  - the position of the current phrase within the utterance.

- Utterance:
  - the number of syllables, words, and phrases in the utterance.

A linguistic feature related to phoneme is obtained by the results of the SR. A syllable which is normally defined as $C^{*}VC^{*}$ is useful as linguistic features of the SS. Here, C is a consonant, V is a vowel, and C^{*} indicates there may be none or more consonants. The consonant or vowel of a phoneme is dependent on the phoneset of the language used in the PL-SI-SR. A linguistic feature related to word is obtained by the results of the WA. A pause in the speech recognition results is defined as a phrase boundary. The SS can be constructed using the same procedure as the standard one from speech data and linguistic features corresponding to speech data.

3.4 Experimental conditions

3.4.1 Target language database conditions

Objective and subjective experiments were conducted to evaluate the effectiveness of the proposed method. Since I can easily gather Japanese native subjects, listening tests for Japanese synthesized speech are desirable. Thus, Japanese was chosen as the target language. Of the 503 phonetically balanced sentences in the ATR Japanese speech database B-set [47] that were uttered by a male speaker MHT, 450 sentences were used for training and the remaining 53 sentences were used for testing.

Since there are a large number of graphemes in Japanese, e.g., hiragana, katakana, romaji, and kanji, a large amount of training data is needed to construct a G2P. Katakana, romaji and kanji can be represented by using hiragana in Japanese. Only hiragana was used as the graphemes in the experiments. Furthermore, assuming languages written with spaces between words, e.g., English and Spanish, a bunsetsu boundary which is a boundary of basic grammatical unit in Japanese was assumed as a word boundary in linguistic features. Table 3.1 shows an example of Japanese text for the experiments.

Table 3.1: Example of Japanese text in the experiment.

Original Japanese text	テレビゲームやパソコンでゲームをして遊ぶ
Japanese text for the experiment	てれびげーむや　ぱそこんで　げーむを　して　あそぶ

3.4.2 Speech recognizer conditions

An English SI-SR was used as the PL-SI-SR. The CMU pronunciation dictionary [48] and the WSJ0, WSJ1 [49], and TIMIT [50] databases were used to train the English SI-SR. The phoneset of English SI-SR has 40 phonemes. Speech signals were sampled at a rate of 16 kHz and windowed by a 25-ms Hamming window with a 10-ms shift. The acoustic feature vector consisted of 39 components comprised of 12-dimensional mel-frequency cepstral coefficients (MFCCs) including the 0th energy coefficient with the first- and second-order derivatives. A triphone three-state left-to-right Gaussian mixture model (GMM)-HMM without skip transitions was used as an acoustic model. The trained GMMs had 32 mixtures for pause and 16 mixtures for the other phonemes. The HTK [51] was used to construct the SR. The training procedures and model structures were the same as that of the HTK Wall Street Journal Training Recipe [52].

To consider phoneme duration, a five-state left-to-right monophone multi-stream multi-space probability distribution (MSD)-HSMMs [42,43,46,53] without skip transitions was trained from the TIMIT database. The other model structure and acoustic feature vector were the same as the SS.

3.4.3 Word aligner and grapheme-to-phoneme convert conditions

A joint-sequence model based WA was constructed from texts with word boundary and phoneme sequences without word boundary. The WA considered the context independent joint uni-gram.

A G2P based on the joint-sequence model was constructed from word-level pairs of text and phoneme sequence obtained by the SR and WA. As a result of a preliminary experiment, a joint eight-gram was used for the G2P structure. The G2P was trained by using the Sequitur G2P [54].

3.4.4 Speech synthesizer conditions

The speech signals were sampled at 16 kHz and windowed with a fundamental frequency (F_0)-adaptive Gaussian window with a 5-ms shift. The acoustic feature vectors were comprised of 183 dimensions: 39-dimension STRAIGHT [55] mel-cepstral coefficients including the 0th coefficient, log F_0, 19-dimension mel-cepstral analysis aperiodicity measures including the 0th coefficient, and their first- and second-order derivatives. A five-state left-to-right context-dependent multi-stream MSD-HSMMs [42, 43, 46, 53, 56] without skip transitions was used as the acoustic model. Each state output distribution was composed of a spectrum, F_0, and aperiodicity streams. The spectrum and aperiodicity streams were modeled using single multi-variate Gaussian distributions with diagonal covariance matrices. The F_0 stream was modeled using an MSD consisting of a Gaussian distribution for voiced frames and a discrete distribution for unvoiced frames. State durations were modeled using a 1-dimensional Gaussian distribution. A parameter generation algorithm considering the global variance (GV) was applied [57]. The HTS [58] was used for constructing the SS.

A syllable is normally defined as C^*VC^* in the phonology. However, it is difficult to construct a language-independent method for estimating syllables. Therefore, in this experiment, a syllable is defined as C^*V assuming a Japanese mora which is basically one hiragana grapheme.

3.5 Experimental results

3.5.1 Effect of speech recognizer

First, the effect of SR was experimentally evaluated. In the proposed method, speech recognition results affect components in the latter part. Therefore, the phoneme sequences obtained from the SR have a big impact on the quality of the synthetic speech.

To objectively evaluate the effect of rescoring using HSMM alignment likelihood, mel-cepstral distortions (MCDs) were calculated [59]. The PL-SI-SR (English SI-SR) was used to estimate pseudo phoneme sequences. Table 3.2 shows the results of MCDs in open data. **1-best** system did not apply HSMM-based rescoring process, i.e., speech recognition results with HMMs were used as the phoneme sequences of the training data. On the other hand, **50-best** system rescored the 50-best hypotheses, which were obtained from HMM-based speech recognition system, using the HSMM alignment likelihoods. From Table 3.2, since **50-best** system achieved lower average MCD than **1-best** sys-

Table 3.2: MCD for synthesized speech obtained various insertion penalty in **1-best** and **50-best**.

	MCD [dB]	
Insertion penalty	**1-best**	**50-best**
-25	6.27	6.26
-20	6.28	6.22
-15	6.33	6.20
-10	6.27	6.27
-5	6.36	6.22
0	6.32	6.37
Average	6.31	6.26

tem, the effectiveness of rescoring using HSMM alignment likelihood was confirmed. Consequently, **50-best** system was used to estimate phoneme sequences in the following experiments.

The effects of the phoneme insertion penalty and the number of iterations of training and recognition were investigated. A speech recognition score is calculated by using an acoustic model likelihood and a phoneme insertion penalty. The acoustic model likelihood tends to increase with a sequence of multiple short phonemes. The phoneme insertion penalty is a penalty parameter to control the number of phonemes included in speech recognition results. Figure 3.3 shows the average number of phonemes per sentence in each system. In Figure 3.3, **PL-SI-SR** means systems using phoneme sequences obtained by the English SI-SR, **SD-SR_$\boldsymbol{i}$** means systems using phoneme sequences obtained by the SD-SR ($\boldsymbol{i}$ denotes iteration count), and **IP_$\boldsymbol{p}$** means the phoneme insertion penalty ($\boldsymbol{p}$ denotes value of phoneme insertion penalty). It is confirmed that the number of phonemes was influenced by the phoneme insertion penalty. The average number of phonemes increased with each iteration $\boldsymbol{i}$. In the proposed method, since the acoustic model was adapted to training data by iterations of training and recognition, the acoustic model likelihood increased with each iteration. Therefore, the influence of the phoneme insertion penalty relatively decreased and the average number of phonemes increased.

To objectively evaluate the effect of the phoneme insertion penalty and the number of iterations of training and recognition, MCDs were calculated. Figure 3.4 shows the results of MCDs in closed and open data. It can actually be seen in Figure 3.4 that **SD-SR_$\boldsymbol{i}$** systems achieved significantly lower MCDs than the **PL-SI-SR** system. For the **SD-SR_$\boldsymbol{i}$** systems, MCD decreased as the number of iterations $\boldsymbol{i}$ increased. Despite the convergence of the MCDs in the closed data, the MCDs of the **SD-SR_6** systems became higher than

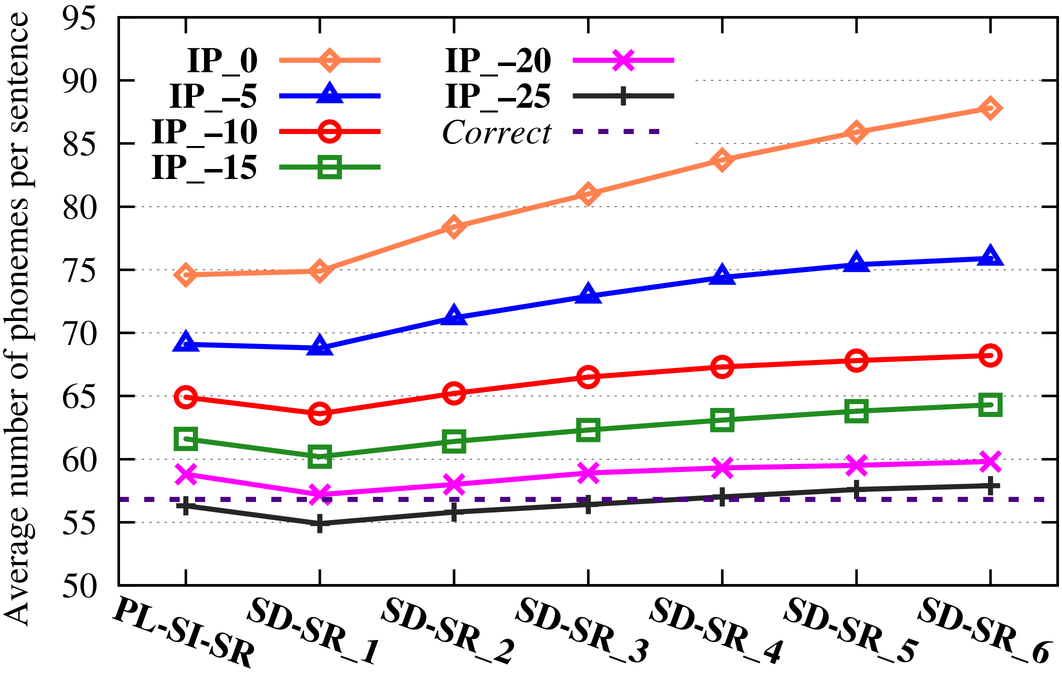
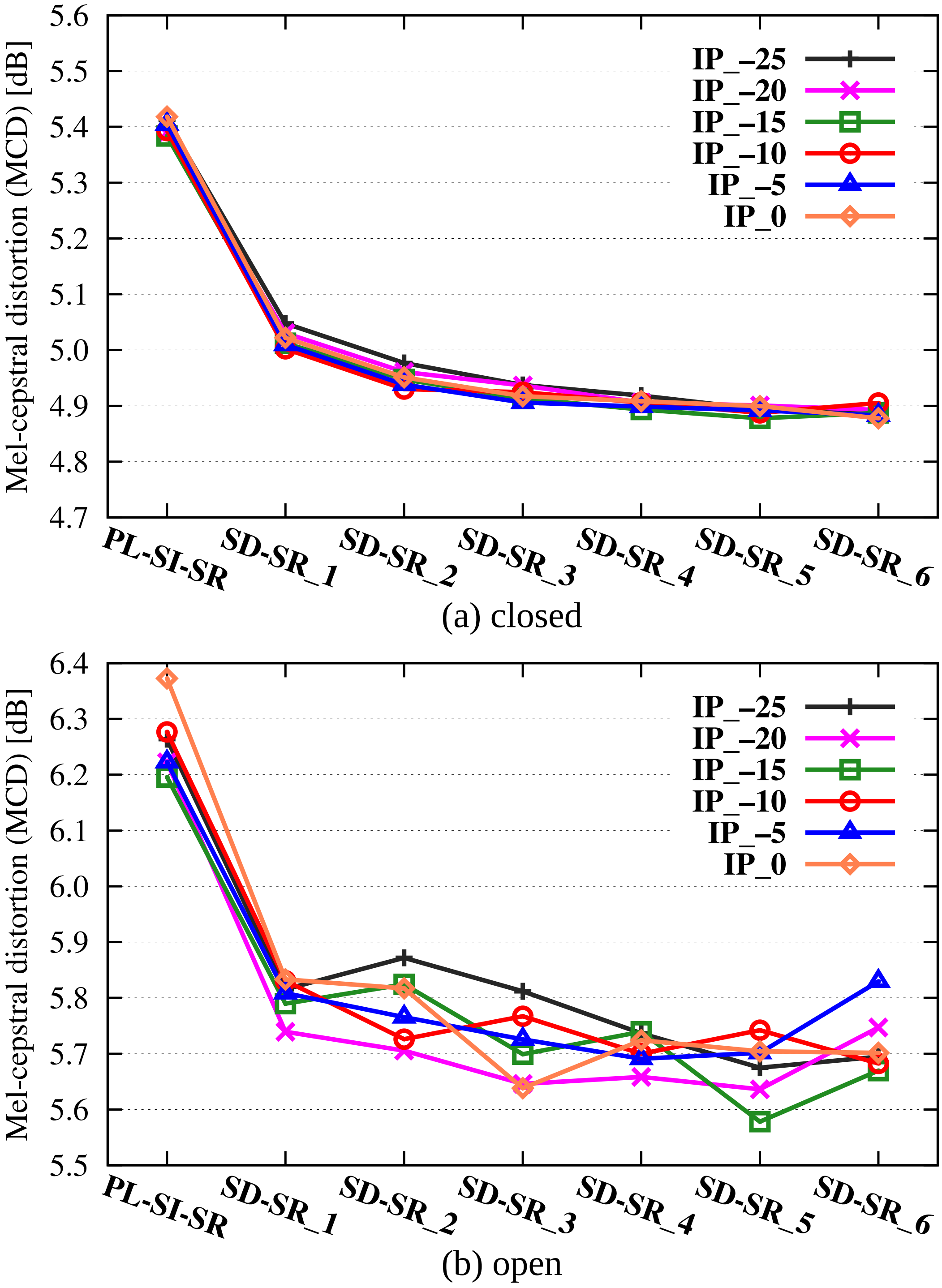


Figure 3.3: Average number of phonemes per sentence in the training data. *Correct* means correct average number of phonemes using Japanese phoneset.

those of the **SD-SR_5** systems in the open data. This is because the **SD-SR_6** systems had an overfitting problem.

A five-point mean opinion score (MOS) listening test with **SD-SR_5** having various insertion penalties was conducted in order to subjectively evaluate the naturalness of the synthesized speech. The subjects were ten Japanese students in our research group. All experiments were carried out using headphones in a soundproof room. For comparison, 20 sentences were chosen at random from the 53 test sentences. Speech samples were presented in random order for each test sentence. The scale of naturalness ran from 5 for “completely natural” to 1 for “completely unnatural” in the MOS test. The results of the MOS listening test are depicted in Figure 3.5. It can be seen from the figure that **IP_−15** performed best. From Figure 3.3, the number of phonemes in **IP_−15** was larger than the correct number of phonemes using the Japanese phoneset. These results suggest that the proposed system compensated for the differences in the phoneset by acoustic-driven short speech units. However, **IP_−10**, **IP_−5**, and **IP_0** did not obtain a higher MOS than the system **IP_−15**, though these systems included the large number of phonemes. Therefore, appropriate setting of phoneme insertion penalty is required to obtain high natural speech. Table 3.3 shows an example of phoneme sequences with word boundaries in training data obtained by **SD-SR_5** systems.



(a) closed

(b) open

Figure 3.4: MCD for synthesized speech obtained various iteration counts of the training and recognition.

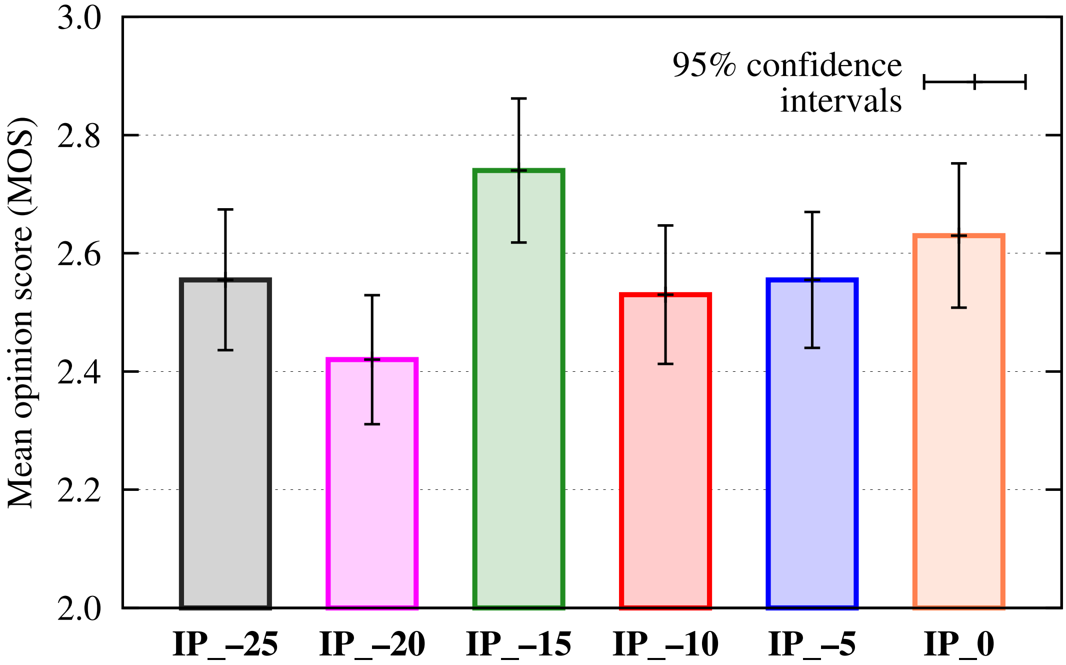


Figure 3.5: MOS of naturalness with 95 % confidence intervals for various insertion penalties in **SD-SR_5**.

Table 3.3: An example of phoneme sequences with word boundaries in training data (ぶんしょは　ねんねん　ふえていく). Where, | represent a word boundary.

System	Phoneme sequence with word boundaries
IP_−25	n b l iy s ih l ay | n eh n n ey n | f r ih p dh iy sp k uw
IP_−20	n b w iy zh ih l aa | n eh n n ey ng f r | ih p dh iy t k uw
IP_−15	n b uh ey s jh ih l aa | n eh n n ey n | f r ih p dh iy t k uw
IP_−10	n b uh ey s jh ih l ay | n eh n n ey ng d f r | ih p dh ey iy t k uw p
IP_−5	n b w oy iy s jh ih l aa | n eh n n ey ng | f r iy eh p dh iy iy t p g uw t
IP_0	n b uh ey ng z s jh ih l aa | n ey eh n n ey ng v f r | ey eh p dh iy iy t p g uw ih p

Table 3.4: Subjective preference scores.

Grapheme	**E-SI-SR**	Neutral	p-value
41.5 %	57.0 %	1.5 %	0.0325

It is confirmed that the pseudo phoneme sequence of the system with little influence of phoneme insertion penalty, such as **IP_0** and **IP_-5**, was composed of acoustic-driven short speech units. In addition, it can be seen that **IP_-20**, **IP_-10**, and **IP_0** contained errors in second word boundary in the example.

3.5.2 Comparing the proposed and grapheme-based systems

Grapheme-based speech synthesis system is often used as a baseline system for language-independent methods. Comparing the proposed and grapheme-based systems, a subjective preference listening test was conducted. The condition of preference listening test was the same as the MOS test. Table 3.4 shows the preference test result. **Grapheme** means a system which uses graphemes as speech unit instead of phonemes, and **E-SI-SR** means a system using the **50-best**, **SD-SR_5**, and **IP_-15** in Section 3.5.1 It can be seen from Table 3.4 that **E-SI-SR** was preferred to **Grapheme**. For this reason, the proposed method (**E-SI-SR**) may be useful for constructing a TTS system of a language with an unknown pronunciation without using language-specific knowledge. Since mappings from grapheme to phoneme are mostly unique in Japanese hiragana, **Grapheme** was able to synthesize speech with small pronunciation errors. In a language in which it is difficult to map from grapheme to phoneme, the proposed method is more expected to improve the performance compared to **Grapheme**.

3.5.3 Impact of components

The proposed method estimates all linguistic features in the training and synthesis parts. To analyze the impact of each component, systems using correct linguistic features were compared. Additionally, a Japanese SI-SR was constructed by using the JNAS database [60] for comparison with the English SI-SR. The phoneset of Japanese SI-SR has 35 phonemes. The acoustic feature vector and model structure were the same as the English ones. Table 3.5 summarizes the compared systems and the following is a description of the compared systems.

Table 3.5: Systems using correct linguistic features.

System	Phoneset	Training part		Synthesis part	Language	Construction
		Phoneme seq.	Word boundary	Phoneme seq.	of PL-SI-SR	component
Oracle	Japanese	Correct	Correct	Correct	-	SS
PhonemeWB	Japanese	Correct	Correct	Estimate	-	G2P, SS
Phoneme	Japanese	Correct	Estimate	Estimate	-	WA, G2P, SS
J-SI-SR	Japanese	Estimate	Estimate	Estimate	Japanese	SR, WA, G2P, SS
E-SI-SR	English	Estimate	Estimate	Estimate	English	SR, WA, G2P, SS

Table 3.6: MCD and RMSE for synthesized speech of systems using correct linguistic features.

System	MCD [dB]	RMSE [log Hz]
Oracle	5.01	0.140
PhonemeWB	5.35	0.189
Phoneme	5.35	0.193
J-SI-SR	5.49	0.196
E-SI-SR	5.58	0.198

- **Oracle**: system using correct linguistic features in the training and synthesis parts.
- **PhonemeWB**: system using correct linguistic features in the training part.
- **Phoneme**: system using correct phoneme sequences in the training part.
- **J-SI-SR**: system using a Japanese SI-SR, **50-best**, **SD-SR_5**, and **IP_-25**. The phoneme insertion penalty was set approximately to the correct number of phonemes.

To objectively evaluate the impact of components, MCD and root mean squared error (RMSE) of log F_0 were used. Table 3.6 lists the results of the objective evaluation. In terms of MCD, the systems closer to **Oracle** obtained a lower MCD. There was a large difference in MCD between **Oracle** and **PhonemeWB**. Phoneme error rate (PER) of the G2P in **PhonemeWB** can be calculated because it uses correct phoneme sequences and word boundaries in the training part. To evaluate pause insertion accuracy, PER excluding pauses was also calculated. The G2P in **PhonemeWB** obtained a PER of 3.40 % and a PER excluding pauses of 0.31 %. Most of phoneme estimation errors of the G2P in **PhonemeWB** were caused by pause insertion errors. This result suggests that pause insertion errors have strong impacts on MCD and improvement of the G2P, especially pause insertion, is necessary to improve MCD. Error rates of the WA can be also calculated in **Phoneme**. The number of error boundaries included in the training data was one and the word boundary error rate was 0.04 %. Therefore, the impact of the WA was not large comparing with the G2P in this experiment. From MCDs in Table 3.6, it can be seen that there was also a difference between **Phoneme** and **J-SI-SR**. This result indicates that there is a difference between the correct phoneme sequence and pseudo phoneme sequence. Accordingly, improving speech recognition accuracy is necessary. Comparing **J-SI-SR** with **E-SI-SR**, there was a relatively large gap of MCDs. Figure 3.6 shows MCD for synthesized speech obtained **E-SI-SR** and **J-SI-SR**. Although speaker adaptation was applied from **PL-SI-SR** to **SD-SR_1** in **J-SI-SR**, the improvement of MCD was small. On the

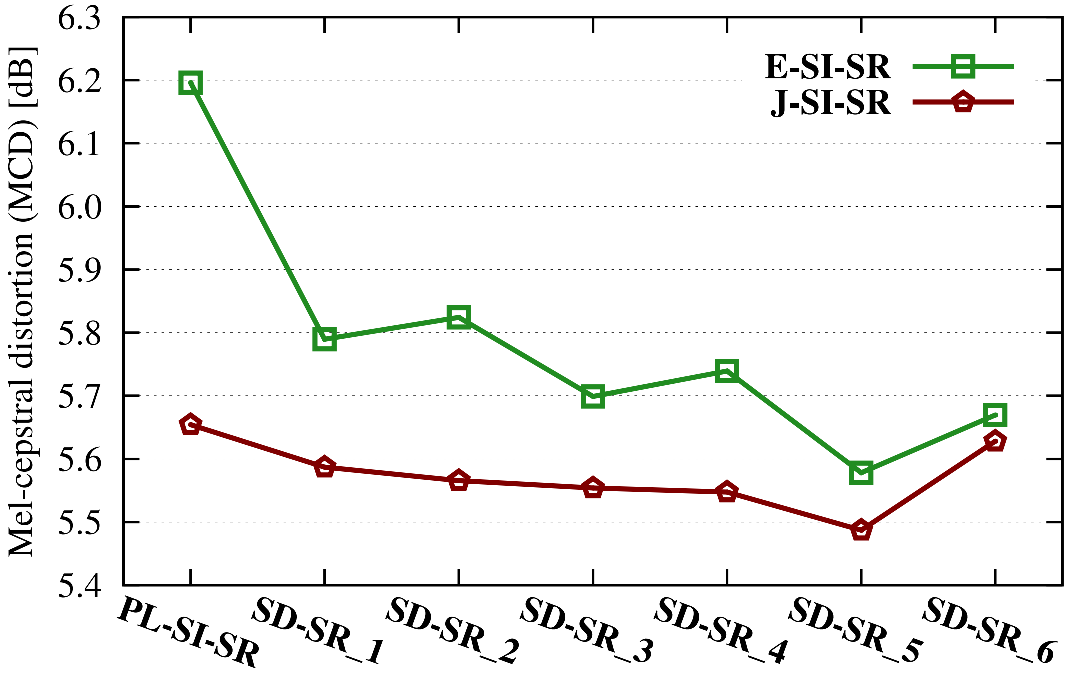


Figure 3.6: MCD for synthesized speech obtained **E-SI-SR** and **J-SI-SR**.

other hand, in **E-SI-SR**, speaker and language adaptation was applied from **PL-SI-SR** to **SD-SR_1**, and the MCD was significantly improved. This indicates that language adaptation is more effective than speaker adaptation in the proposed method. Additionally, from Table 3.6, RMSE showed the similar tendency as MCD.

To subjectively evaluate the impact of components, a five-point MOS listening test was conducted. Figure 3.7 shows the MOS of naturalness. As in the case of the objective evaluation in Table 3.6, the systems closer to **Oracle** obtained a higher MOS. There was a large difference in MOS between **Phoneme** and **J-SI-SR** and between **J-SI-SR** and **E-SI-SR**. These results indicate that speech recognition accuracy and phoneset of speech recognizer affect naturalness of synthetic speech in the proposed method.

Moreover, to evaluate intelligibility, intelligibility test was conducted. The subjects were asked to transcribe semantically unpredictable sentences (SUSs) by typing in the sentence they heard. 100 SUSs with each four words from the JEITA standard [61] were used for the evaluation. The subjects were ten Japanese students in our research group. Each subject typed 100 SUSs of a system chosen randomly. The average grapheme error rate (GER) was calculated from these transcripts. Table 3.7 lists the results of the intelligibility test in terms of GER. **Oracle**, **PhonemeWB**, and **Phoneme**, which used the phoneset based on the phonology, achieved low GER. Like the MOS evaluation in Figure 3.7, there

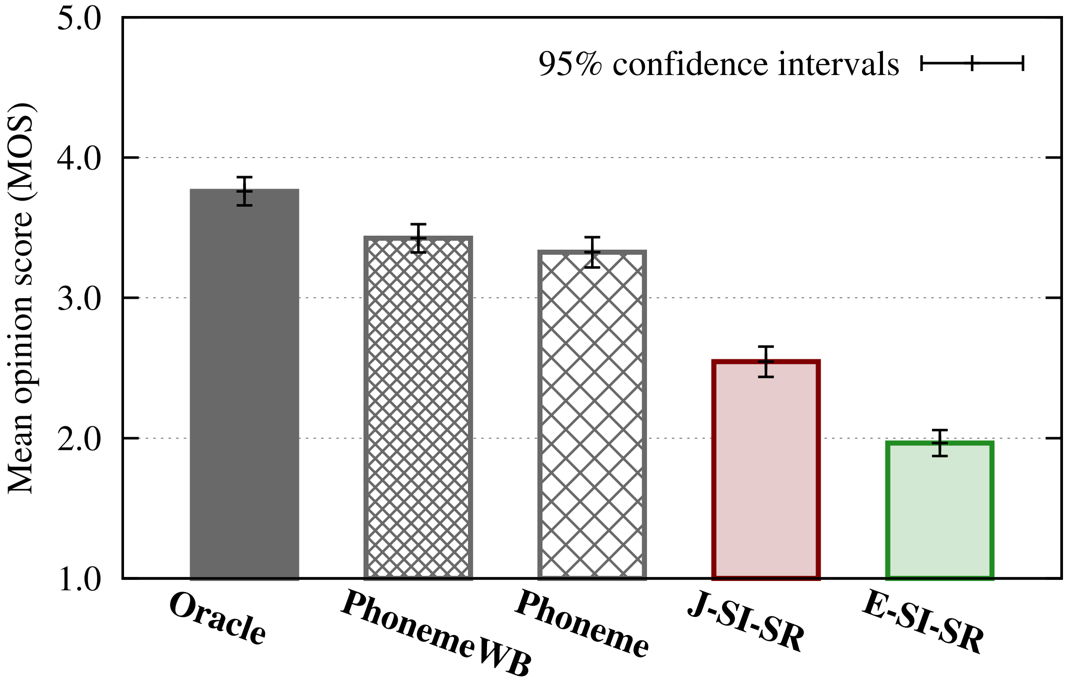


Figure 3.7: MOS of naturalness with 95% confidence intervals for systems using correct linguistic features.

Table 3.7: GER of systems using correct linguistic features.

System	GER [%]
Oracle	5.73
PhonemeWB	6.69
Phoneme	5.54
J-SI-SR	22.52
E-SI-SR	33.33

was a large difference in GER between **Phoneme** and **J-SI-SR** and between **J-SI-SR** and **E-SI-SR**. Ambiguous pronunciations had a bad influence on the GER. In **E-SI-SR**, several words were partially missing phonemes due to estimation errors in the G2P. The cause of these errors was the G2P training with training data including word boundary errors, such as word boundary errors in Table 3.3. In the case of **Phoneme** which used correct phoneme sequences, GER (5.54 %) and word boundary error rate (0.04 %) were low. Therefore, it is necessary to develop a noise-robust WA and improve the SR. It is considered that the low intelligibility influenced the low naturalness of **J-SI-SR** and **E-SI-SR**. In the future, I should investigate methods to improve intelligibility.

Table 3.8: Number of native paid listeners.

Language	Number of native paid listeners
Bengali	48
Hindi	69
Malayalam	72
Marathi	69
Tamil	70
Telugu	70

Table 3.9: MOS of naturalness and speaker similarity in the Blizzard Challenge 2015.

	MOS of naturalness		MOS of similarity	
Language	**Base**	**NITech**	**Base**	**NITech**
Bengali	2.2	2.5	2.5	3.1
Hindi	3.2	2.3	2.6	2.8
Malayalam	1.6	1.7	1.8	2.3
Marathi	2.7	2.2	2.3	2.5
Tamil	2.2	2.4	1.8	2.3
Telugu	1.9	2.1	2.1	3.1

3.6 Blizzard Challenge 2015 evaluation

The Blizzard Challenge was started in order to better understand and compare research techniques in constructing corpus-based speech synthesizers with the same data in 2005 [62]. The task of the Blizzard Challenge 2015 is constructing TTS systems for six Indian languages (Bengali, Hindi, Malayalam, Marathi, Tamil, and Telugu) [63]. These Indian languages have millions of speakers. However, these languages do not have a lot of resources for constructing a TTS system. The challenge is to construct TTS systems in each Indian language from the provided speech data sampled at 16 kHz and the corresponding Unicode text. About four or two hours of speech data in each of the six Indian languages are provided. To evaluate the synthesized speech, large-scale subjective evaluation tests were conducted by organizers of the Blizzard Challenge 2015. Table 3.8 summarizes the number of native paid listeners. I participated in the Blizzard Challenge 2015 [41] using the proposed method in this paper.

Table 3.9 shows results of five-point MOS tests in the read text task of the Blizzard Challenge 2015. In Table 3.9, **Base** means a baseline system used language-specific knowledge which was constructed by organizers using the FestVox [64] in the unit selection framework and **NITech** means my system. **NITech** systems were constructed without using language-specific knowledge based on Section 3.3.2 and system conditions were the same as Section 3.4.2, 3.4.3, and 3.4.4 Iteration count i and phoneme insertion penalty were adjusted for each language. Since Hindi has relatively rich-resource for constructing a TTS system in Indian languages, **Base** achieved higher MOS of naturalness than **NITech**. By contrast, **NITech** obtained higher MOSs of naturalness than **Base** in Bengali, Malayalam, Tamil, and Telugu. Furthermore, **NITech** achieved higher MOSs of speaker similarity than **Base** in all languages. For this reason, the proposed method is useful for constructing a TTS system of low-resource languages.

3.7 Summary

This paper has presented automatic construction of a text-to-speech (TTS) system from a target language database consisting of only speech data and corresponding Unicode texts. A grapheme-to-phoneme converter and speech synthesizer were constructed from speech recognition results of a proxy language speech recognizer. I applied this method to Japanese and evaluated the naturalness of its output. Experimental results showed that an appropriate phoneme insertion penalty and iteration count for training and recognition were important for the proposed method. The proposed TTS system that does not use language-specific knowledge could synthesize more natural speech compared with that from a grapheme-based TTS system. To improve the proposed method, the impact of each component was analyzed. The results suggest that pause insertion accuracy, speech recognition accuracy, and phoneset of speech recognizer affected objective measures.

Additionally, I applied the proposed method to six Indian languages. Subjective experiments of the Blizzard Challenge 2015 showed that the proposed system achieved higher naturalness than a baseline system of unit selection framework in four languages out of six languages. In terms of speaker similarity, the proposed system outperformed the baseline system in all languages.

Future work will include a multilingual speaker-independent speech recognizer based on the international phonetic alphabet (IPA) [65] or GlobalPhone [66] to obtain accurate phoneme sequences. Furthermore, investigations of prosodic attributes, e.g. accent, stress, and tone, and languages not written with space between words, e.g. Mandarin, Japanese, and Thai, will be needed in order to establish a more language-independent method. Additionally, I will perform experiments on various written languages.

Chapter 4

A Bayesian framework for image recognition based on hidden Markov eigen-image models

4.1 Background

Image recognition is a technique for identifying objects in an image. Typical applications include biometrics authentication, e.g., fingerprint and face, optical character recognition (OCR), and general object recognition. As computer processing power increases, machine learning approaches based on statistical learning theory have been successfully applied in the field of image recognition. Moreover, not only applying general statistical classifier, approaches considering the specific problems of image recognition, e.g., geometric variations such as size, location, and rotation, image size variations, lighting conditions, object deformation, and occlusion, have been actively studied.

Among the specific problems of image recognition, geometric variations of an object to be recognized are a serious problem in image recognition. Therefore, much research work has been conducted on this problem. These can broadly be divided into three approaches: 1) task-dependent normalization techniques, 2) local features, and 3) the integration of geometric invariants into model structures. For approach 1), task-dependent normalization techniques have been developed for each image recognition task [67, 68]. However, it is costly to develop a normalization technique for each task. For approach 2), bag-of-features (BoF) approaches using local features, e.g., scale-invariant feature transform (SIFT) [69], have been proposed as invariant to local geometric variation [70]. Unfortunately, these approaches cannot consider positional relationships because they respond to geometric variations by removing the global information of input images. Therefore, methods considering the positional relationships of local features have been developed [71, 72]. For approach 3), convolutional neural network (CNN) based techniques, which integrate geometric invariants into model structures, have achieved significant improvements in the image recognition field [73, 74]. In addition to the structure of the standard feed-forward neural networks used as model structures, CNNs have geometric invariants based on multiple convolutional and pooling layers. However, since pooling is independently performed in each local window, it is difficult to represent global geometric transforms over an entire image. Another way to integrate the normalization processes into model structures is using hidden Markov models (HMMs) [4,5]. Geometric matching between input images and model parameters is represented by discrete hidden variables and the normalization process is included in the calculation of probabilities. However, the extension of HMMs to two dimensions for two-dimensional data, e.g., pixel values of an image, generally leads to an exponential increase in the amount of computation needed for training. To overcome this problem, several low computational complexity HMM structures have been proposed [6–12]. Among them, separable lattice HMMs (SL-HMMs) have been proposed to reduce computational complexity while retaining outstanding properties that model two-dimensional data [12]. SL-HMMs can perform elastic matching in both the horizontal and vertical directions, which makes it possible to model not only invariances to the size and location of an object but also nonlinear warping in both dimensions. One of the advantages of SL-HMMs against CNNs is explicit modeling of the generative process, which can represent geometric variations over an entire image. Furthermore, some extensions to structures representing typical geometric variations that are seen in many image recognition tasks have already been proposed, e.g., a structure for rotational variations [13], a structure with multiple horizontal and vertical Markov chains [14], and explicit state duration modeling [15]. By selecting an appropriate model structure reflecting the data generation process for a target task, human knowledge can effectively be utilized as prior information, and this makes it possible to construct models with a small amount of training data. It is also an interesting property of SL-HMMs that images with various sizes can directly be used as inputs without image size normalization. However, SL-HMMs still have a limitation in their application to image recognition: observations are assumed to be generated independently from corresponding HMM states. It is insufficient to represent variations in images, e.g., lighting conditions and object deformation. To overcome this limitation, hidden Markov eigen-image models (HMEMs) have been proposed [16]. The basic idea of the HMEMs is that eigen-images [17, 18] are generated from an SL-HMM. In the HMEM, the eigen-images are represented by probabilistic hidden variable models, such as factor analysis (FA) [19–21] or probabilistic principal component analysis (PPCA) [22]. Therefore, HMEMs have the good properties of both SL-HMMs and FA/PPCA: size and location invariant image recognition and a linear feature extraction based on statistical analysis.

In some image recognition tasks, only a small amount of training data is available, therefore efforts to achieve a high generalization ability are required. However, the training of HMEMs easily falls into the over-fitting problem because HMEMs have a complex model structure. Also, the maximum likelihood (ML) criterion has typically been used in training HMEMs [16]. Since the ML criterion produces a point estimate of model parameters, the estimation accuracy may be degraded, especially when there is insufficient training data. In this study, I focus on estimating HMEMs with a high generalization ability by using the Bayesian criterion. The Bayesian criterion assumes that model parameters are random variables, and a high generalization ability can be obtained by marginalizing all model parameters in estimating predictive distributions. Moreover, the Bayesian criterion can utilize prior distributions representing useful prior information on model parameters. However, the Bayesian criterion requires complicated integral and expectation computations to obtain posterior distributions when models have hidden variables. To overcome this problem, the variational Bayesian (VB) method [23] has been proposed as an approximation method. Additionally, to alleviate the local maximum problem dependent on the initial parameters, I apply the deterministic annealing expectation maximization (DAEM) algorithm [24, 25] to the training of HMEMs using the VB method. I show that the VB method applying the DAEM algorithm can significantly improve the performance in image recognition experiments. Third, approaches to image recognition based on CNNs are performed for comparison with the proposed method. Comparative experiment results show that the proposed method is more robust to geometric variations than CNNs when the amount of training data is insufficient.

4.2 Hidden Markov eigen-image models

4.2.1 Probabilistic eigen-image models

Factor analysis (FA) [19, 20] and probabilistic principal component analysis (PPCA) [22] are statistical methods for modeling the covariance structure with a small number of hidden variables. In this paper, I call them probabilistic eigen-image models (PEMs). In image modeling using PEMs, an image is assumed to be a fixed-length T-dimensional observation vector $\boldsymbol{o} = [\, o_1 \; o_2 \; \cdots \; o_T \,]^\top$. Then, observation vector $\boldsymbol{o}$ is assumed to be generated from a G-dimensional factor vector $\boldsymbol{x} = [\, x_1 \; x_2 \; \cdots \; x_G \,]^\top$ $(G < T)$ and a T-dimensional noise vector $\boldsymbol{v}$:

$$\boldsymbol{o} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{v}, \tag{4.1}$$

where $\boldsymbol{W} = [\, \boldsymbol{w}_1 \; \boldsymbol{w}_2 \; \cdots \; \boldsymbol{w}_G \,]$ is a $T \times G$ matrix known as the factor loading matrix. The factor vector $\boldsymbol{x}$ is a hidden variable assumed to be distributed in accordance with a

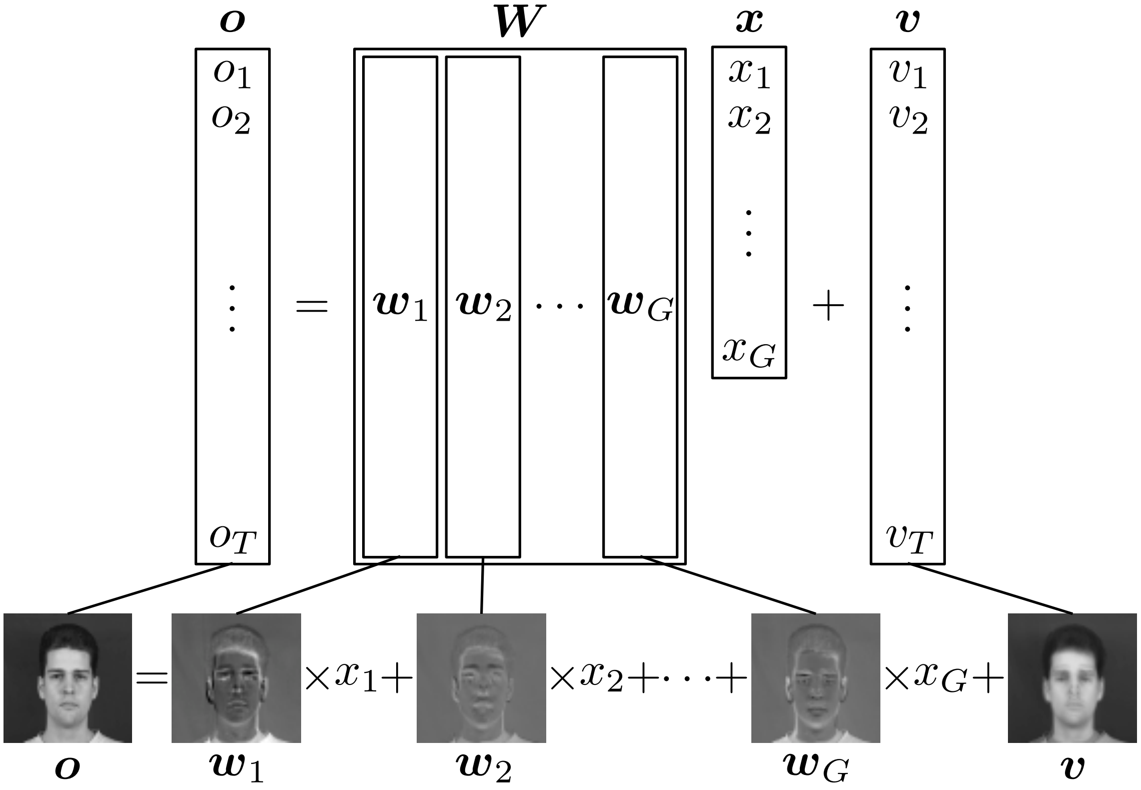


Figure 4.1: Model structure of PEMs in face image modeling

Gaussian distribution $\mathcal{N}(\boldsymbol{x}\,|\,\boldsymbol{0},\boldsymbol{I})$ where $\boldsymbol{0}$ and $\boldsymbol{I}$ respectively denote the zero vector and the identity matrix, and the noise vector $\boldsymbol{v}$ is distributed in accordance with $\mathcal{N}(\boldsymbol{v}\,|\,\boldsymbol{\mu},\boldsymbol{\Sigma})$. If $\boldsymbol{\Sigma}$ is assumed to be a diagonal covariance matrix $\boldsymbol{\Sigma} = \mathrm{diag}[\,\sigma_1^2\;\sigma_2^2\;\cdots\;\sigma_T^2\,]$, this model is called FA, and PPCA is a special case of FA in which the noise covariance matrix is isotropic, i.e., $\boldsymbol{\Sigma} = \sigma^2\boldsymbol{I}$. Figures 4.1 and 4.2 respectively show the model structure of PEMs in face image modeling and the graphical model representation of PEMs. The likelihood of observation $\boldsymbol{o}$ given factor vector $\boldsymbol{x}$ and model parameters $\boldsymbol{\Lambda}$ can be written as:

$$P(\boldsymbol{o}|\,\boldsymbol{x},\boldsymbol{\Lambda}) = \mathcal{N}(\boldsymbol{o}|\,\boldsymbol{W}\boldsymbol{x}+\boldsymbol{\mu},\boldsymbol{\Sigma}) \tag{4.2}$$

because the product $\boldsymbol{W}\boldsymbol{x}$ becomes a constant vector added to the noise vector $\boldsymbol{v}$. The marginal distribution of observation $\boldsymbol{o}$ is obtained by integrating out the hidden variable $\boldsymbol{x}$:

$$\begin{aligned} P(\boldsymbol{o}|\,\boldsymbol{\Lambda}) &= \int P(\boldsymbol{o}|\,\boldsymbol{x},\boldsymbol{\Lambda})P(\boldsymbol{x})\mathrm{d}\boldsymbol{x} \\ &= \mathcal{N}(\boldsymbol{o}|\,\boldsymbol{\mu},\boldsymbol{W}\boldsymbol{W}^\top+\boldsymbol{\Sigma}). \end{aligned} \tag{4.3}$$

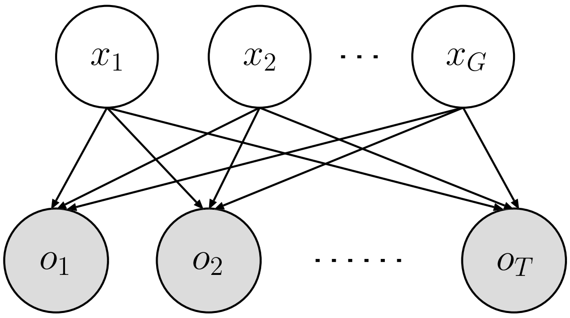


Figure 4.2: Graphical model representation of PEMs. Circles represent random variables, clear means hidden, and shaded means observed.

From the above equation, it is obvious that PEMs are a Gaussian distribution whose covariance matrix is constrained by the factor loading matrix $\boldsymbol{W}$ and the noise covariance matrix $\boldsymbol{\Sigma}$. That is, PEMs can capture the correlation among observations with a small number of parameters instead of using a full covariance matrix.

4.2.2 Separable lattice hidden Markov models

In the case that observations are two-dimensional data, e.g., pixel values of an image, observations are assumed to be given on a two-dimensional lattice as:

$$\boldsymbol{o} = \{\boldsymbol{o_t} \,|\, \boldsymbol{t} = (t^{(1)}, t^{(2)}) \in \boldsymbol{T}\}, \tag{4.4}$$

where $\boldsymbol{T} = \{(1,1),(1,2),\ldots,(1,T^{(2)}),(2,1),\ldots,(t^{(1)},t^{(2)}),\ \ldots,(T^{(1)},T^{(2)})\}$ denotes the two-dimensional image lattice, $\boldsymbol{t}$ denotes a two-dimensional coordinate lattice, $t^{(m)}$ is the coordinate of the m-th dimension, $T^{(m)}$ is the number of coordinates in the m-th dimension, and $m \in \{1,2\}$ denotes the dimension index representing horizontal and vertical direction. In two-dimensional HMMs, observation $\boldsymbol{o_t}$ is emitted from a state indicated by hidden variable $\boldsymbol{z_t}$. The hidden variables $\boldsymbol{z_t} \in \boldsymbol{K}$ can take one of the $K^{(1)}K^{(2)}$ states, which are assumed to be arranged on a two-dimensional state lattice $\boldsymbol{K} = \{(1,1),(1,2),\ldots,(1,K^{(2)}),(2,1),\ldots,(k^{(1)},k^{(2)})\ldots,\ (K^{(1)},K^{(2)})\}$, where $K^{(m)}$ is the number of states in the m-th dimension. In other words, a set of hidden variables represents a segmentation of observations into the $K^{(1)}K^{(2)}$ states, and each state corresponds to a segmented region in which the observation vectors are assumed to be generated from the same distribution. The number of possible state sequences in two-dimensional HMMs is $(K^{(1)}K^{(2)})^{T^{(1)}T^{(2)}}$. Therefore, standard two-dimensional HMMs require high computational costs.

Separable lattice hidden Markov models (SL-HMMs) have been proposed to reduce com-

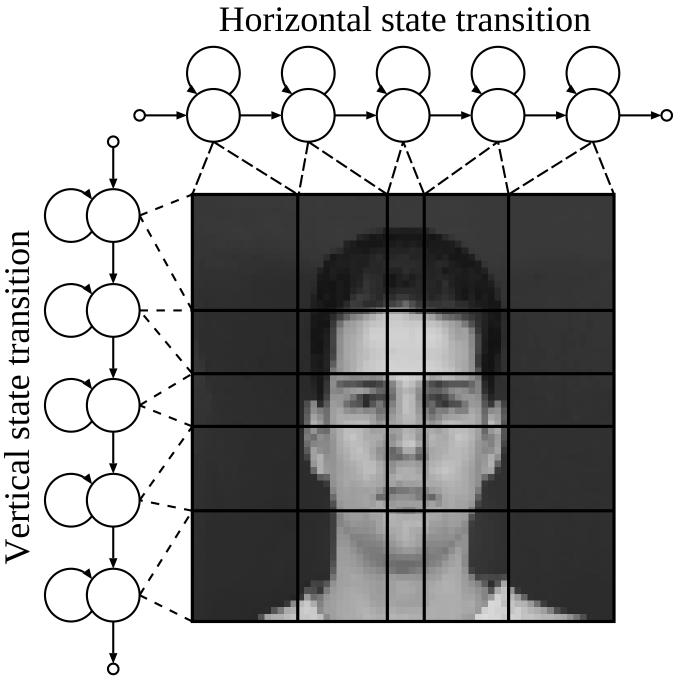


Figure 4.3: Model structure of SL-HMMs in face image modeling

putational complexity [12]. In SL-HMMs, to reduce the number of possible state sequences, hidden variables are constrained to be composed of two Markov chains as follows.

$$\boldsymbol{z} = \{\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}\}, \tag{4.5}$$

$$\boldsymbol{z}^{(m)} = \{z^{(m)}_{t^{(m)}} \,|\, 1 \leq t^{(m)} \leq T^{(m)}\}, \tag{4.6}$$

where $\boldsymbol{z}^{(m)}$ is the Markov chain along with the m-th coordinate, and $z^{(m)}_{t^{(m)}} \in \{1, \ldots, K^{(m)}\}$. The composite structure of hidden variables in SL-HMMs is defined as the product of hidden state sequences as:

$$\boldsymbol{z_t} = (z^{(1)}_{t^{(1)}}, z^{(2)}_{t^{(2)}}). \tag{4.7}$$

This means that hidden state sequences are independent of each dimension and the segmented regions of observations are constrained to rectangles. That is, it allows an observation lattice to be elastic both horizontally and vertically. Using this structure, the number of possible state sequences can be reduced from $(K^{(1)}K^{(2)})^{T^{(1)}T^{(2)}}$ to $(K^{(1)})^{T^{(1)}}(K^{(2)})^{T^{(2)}}$.

Figures 4.3 and 4.4 respectively show the model structure of SL-HMMs in face image modeling and the graphical model representation of SL-HMMs. The likelihood of ob-

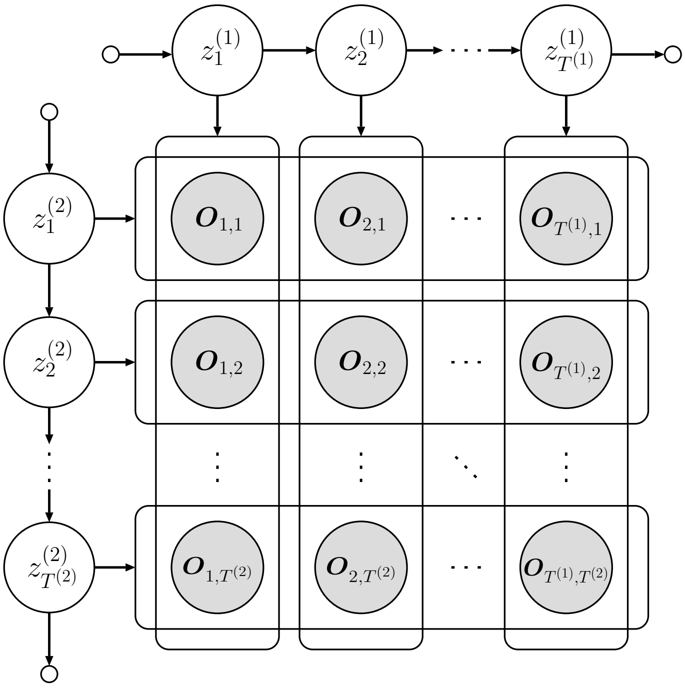


Figure 4.4: Graphical model representation of SL-HMMs. Rounded boxes represent group of variables, and arrows pointing to each box represent dependency in regard to all variables in box instead of drawing arrows to all variables.

servations $\boldsymbol{o}$ is obtained by summing the hidden variable $\boldsymbol{z}$:

$$P(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}) = \sum_{\boldsymbol{z}} P(\boldsymbol{o}\,|\,\boldsymbol{z},\boldsymbol{\Lambda})P(\boldsymbol{z}\,|\,\boldsymbol{\Lambda}). \tag{4.8}$$

In the application of image modeling, SL-HMMs can perform elastic matching in both the horizontal and vertical directions by assuming transition probabilities with left-to-right and top-to-bottom topologies, which makes it possible to model not only invariances to the size and location of an object but also nonlinear warping in both dimensions.

4.2.3 Hidden Markov eigen-image models

A hidden Markov eigen-image model (HMEM) is defined as a model integrating a PEM and an SL-HMM [16]. The basic idea of HMEMs is that eigen-images are generated from an SL-HMM. Figures 4.5 and 4.6 show the model structure of HMEMs in face image modeling and the graphical model representation of HMEMs, respectively. The

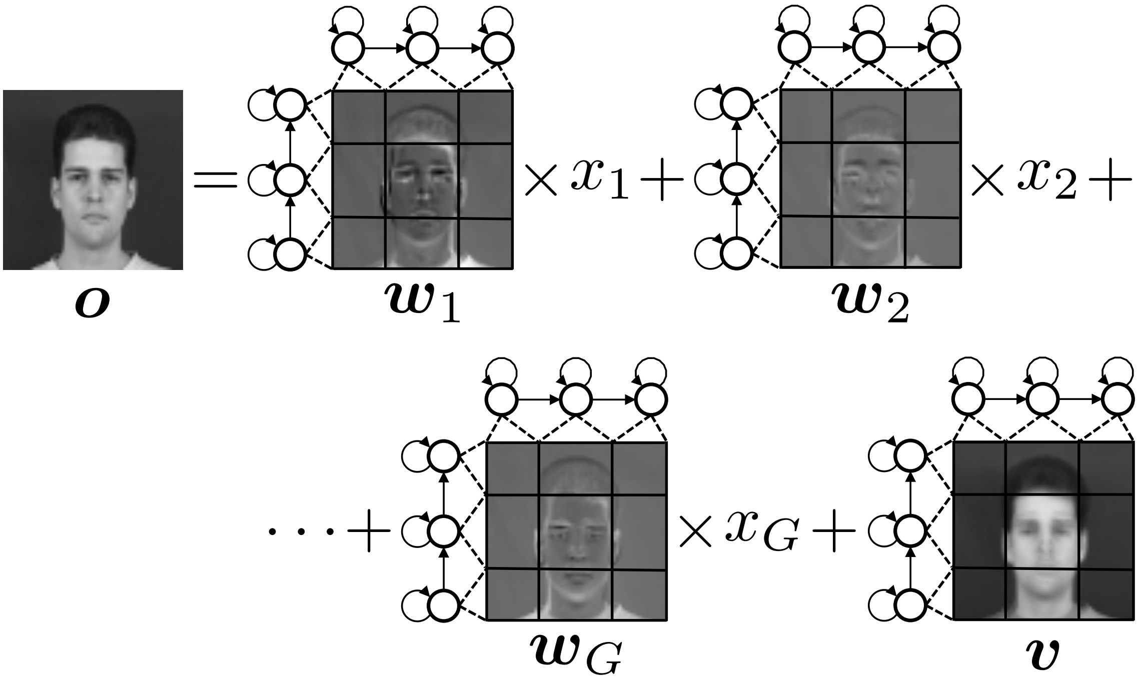


Figure 4.5: Model structure of HMEMs in face image modeling

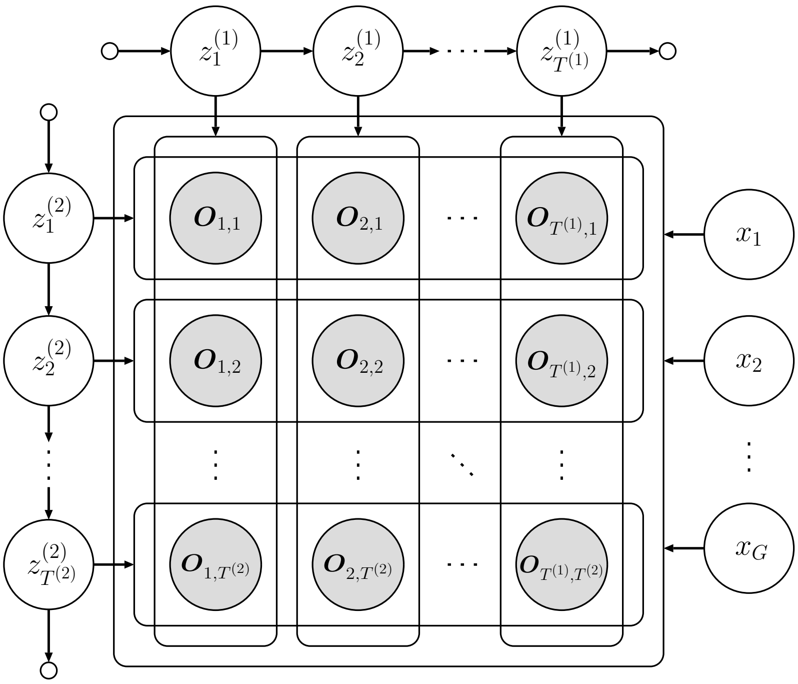


Figure 4.6: Graphical model representation of HMEMs

likelihood function of HMEMs is defined as follows.

$$P(\boldsymbol{o}\,|\,\boldsymbol{\Lambda}) = \sum_{\boldsymbol{z}} \int P(\boldsymbol{o}\,|\,\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda}) P(\boldsymbol{x}) P(\boldsymbol{z}\,|\,\boldsymbol{\Lambda}) \mathrm{d}\boldsymbol{x}, \tag{4.9}$$

$$P(\boldsymbol{o}\,|\,\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda}) = \prod_{\boldsymbol{t}} P(\boldsymbol{o_t}\,|\,\boldsymbol{x}, \boldsymbol{z_t}, \boldsymbol{\Lambda}), \tag{4.10}$$

$$P(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}\,|\,\boldsymbol{0}, \boldsymbol{I}), \tag{4.11}$$

$$P(\boldsymbol{z}\,|\,\boldsymbol{\Lambda}) = \prod_{m=1}^{2} \left[P(z_1^{(m)}\,|\,\boldsymbol{\Lambda}) \prod_{t^{(m)}=2}^{T^{(m)}} P(z_{t^{(m)}}^{(m)}\,|\,z_{t^{(m)}-1}^{(m)}, \boldsymbol{\Lambda}) \right], \tag{4.12}$$

where $\boldsymbol{x}$ is a factor vector and $\boldsymbol{z}$ represents state variables as used in SL-HMMs. The model parameters $\boldsymbol{\Lambda}$ of HMEMs are summarized as follows.

$$\boldsymbol{\Lambda} = \{\boldsymbol{\pi}^{(1)}, \boldsymbol{\pi}^{(2)}, \boldsymbol{a}^{(1)}, \boldsymbol{a}^{(2)}, \boldsymbol{b}\}. \tag{4.13}$$

1) $\boldsymbol{\pi}^{(m)} = \{\pi_{k^{(m)}}^{(m)}\,|\,1 \leq k^{(m)} \leq K^{(m)}\}$: an initial state probability distribution. The probability of state $k^{(m)}$ at $t^{(m)} = 1$ is represented by:

$$\pi_{k^{(m)}}^{(m)} = P(z_1^{(m)} = k^{(m)}\,|\,\boldsymbol{\Lambda}). \tag{4.14}$$

2) $\boldsymbol{a}^{(m)} = \{a_{k^{(m)},\bar{k}^{(m)}}^{(m)}\,|\,1 \leq k^{(m)}, \bar{k}^{(m)} \leq K^{(m)}\}$: a state transition probability matrix. The probability of moving from state $k^{(m)}$ to state $\bar{k}^{(m)}$ is represented by:

$$a_{k^{(m)},\bar{k}^{(m)}}^{(m)} = P(z_{t^{(m)}}^{(m)} = \bar{k}^{(m)}\,|\,z_{t^{(m)}-1}^{(m)} = k^{(m)}, \boldsymbol{\Lambda}). \tag{4.15}$$

3) $\boldsymbol{b} = \{\boldsymbol{b_k}(\boldsymbol{o_t})\,|\,\boldsymbol{k} \in \boldsymbol{K}\}$: an output probability distribution. The probability of an observation $\boldsymbol{o_t}$ being generated from a factor $\boldsymbol{x}$ and state $\boldsymbol{k}$ is represented by $\boldsymbol{b_k}(\boldsymbol{o_t}) = P(\boldsymbol{o_t}\,|\,\boldsymbol{x}, \boldsymbol{z_t} = \boldsymbol{k}, \boldsymbol{\Lambda})$, where $\boldsymbol{k}$ denotes the two-dimensional state index in the two-dimensional state lattice $\boldsymbol{K}$. The output probability distribution of state $\boldsymbol{k}$ can be represented by:

$$\begin{aligned} P(\boldsymbol{o_t}\,|\,\boldsymbol{x}, \boldsymbol{z_t} = \boldsymbol{k}, \boldsymbol{\Lambda}) &= \mathcal{N}(\boldsymbol{o_t}\,|\,\boldsymbol{W_k}\boldsymbol{x} + \boldsymbol{\mu_k}, \boldsymbol{\Sigma_k}) \\ &= \mathcal{N}(\boldsymbol{o_t}\,|\,\tilde{\boldsymbol{W}}_{\boldsymbol{k}}\tilde{\boldsymbol{x}}, \boldsymbol{\Sigma_k}), \end{aligned} \tag{4.16}$$

where $\boldsymbol{W_k}$, $\boldsymbol{\mu_k}$, and $\boldsymbol{\Sigma_k}$ respectively denote the state level factor loading matrix, mean vector, and the diagonal covariance matrix in state $\boldsymbol{k}$. For simplicity, the extended factor loading matrix and factor vector are defined as:

$$\tilde{\boldsymbol{W}}_{\boldsymbol{k}} = [\,\boldsymbol{W_k}\ \boldsymbol{\mu_k}\,], \tag{4.17}$$

$$\tilde{\boldsymbol{x}} = [\,\boldsymbol{x}^\top\ 1\,]^\top. \tag{4.18}$$

By incorporating the state transition structure into the factor loading matrix, eigen-images can be transformed to match an input image, and this state transition structure performs size and location normalization. Once the state sequences are given, HMEMs are regarded as PEMs, which are given normalized data. Therefore, HMEMs overcome the limitation of SL-HMMs, i.e., the correlation among all observations can be modeled through the factor variables. Thus, HMEMs have the good properties of both PEMs and SL-HMMs: a linear feature extraction based on statistical analysis and invariances to the size and location of the object to be recognized. Moreover, the structure of HMEMs includes conventional PEMs and SL-HMMs as special cases; HMEMs with the same number of states as the number of pixels of the input images become the conventional PEMs [19,20,22], and HMEMs with a zero factor become the SL-HMMs [12].

4.3 Bayesian framework for training of HMEMs

4.3.1 Maximum likelihood criterion

In Bayesian statistics, it is important to estimate model with a high generalization ability from training data $\boldsymbol{o}$. The maximum likelihood (ML) criterion has typically been used to train statistical models in Bayesian statistics. The optimal model parameters $\boldsymbol{\Lambda}^{(\mathrm{ML})}$ are estimated in the ML criterion by maximizing the likelihood of training data $P(\boldsymbol{o} \,|\, \boldsymbol{\Lambda})$ as:

$$\boldsymbol{\Lambda}^{(\mathrm{ML})} = \arg\max_{\boldsymbol{\Lambda}} P(\boldsymbol{o} \,|\, \boldsymbol{\Lambda}). \tag{4.19}$$

The predictive distribution of testing data $\boldsymbol{o}^{(\mathrm{test})}$ in the testing stage is calculated by $P(\boldsymbol{o}^{(\mathrm{test})} \,|\, \boldsymbol{\Lambda}^{(\mathrm{ML})})$.

Since HMEMs have hidden variables $\boldsymbol{x}$ and $\boldsymbol{z}$, it is difficult to obtain an analytic solution to Eq. (4.19). The parameters of HMEMs can be estimated via the expectation maximization (EM) algorithm [31], which is a local maximum solution based on an iterative procedure. This procedure maximizes the expectation of the complete-data log-likelihood. However, calculating the expectation is still computationally intractable due to the combination of hidden variables $\boldsymbol{x}$ and $\boldsymbol{z}$. Variational methods have been used to approximate the EM algorithm in probabilistic graphical models with hidden variables [75]. An approximate posterior distribution is estimated by maximizing the lower bound of the log-marginal likelihood instead of the true log-likelihood. The variational method makes the parameter estimation of HMEMs possible with a realistic calculation time [16].

4.3.2 Bayesian criterion

Since the ML method produces a point estimate of model parameters, the estimation accuracy may be degraded due to the over-fitting problem when there is insufficient data. In this paper, I propose using the Bayesian framework for training HMEMs. The framework has two advantageous properties for model training: using prior distributions and marginalizing model parameters. Therefore, the framework can be expected higher generalization ability than the ML method.

The predictive distribution of the Bayesian criterion is given by:

$$P(\boldsymbol{o}^{(\mathrm{test})}\,|\,\boldsymbol{o}) = \int P(\boldsymbol{o}^{(\mathrm{test})}\,|\,\boldsymbol{\Lambda})P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o})\mathrm{d}\boldsymbol{\Lambda}. \tag{4.20}$$

Posterior distribution $P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o})$ for a set of model parameters $\boldsymbol{\Lambda}$ can be written with the Bayes theorem:

$$P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o}) = \frac{P(\boldsymbol{o}\,|\,\boldsymbol{\Lambda})P(\boldsymbol{\Lambda})}{P(\boldsymbol{o})}, \tag{4.21}$$

where $P(\boldsymbol{\Lambda})$ is a prior distribution and $P(\boldsymbol{o})$ is evidence. Model parameters $\boldsymbol{\Lambda}$ are estimated as posterior distribution $P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o})$, and posterior distribution is integrated out in Eq. (4.20) so that the effect of over-fitting is mitigated. That is, the Bayesian criterion has a higher generalization ability than the ML criterion when there is insufficient training data. However, the criterion requires complicated integral and expectation computations to obtain posterior distributions when models have hidden variables. Markov chain Monte Carlo (MCMC) [76] and variational Bayesian (VB) [23] methods have been proposed as approaches to approximation to overcome this problem.

4.3.3 Variational Bayesian method

Posterior distribution

An approximate posterior distribution is estimated in the VB method by maximizing the lower bound of log-marginal likelihood instead of the true likelihood. The lower bound of the log-marginal likelihood $\mathcal{F}$ is defined by using Jensen's inequality:

$$\begin{aligned}\ln P(\boldsymbol{o}) &= \ln \sum_{\boldsymbol{z}} \iint Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})\frac{P(\boldsymbol{o},\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})}{Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda} \\ &\geq \sum_{\boldsymbol{z}} \iint Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda}) \ln \frac{P(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})P(\boldsymbol{x})P(\boldsymbol{z}\,|\,\boldsymbol{\Lambda})P(\boldsymbol{\Lambda})}{Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda} \\ &\triangleq \mathcal{F}(Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})), \end{aligned} \tag{4.22}$$

where $Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})$ is an arbitrary distribution. The difference between the true log-marginal likelihood $\ln P(\boldsymbol{o})$ and the lower bound $\mathcal{F}$ is given by the Kullback-Leibler (KL) divergence $\mathrm{KL}[\cdot \,||\, \cdot]$ between the arbitrary distribution $Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})$ and the true posterior distribution $P(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda} \,|\, \boldsymbol{o})$ as:

$$\ln P(\boldsymbol{o}) - \mathcal{F}(Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})) = \mathrm{KL}[Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda}) \,||\, P(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda} \,|\, \boldsymbol{o})]. \tag{4.23}$$

Since the true log-marginal likelihood $\ln P(\boldsymbol{o})$ is independent of the arbitrary distribution $Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})$, maximizing the lower bound $\mathcal{F}$ is equivalent to minimizing the KL divergence. In other words, $Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})$ can be regarded as an approximation of the true posterior distribution $P(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda} \,|\, \boldsymbol{o})$.

To reduce computational complexity, random variables are assumed to be conditionally independent of one another, i.e.,

$$\begin{aligned} Q(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda}) &\approx Q(\boldsymbol{x})Q(\boldsymbol{z})Q(\boldsymbol{\Lambda}) \\ &\approx Q(\boldsymbol{x})Q(\boldsymbol{z}^{(1)})Q(\boldsymbol{z}^{(2)})Q(\boldsymbol{\Lambda}), \end{aligned} \tag{4.24}$$

where $Q(\boldsymbol{x}), Q(\boldsymbol{z}^{(m)})$, and $Q(\boldsymbol{\Lambda})$ are called VB posterior distributions that, respectively, satisfy $\int Q(\boldsymbol{x})\mathrm{d}\boldsymbol{x} = 1$, $\sum_{\boldsymbol{z}^{(m)}} Q(\boldsymbol{z}^{(m)}) = 1$, and $\int Q(\boldsymbol{\Lambda})\mathrm{d}\boldsymbol{\Lambda} = 1$. Under this assumption, the optimal VB posterior distributions $Q(\boldsymbol{x})$, $Q(\boldsymbol{z}^{(m)})$, and $Q(\boldsymbol{\Lambda})$ that maximize the objective function $\mathcal{F}$ are given as:

$$Q(\boldsymbol{x}) \propto P(\boldsymbol{x}) \exp\left[\sum_{\boldsymbol{z}} \int Q(\boldsymbol{z})Q(\boldsymbol{\Lambda}) \ln P(\boldsymbol{o} \,|\, \boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})\mathrm{d}\boldsymbol{\Lambda}\right], \tag{4.25}$$

$$Q(\boldsymbol{z}^{(m)}) \propto \exp\left[\sum_{\boldsymbol{z}^{(\bar{m})}} \iint Q(\boldsymbol{x})Q(\boldsymbol{z}^{(\bar{m})})Q(\boldsymbol{\Lambda}) \ln P(\boldsymbol{o} \,|\, \boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})P(\boldsymbol{z}^{(m)} \,|\, \boldsymbol{\Lambda})\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda}\right], \tag{4.26}$$

$$Q(\boldsymbol{\Lambda}) \propto P(\boldsymbol{\Lambda}) \exp\left[\sum_{\boldsymbol{z}} \int Q(\boldsymbol{x})Q(\boldsymbol{z}) \ln P(\boldsymbol{o} \,|\, \boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\Lambda})P(\boldsymbol{z} \,|\, \boldsymbol{\Lambda})\mathrm{d}\boldsymbol{x}\right], \tag{4.27}$$

where $\bar{m}$ represents the dimension index that is an alternative to the m-th dimension. Since the VB posterior distributions in Eqs. (4.25)–(4.27) are dependent on each other, these updates need to be iterated by using the EM algorithm.

$$\begin{aligned} &\text{(VB E-step):} \\ &\quad Q^{(i+1)}(\boldsymbol{x}) = \arg\max_{Q(\boldsymbol{x})} \mathcal{F}(Q(\boldsymbol{x})Q^{(i)}(\boldsymbol{z})Q^{(i)}(\boldsymbol{\Lambda})) \\ &\quad Q^{(i+1)}(\boldsymbol{z}) = \arg\max_{Q(\boldsymbol{z})} \mathcal{F}(Q^{(i+1)}(\boldsymbol{x})Q(\boldsymbol{z})Q^{(i)}(\boldsymbol{\Lambda})) \\ &\text{(VB M-step):} \\ &\quad Q^{(i+1)}(\boldsymbol{\Lambda}) = \arg\max_{Q(\boldsymbol{\Lambda})} \mathcal{F}(Q^{(i+1)}(\boldsymbol{x})Q^{(i+1)}(\boldsymbol{z})Q(\boldsymbol{\Lambda})) \end{aligned}$$

The update equations increase the value of the objective function $\mathcal{F}$ at each iteration until convergence by adding 1 to iteration count i. The details of VB E- and M-step are given in Appendix A.1. In the proposed method, HMEMs that have the same number of states as the number of pixels of the input images are equivalent to the conventional FA using the VB method [21], and HMEMs with a zero factor become the SL-HMMs using the VB method [77].

Prior distribution

The VB method has an advantage in that it can utilize prior distributions representing useful prior information on model parameters. Although arbitrary distributions can be used as prior distributions, conjugate prior distributions are widely used as prior distributions. A conjugate prior distribution is a distribution where the resulting posterior distribution belongs to the same distribution family as the prior distribution. The conjugate prior distributions of an HMEM are defined as:

$$\begin{aligned} P(\boldsymbol{\Lambda}) = & \prod_{m=1}^{2}\left[\mathcal{D}(\boldsymbol{\pi}^{(m)}\,|\,\boldsymbol{\phi}^{(m)})\prod_{k^{(m)}=1}^{K^{(m)}}\mathcal{D}(\boldsymbol{a}^{(m)}_{k^{(m)}}\,|\,\boldsymbol{\alpha}^{(m)}_{k^{(m)}})\right] \\ & \times\prod_{\boldsymbol{k}}\prod_{d=1}^{D}\mathcal{N}(\tilde{\boldsymbol{w}}_{\boldsymbol{k},d}\,|\,\boldsymbol{h}_{\boldsymbol{k},d},\boldsymbol{U}_{\boldsymbol{k}}^{-1}\sigma^2_{\boldsymbol{k},d})\mathcal{G}((\sigma^2_{\boldsymbol{k},d})^{-1}\,|\,\eta_{\boldsymbol{k}},\nu_{\boldsymbol{k},d}), \end{aligned} \tag{4.28}$$

where D is the dimension of observation $\boldsymbol{o}$, $\mathcal{D}(\cdot)$ is a Dirichlet distribution, $\mathcal{N}(\cdot)\mathcal{G}(\cdot)$ is a Gauss-gamma distribution, and $\tilde{\boldsymbol{w}}_{\boldsymbol{k},d}$ and $\sigma^2_{\boldsymbol{k},d}$ are defined as:

$$\boldsymbol{W}_{\boldsymbol{k}} = [\,\boldsymbol{w}_{\boldsymbol{k},1}\ \boldsymbol{w}_{\boldsymbol{k},2}\ \cdots\ \boldsymbol{w}_{\boldsymbol{k},D}\,]^{\top}, \tag{4.29}$$

$$\boldsymbol{w}_{\boldsymbol{k},d} = [\,w_{\boldsymbol{k},d,1}\ w_{\boldsymbol{k},d,2}\ \cdots\ w_{\boldsymbol{k},d,G}\,]^{\top}, \tag{4.30}$$

$$\boldsymbol{\mu}_{\boldsymbol{k}} = [\,\mu_{\boldsymbol{k},1}\ \mu_{\boldsymbol{k},2}\ \cdots\ \mu_{\boldsymbol{k},D}\,]^{\top}, \tag{4.31}$$

$$\tilde{\boldsymbol{w}}_{\boldsymbol{k},d} = [\,\boldsymbol{w}_{\boldsymbol{k},d}^{\top}\ \mu_{\boldsymbol{k},d}\,], \tag{4.32}$$

$$\boldsymbol{\Sigma}_{\boldsymbol{k}} = \mathrm{diag}[\,\sigma^2_{\boldsymbol{k},1}\ \sigma^2_{\boldsymbol{k},2}\ \cdots\ \sigma^2_{\boldsymbol{k},D}\,], \tag{4.33}$$

where $\boldsymbol{W}_{\boldsymbol{k}}$ is assumed to be independent of each dimension. These distributions can be represented by a set of hyper-parameters $\{\boldsymbol{\phi}^{(1)},\boldsymbol{\phi}^{(2)},\boldsymbol{\alpha}^{(1)}_{k^{(1)}},\boldsymbol{\alpha}^{(2)}_{k^{(2)}},\boldsymbol{h}_{\boldsymbol{k},d},\boldsymbol{U}_{\boldsymbol{k}},\eta_{\boldsymbol{k}},\nu_{\boldsymbol{k},d}\}$. When conjugate prior distributions are used for prior distributions, the posterior distributions are represented by the same parameter set $\{\hat{\boldsymbol{\phi}}^{(1)},\hat{\boldsymbol{\phi}}^{(2)},\hat{\boldsymbol{\alpha}}^{(1)}_{k^{(1)}},\hat{\boldsymbol{\alpha}}^{(2)}_{k^{(2)}},\hat{\boldsymbol{h}}_{\boldsymbol{k},d},\hat{\boldsymbol{U}}_{\boldsymbol{k}},\hat{\eta}_{\boldsymbol{k}},\hat{\nu}_{\boldsymbol{k},d}\}$. Figures 4.7 and 4.8 respectively show a graphical model representation with the model parameters of HMEMs using the ML and VB methods.

Since the prior distributions of model parameters affect the estimation of posterior distributions in the VB method, determining prior distributions is a serious problem in estimat-

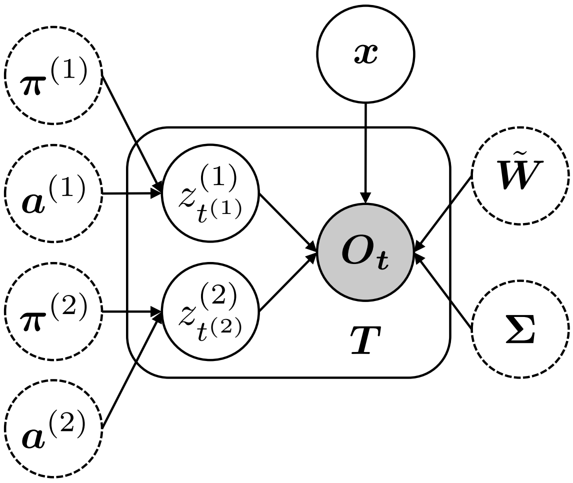


Figure 4.7: Graphical model representation for ML method of HMEMs. Dotted circles represent model parameters, and plate $\boldsymbol{T}$ is abbreviation of $\boldsymbol{T}$ nodes.

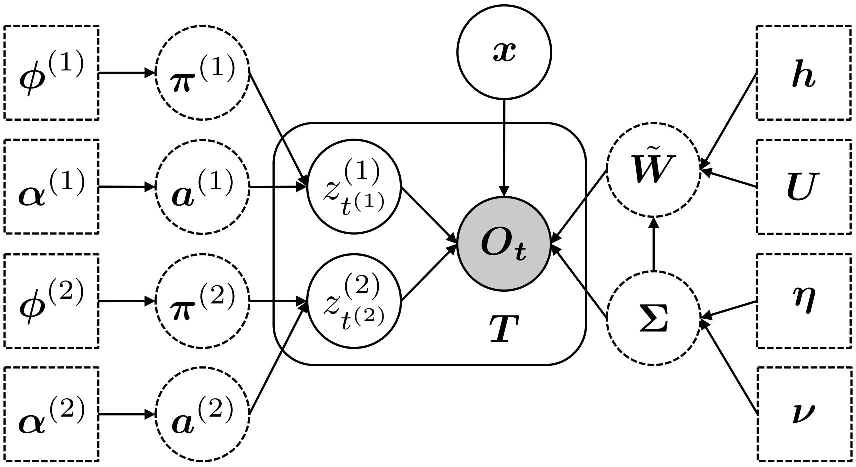


Figure 4.8: Graphical model representation for VB method of HMEMs. Dotted rectangles represent hyper-parameters.

ing appropriate models. I set the prior distribution as:

$$P(\boldsymbol{\Lambda}) \propto P(\boldsymbol{\Lambda} \,|\, \boldsymbol{o}^{(\mathrm{prior})})^{\tau}, \tag{4.34}$$

where $\boldsymbol{o}^{(\mathrm{prior})}$ is data given in advance (I call this prior data). I can control the influence of the prior distribution on the posterior distribution by adjusting tuning parameters τ. The hyper-parameters based on prior data $\boldsymbol{o}^{(\mathrm{prior})}$ are given in Appendix A.2.

Predictive distribution

Predictive distribution $P(\boldsymbol{o}^{(\mathrm{test})}\,|\,\boldsymbol{o})$ is calculated using Eq. (4.20). Although predictive distribution includes a complicated expectation calculation, the same approximation based on the VB method as that in posterior distribution training can be applied, and the lower bound $\mathcal{F}^{(\mathrm{test})}$ is defined as:

$$\begin{aligned}&\mathcal{F}^{(\mathrm{test})}(Q(\boldsymbol{x})Q(\boldsymbol{z})Q(\boldsymbol{\Lambda}))\\&\triangleq \sum_{\boldsymbol{z}}\iint Q(\boldsymbol{x})Q(\boldsymbol{z})Q(\boldsymbol{\Lambda})\ln\frac{P(\boldsymbol{o}^{(\mathrm{test})},\boldsymbol{x},\boldsymbol{z}\,|\,\boldsymbol{\Lambda})P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o})}{Q(\boldsymbol{x})Q(\boldsymbol{z})Q(\boldsymbol{\Lambda})}\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda}.\end{aligned} \tag{4.35}$$

Additionally, the posterior distribution $P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o})$ is approximated to $Q(\boldsymbol{\Lambda})$. From this approximation, the predictive distribution can update only $Q(\boldsymbol{x})$ and $Q(\boldsymbol{z})$ (VB E-step).

In image recognition based on HMEMs using the VB method, posterior distributions $P(\boldsymbol{\Lambda}\,|\,\boldsymbol{o}_c)$ are trained by using images for each class c, i.e., subject, separately in the training stage. Then, in the testing stage, the likelihood of testing data $\boldsymbol{o}^{(\mathrm{test})}$, which is calculated by the predictive distribution $P(\boldsymbol{o}^{(\mathrm{test})}\,|\,\boldsymbol{o}_c)$, is compared among all subjects. The class that obtains the highest likelihood is chosen as the identification result.

4.3.4 Deterministic annealing EM algorithm

An iterative procedure, such as the EM algorithm, suffers from the local maximum problem dependent on the initial parameters, especially models with a complex structure. A deterministic annealing EM (DAEM) algorithm has been proposed to relax this problem [24, 25]. In this paper, I apply the DAEM algorithm to the training of HMEMs using the VB method.

In this paper, negative free energy $f(\boldsymbol{o},\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})$ is defined by using four temperature parameters as:

$$f(\boldsymbol{o},\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})\triangleq\ln\sum_{\boldsymbol{z}}\iint P^{\beta_1}(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})P^{\beta_2}(\boldsymbol{x})P^{\beta_3}(\boldsymbol{z}\,|\,\boldsymbol{\Lambda})P^{\beta_4}(\boldsymbol{\Lambda})\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda}, \tag{4.36}$$

where $\beta_1, \beta_2, \beta_3$, and β_4 are respectively the temperature parameter of the output probability distributions, the factor probability distributions, the initial and state transition probability distributions, and the prior distributions. The effect of each distribution can be controlled appropriately by introducing multiple temperature parameters.

Instead of Eq. (4.22), a lower bound $\mathcal{F}^{(\mathrm{DAEM})}$ for the VB DAEM algorithm is defined by using Jensen's inequality:

$$\begin{aligned}&\mathcal{F}^{(\mathrm{DAEM})}(Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda}))\\ &\triangleq \sum_{\boldsymbol{z}} \iint Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda}) \ln \frac{P^{\beta_1}(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})P^{\beta_2}(\boldsymbol{x})P^{\beta_3}(\boldsymbol{z}\,|\,\boldsymbol{\Lambda})P^{\beta_4}(\boldsymbol{\Lambda})}{Q(\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})} \mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda}. \end{aligned} \tag{4.37}$$

To reduce computational complexity, random variables $\boldsymbol{x}, \boldsymbol{z}^{(m)}$, and $\boldsymbol{\Lambda}$ are assumed to be conditionally independent of one another, which is the same as Eq. (4.24). The optimal VB posterior distributions $Q(\boldsymbol{x}), Q(\boldsymbol{z}^{(m)})$, and $Q(\boldsymbol{\Lambda})$ that maximize the objective function $\mathcal{F}^{(\mathrm{DAEM})}$ are given by the variational method as:

$$Q(\boldsymbol{x}) \propto P^{\beta_2}(\boldsymbol{x}) \exp\left[\sum_{\boldsymbol{z}} \int Q(\boldsymbol{z})Q(\boldsymbol{\Lambda}) \ln P^{\beta_1}(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})\mathrm{d}\boldsymbol{\Lambda}\right], \tag{4.38}$$

$$Q(\boldsymbol{z}^{(m)}) \propto \exp\left[\sum_{\boldsymbol{z}^{(\bar{m})}} \iint Q(\boldsymbol{x})Q(\boldsymbol{z}^{(\bar{m})})Q(\boldsymbol{\Lambda}) \ln P^{\beta_1}(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})P^{\beta_3}(\boldsymbol{z}^{(m)}\,|\,\boldsymbol{\Lambda})\mathrm{d}\boldsymbol{x}\mathrm{d}\boldsymbol{\Lambda}\right], \tag{4.39}$$

$$Q(\boldsymbol{\Lambda}) \propto P^{\beta_4}(\boldsymbol{\Lambda}) \exp\left[\sum_{\boldsymbol{z}} \int Q(\boldsymbol{x})Q(\boldsymbol{z}) \ln P^{\beta_1}(\boldsymbol{o}\,|\,\boldsymbol{x},\boldsymbol{z},\boldsymbol{\Lambda})P^{\beta_3}(\boldsymbol{z}\,|\,\boldsymbol{\Lambda})\mathrm{d}\boldsymbol{x}\right]. \tag{4.40}$$

By applying the DAEM algorithm, the temperature parameters β_l $(l = 1, \ldots, 4)$ are attached to the original VB posterior distributions in Eqs. (4.25)–(4.27). The VB posterior distributions are given in Appendix A.3.

In the annealing process, the temperature parameters β_l are gradually increased from $\beta_l \simeq 0$ to $\beta_l = 1$. When $\beta_l \simeq 0$, the VB posterior distributions take a form with nearly uniform distribution. While the temperature parameter is increasing, the form of the distributions becomes close to that of the original VB posterior distributions. Finally, at $\beta_l = 1$, the distributions take the form of the original VB posterior distributions. Since the DAEM algorithm reduces the effect of the local maximum problem dependent on the initial parameters, reliable model parameters can be estimated.

4.4 Experiments

4.4.1 Conditions

Face recognition experiments were conducted on the XM2VTS database [78] to evaluate the effectiveness of the proposed method. The experimental conditions are summarized in Table 4.1. I prepared two datasets for these experiments. **Dataset1** did not include

Table 4.1: Experimental conditions

Database	XM2VTS [78]	
Original image size	720×576	
Datasets	**Dataset1**	**Dataset2**
Cropped image size	550×550	480×480–720×720
Center coordinates of cropping	(360, 288)	$(360 \pm 80, 288 \pm 20)$
Subsampled image size	64×64, grayscale	
Number of subjects (classes)	100	
Number of training data	6 images per subject	
Number of testing data	2 images per subject	
Compared methods	**FA/PPCA-VB/ML-DAEM/EM**	
HMM structure	Left-to-right and top-to-bottom without skip transitions	
Number of HMM states	40×40	
Number of factors	0, 1, 2, 3, 4, 5	
Prior distribution	Universal background model	
Tuning parameter τ	$\frac{1}{100}, \frac{1}{500}, \frac{1}{1000}, \frac{1}{2000}, \frac{1}{3000}, \frac{1}{4000}, \frac{1}{5000}$	
Schedule of temperature θ_l	$\theta_{1,2,3} = 2^0$, $\theta_4 = 2^{-6}$	

large size and location variations, while **Dataset2** did. The cropped image sizes and center coordinates of cropping were randomly generated by the Gaussian distribution in **Dataset2**. Figure 4.9 shows some examples of images for the experiments.

In the prior distributions, I used all training images for all subjects as prior data. This is the same idea as that in the universal background model (UBM) [79]. The UBM was trained with the SL-HMM structure and was extended to the HMEM structure. I controlled the influence of the prior distribution on the posterior distribution by adjusting tuning parameters τ.

The temperature parameter $\beta_l(j)$ for the DAEM algorithm was updated by:

$$\beta_l(j) = \left(\frac{j}{J}\right)^{\theta_l}, \tag{4.41}$$

where $j = 1, \ldots, J$ denotes the number of iterations of temperature updates. When set to $J = 1$, this algorithm is the same as the EM algorithm. In these experiments, the number of temperature parameter updates was set to $J = 60$. The schedules of temperatures $\theta_{1,2,3} = 2^0$ and $\theta_4 = 2^{-6}$ were used, and I could stably obtain a high recognition performance with each method in the preliminary experiments.

The training recipe of **FA-VB-DAEM** is summarized in Table 4.2. In steps 1–2, prior

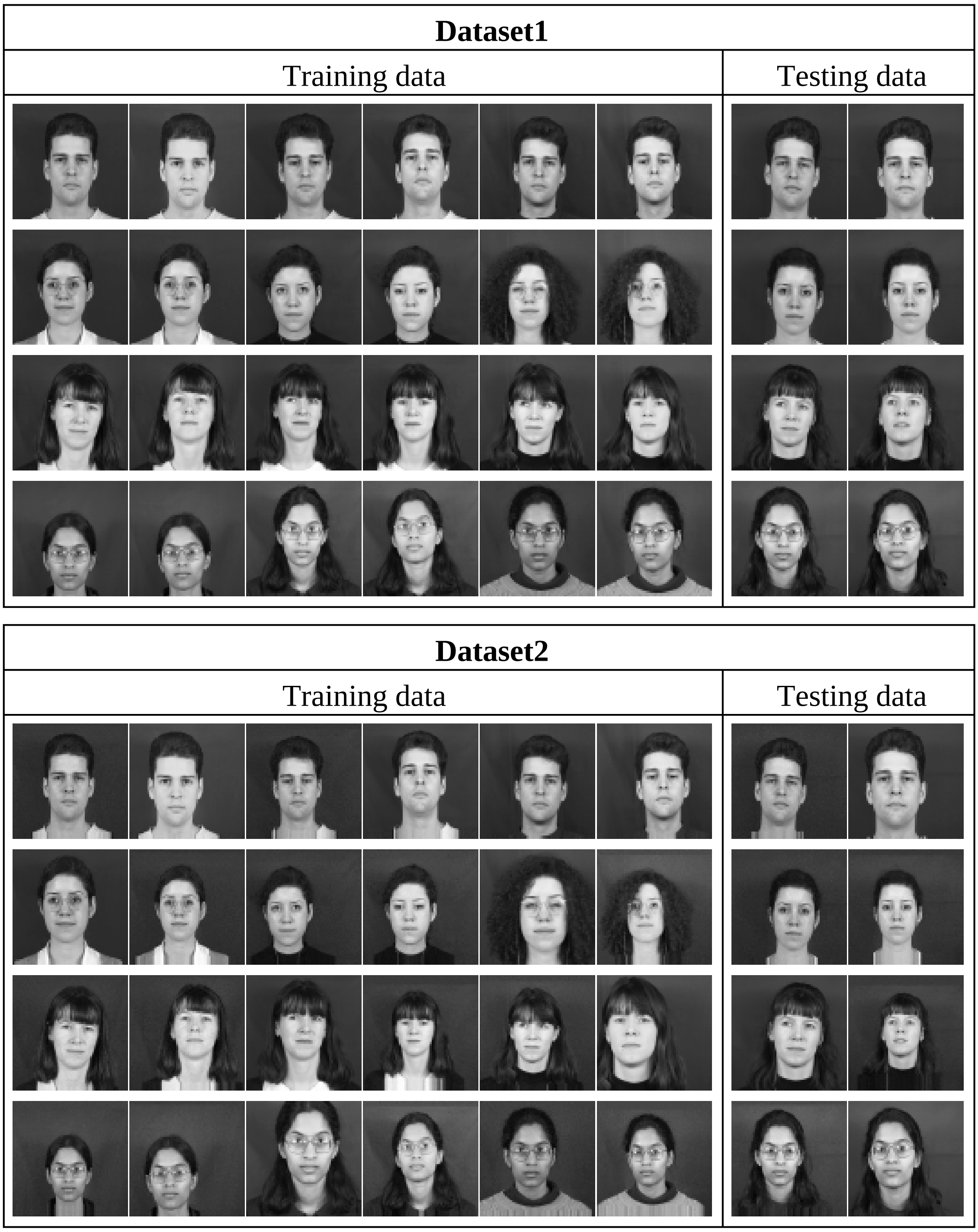


Figure 4.9: Examples of images for experiments

Table 4.2: Training recipe of **FA-VB-DAEM**

1. Train UBM from all training images for all subjects with flat hyper-parameters.
2. Set hyper-parameters Eqs. (A.27)–(A.32) from UBM by adjusting tuning parameter τ.
3. Set Eqs. (A.8) and (A.10) to uniform probabilities, Eq. (A.33) to zero vector, and Eq. (A.34) to identity matrix.
4. Compute Eqs. (A.36)–(A.41).
5. Update temperature parameters $\beta_l(j)$ using Eq. (4.41).
6. (VB E-step): Update Eqs. (A.13)–(A.15), Eqs. (A.8) and (A.10), and Eqs. (A.33) and (A.34).
7. (VB M-step): Update Eqs. (A.36)–(A.41).
8. Go to step 6 until convergence of lower bound $\mathcal{F}^{(\mathrm{DAEM})}$ using Eq. (4.37).
9. Go to step 5 by adding 1 to j until $j = J$.

distributions train all training images for all subjects. Steps 3–4 and 5–9 are respectively an initialization and iterative procedure, and they are performed for each class.

For comparison with the proposed method, two convolutional neural network (CNN)-based approaches (**CNN** and **CaffeNet**) were performed [73, 74]. For **CNN**, a CNN was trained by using a Caffe [80] based on both **Dataset1** and **Dataset2**. For **CaffeNet**, a pre-trained CNN (CaffeNet) [74, 80], which was trained by using the ImageNet Large Scale Visual Recognition Challenge 2012 (ILSVRC2012) dataset [81], was used to extract image features. The details of CNN approaches are as follows.

CNN: The architecture of the CNN model was $I(64,1){-}C(128,10,1,55){-}P(3,2,27)$ $-C(256,5,1,23) - P(3,2,11) - F(800) - F(600) - F(400) - O(100)$, where $I(i,d)$ indicates an input layer with a d dimensional $i \times i$ sized image, $C(f,w,s,o)$ indicates a convolutional layer with f filters of a $w \times w$ sized window with a stride of s and $o \times o$ sized output, $P(w,s,o)$ indicates a pooling layer, $F(n)$ indicates a fully connected layer with n units, and $O(c)$ indicates an output layer with c classes. The ReLU activation function was used in the convolutional and fully connected layers. The stochastic gradient descent (SGD) algorithm with mini-batch of size 200 was used for training and dropout with a probability of 0.5 was used to the convolutional and fully connected layers.

CaffeNet: The image-feature vectors were composed of 4096 dimensions extracting the pre-trained CaffeNet of the 7th fully-connected layer. The one-nearest neighbor was then used as the classifier.

4.4.2 Results

The subject of the bottom of Figure 4.9 in **Dataset2** was modeled by using three factors and tuning parameter $\tau = \frac{1}{2000}$ in order to visualize model parameters. Figure 4.10 shows the values of the mean vector $\boldsymbol{\mu}_k$ in Eq. (4.31), and the eigen-images $\boldsymbol{W}_k$ in Eq. (4.29) were represented in grayscale. From Figure 4.10, with the EM algorithm approaches **FA-VB/ML-EM**, the shape of the mean and eigen-images collapsed due to the local maximum problem. Although **FA-ML-DAEM** could comparatively smoothly represent the subjects' faces, the faces were modeled in two places because geometric variations were not fully absorbed by the SL-HMM structure. In comparison, it was confirmed that **FA-VB-DAEM** clearly preserved the subjects' faces, even though the training data included much geometric variation. Therefore, the proposed method overcame the over-fitting and local maximum problems and was able to absorb geometric variations. Additionally, the 1st eigen-image of **FA-VB-DAEM** seems to represent the hairstyle differences of the subject. These results indicate that the geometric variations of the object to be recognized were absorbed by the SL-HMM structure, and object deformations are represented by the PEM structure.

Figures 4.11(a) and 4.11(b) respectively show the recognition rates for **Dataset1** and **Dataset2**. The tuning parameter τ of the prior distribution with which the highest recognition was obtained was used for each factor. From Figure 4.11, the VB method achieved significantly better recognition rates than the ML method. Comparing the DAEM and EM algorithms, the DAEM algorithm outperformed the EM algorithm for each method. In particular, **FA-VB-DAEM** greatly improved the accuracy of recognition performance. The highest recognition rates for **Dataset1** and **Dataset2** were 95.5% and 89.0% when using **FA-VB-DAEM** with five factors, respectively. Compared with the HMEM (one to five factors) and SL-HMM (zero factor) structures, the HMEMs did not obtain higher recognition rates than the SL-HMMs in the ML method. In contrast, in the VB method, the HMEMs obtained higher recognition rates than the SL-HMMs. Additionally, compared with the FA and PPCA structures, **FA-ML-DAEM/EM** with a high degree of freedom in noise covariance matrix did not achieve higher recognition rates than **PPCA-ML-DAEM/EM** with the ML method. By contrast, **FA-VB-DAEM/EM** achieved higher recognition rates than **PPCA-VB-DAEM/EM**. These results suggest that although HMEMs with FA have a higher potential than those with PPCA and SL-HMM structures, a high generalization ability was not obtained due to over-fitting and the local maximum problems caused by the ML method and EM algorithm. Contrary to this, the proposed VB DAEM algorithm mitigated these problems and achieved a high recognition performance. Therefore, the proposed method is useful for applying image recognition.

Comparing the proposed method with CNN-based approaches, **FA-VB-DAEM** achieved

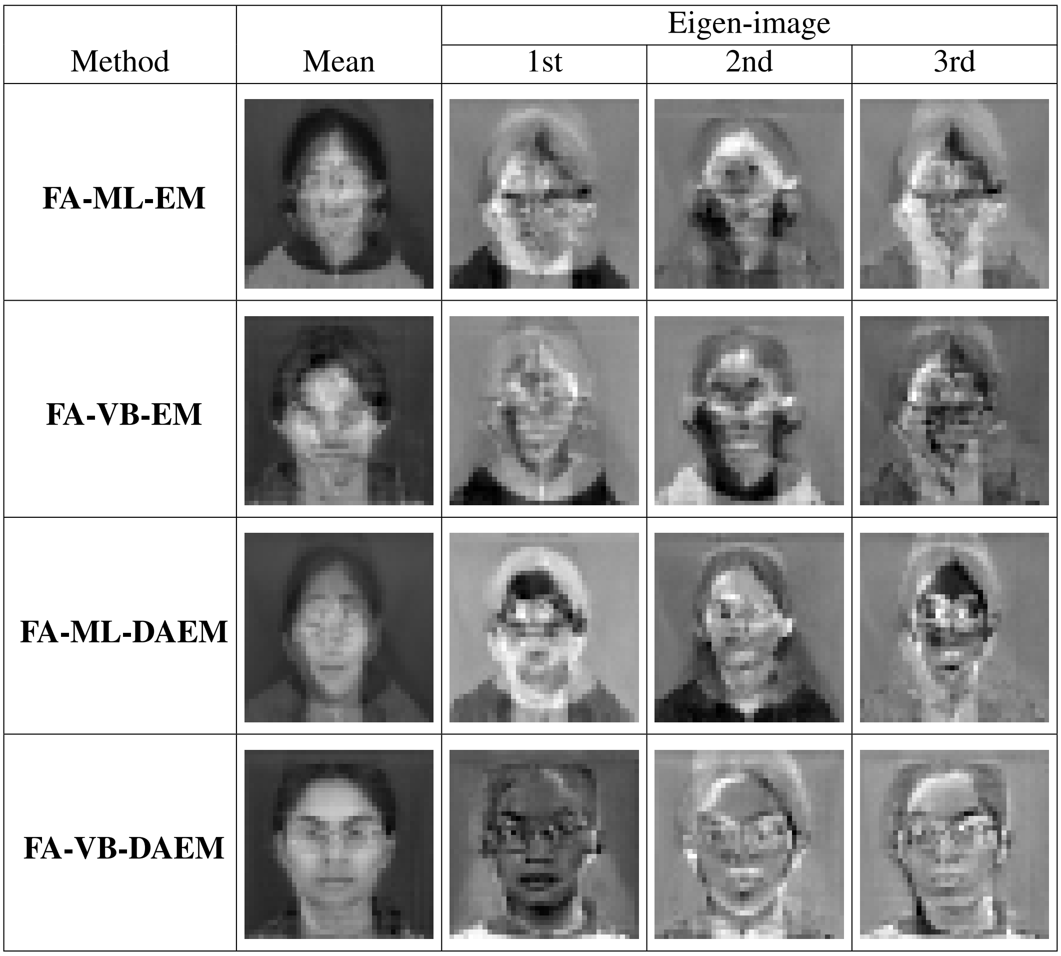
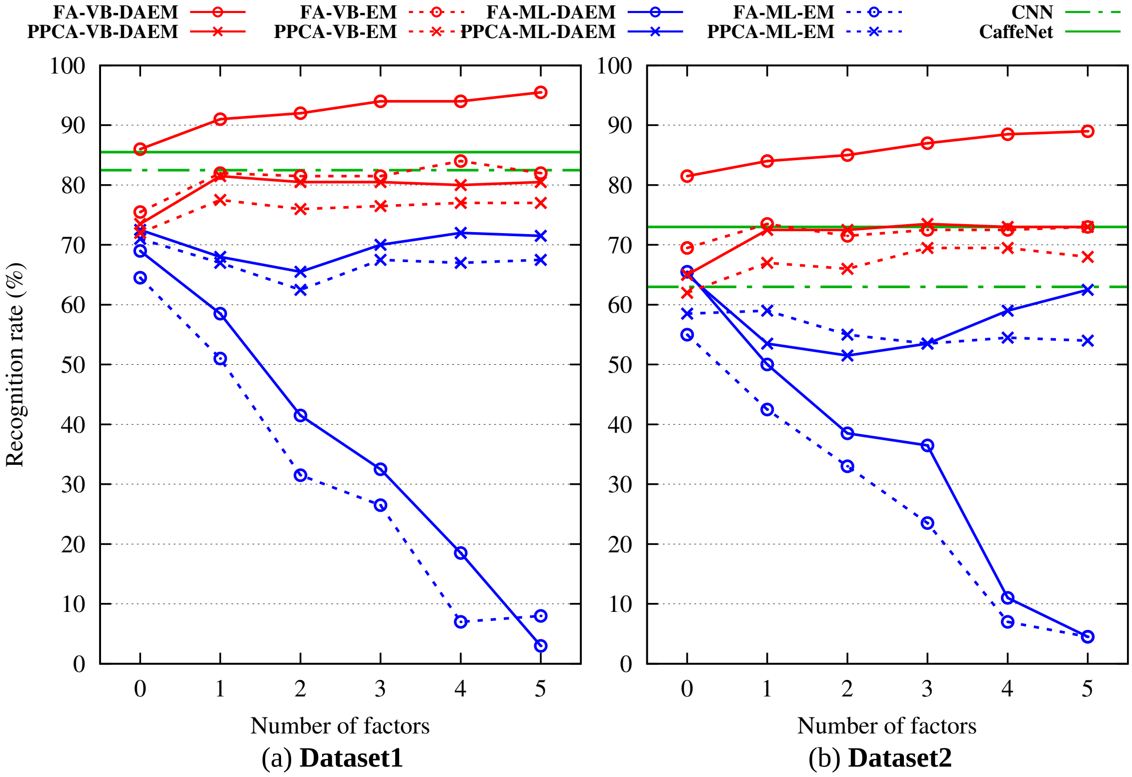


Figure 4.10: Values of mean vector and eigen-images represented by grayscale

better recognition rates than CNN-based approaches. Furthermore, while CNN-based approaches decreased recognition rates significantly for **Dataset2**, which included many size and location variations, **FA-VB-DAEM** maintained high recognition rates. These results suggest that the proposed method is more robust to geometric variations than CNN-based approaches and more effective than CNN-based approaches when the amount of training data is insufficient. However, since **CNN** did not obtain better recognition rates than **CaffeNet**, the number of training images in the experiments was too small to train the **CNN**. Therefore, in the future, I should perform comparative experiments on large datasets.



(a) **Dataset1**

(b) **Dataset2**

Figure 4.11: Recognition rates obtained in image recognition experiments on **Dataset1** and **Dataset2** (circle point: FA, cross point: PPCA, red line: VB, blue line: ML, solid line: DAEM, dotted line: EM).

4.5 Summary

I proposed image recognition based on hidden Markov eigen-image models (HMEMs) using the variational Bayesian (VB) method with the deterministic annealing expectation maximization (DAEM) algorithm. Face recognition experiments were performed on the XM2VTS database. HMEMs based on the VB method demonstrated better recognition performance than the maximum likelihood method. In particular, the VB DAEM algorithm greatly improved the accuracy of recognition performance. This is because the proposed method mitigated the over-fitting and local maximum problems. Additionally, comparative experiment results showed that the proposed method was more robust to geometric variations than convolutional neural networks when the amount of training data is insufficient. Subjects for future work include applying the Bayesian framework to image recognition based on the parameter sharing structures of HMEMs, which share a factor loading matrix across classes, and performing experiments on various image recognition tasks.

Chapter 5

Conclusion

I described a statistical approach to speech synthesis and image recognition based on HMMs. Basic theories and fundamental algorithms of the HMM were reviewed in Chapter 2.

In Chapter 3, the method for constructing TTS systems for languages with unknown pronunciations was proposed. A grapheme-to-phoneme converter and speech synthesizer were constructed from speech recognition results of a proxy language speech recognizer. I applied this method to Japanese and evaluated the naturalness of its output. Experimental results showed that an appropriate phoneme insertion penalty and iteration count for training and recognition were important for the proposed method. The proposed TTS system that does not use language-specific knowledge could synthesize more natural speech compared with that from a grapheme-based TTS system. Future work will include a multilingual speaker-independent speech recognizer based on the IPA [65] or GlobalPhone [66] to obtain accurate phoneme sequences. Furthermore, investigations of prosodic attributes, e.g. accent, stress, and tone, and languages not written with space between words, e.g. Mandarin, Japanese, and Thai, will be needed in order to establish a more language-independent method. Additionally, I will perform experiments on various written languages.

In Chapter 4, the image recognition system based on HMEMs using the variational Bayesian method with DAEM algorithm was proposed. Face recognition experiments were performed on the XM2VTS database. HMEMs based on the VB method demonstrated better recognition performance than the maximum likelihood method. In particular, the VB DAEM algorithm greatly improved the accuracy of recognition performance. This is because the proposed method mitigated the over-fitting and local maximum problems. Additionally, comparative experiment results showed that the proposed method was more robust to geometric variations than convolutional neural networks when the amount of training data is insufficient. Subjects for future work include applying the Bayesian framework to image recognition based on the parameter sharing structures of HMEMs, which share a factor loading matrix across classes, and performing experiments on various image recognition tasks.

Bibliography

[1] Ethnologue. https://www.ethnologue.com.

[2] The Unicode Consortium. The Unicode standard: Worldwide character encoding. *Addison-Wesley Longman*, 1991.

[3] M. Bisani and H. Ney. Joint-sequence models for grapheme-to-phoneme conversion. *Speech Communication*, Vol. 50, No. 5, pp. 434–451, 2008.

[4] F. S. Samaria. Face recognition using hidden Markov models. *Ph. D. dissertation, University of Cambridge*, 1994.

[5] A. V. Nefian and M. H. Hayes Ⅲ. Hidden Markov models for face recognition. *International Conference on Acoustics, Speech and Signal Processing*, Vol. 5, pp. 2721–2724, 1998.

[6] S.-S. Kuo and O. E. Agazzi. Keyword spotting in poorly printed documents using pseudo 2-D hidden Markov models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 16, No. 8, pp. 842–848, 1994.

[7] A. V. Nefian and M. H. Hayes Ⅲ. Maximum likelihood training of the embedded HMM for face detection and recognition. *International Conference on Image Processing*, Vol. 1, pp. 33–36, 2000.

[8] X. Ma, D. Schonfeld, and A. Khokhar. Image segmentation and classification based on a 2D distributed hidden Markov model. *Society of Photo-optical Instrumentation Engineers*, Vol. 6822, , 2008.

[9] J. Li, A. Najmi, and R. M. Gra. Image classification by a two dimensional hidden Markov model. *IEEE Transactions on Signal Processing*, Vol. 48, No. 2, pp. 517–533, 2000.

[10] H. Othman and T. Aboiilnasr. A simplified second-order HMM with application to face recognition. *International Symposium on Circuits and Systems*, Vol. 2, pp. 161–164, 2001.

[11] J.-T. Chien and C.-P. Liao. Maximum confidence hidden Markov modeling for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, No. 4, pp. 606–616, 2008.

[12] D. Kurata, Y. Nankaku, K. Tokuda, T. Kitamura, and Z. Ghahramani. Face recognition based on separable lattice HMMs. *International Conference on Acoustics, Speech and Signal Processing*, Vol. 5, pp. 737–740, 2006.

[13] A. Tamamori, Y. Nankaku, and K. Tokuda. An extension of separable lattice 2-D HMMs for rotational data variations. *IEICE Transactions on Information and Systems*, Vol. E95-D, No. 8, pp. 2074–2083, 2012.

[14] K. Kumaki, Y. Nankaku, and K. Tokuda. Face recognition based on extended separable lattice 2-D HMMs. *International Conference on Acoustics, Speech and Signal Processing*, pp. 2209–2212, 2012.

[15] Y. Takahashi, A. Tamamori, Y. Nankaku, and K. Tokuda. Face recognition based on separable lattice 2-D HMM with state duration modeling. *International Conference on Acoustics, Speech and Signal Processing*, pp. 2162–2165, 2010.

[16] Y. Nankaku and K. Tokuda. Face recognition using hidden Markov eigenface models. *International Conference on Acoustics, Speech and Signal Processing*, Vol. 2, pp. 469–472, 2007.

[17] M. A. Turk and A. Pentland. Face recognition using eigenfaces. *Conference on Computer Vision and Pattern Recognition*, pp. 586–591, 1991.

[18] S. Watanabe and N. Pakvasa. Subspace method of pattern recognition. *International Joint Conference on Pattern Recognition*, pp. 25–32, 1973.

[19] D. B. Rubin and D. T. Thayer. EM algorithms for ML factor analysis. *Psychometrika*, Vol. 47, No. 1, pp. 66–76, 1982.

[20] A. I. Rosti and M. J. F. Gales. Generalised linear Gaussian models. *Cambridge University*, 2001.

[21] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. *In Advances in Neural Information Processing Systems 12*, pp. 449–455, 2000.

[22] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analyzers. *Neural Computation*, Vol. 11, No. 2, pp. 443–482, 1999.

[23] H. Attias. Inferring parameters and structure of latent variable models by variational Bayes. *Conference on Uncertainty in Artificial Intelligence*, pp. 21–30, 1999.

[24] N. Ueda and R. Nakano. Deterministic annealing EM algorithm. *Neural Networks*, Vol. 11, No. 2, pp. 271–282, 1998.

[25] K. Katahira, K. Watanabe, and M. Okada. Deterministic annealing variant of variational Bayes method. *Journal of Physics: Conference Series, 95, 012015*, 2008.

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

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

[28] 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.

[29] 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.

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

[31] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society Series B (Methodological)*, Vol. 39, No. 1, pp. 1–38, 1977.

[32] 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.

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

[34] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura. Speech synthesis based on hidden Markov models. *Proceedings of the IEEE*, Vol. 101, No. 5, pp. 1234–1252, 2013.

[35] H. Zen, A. Senior, and M. Schuster. Statistical parametric speech synthesis using deep neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7962–7966, 2013.

[36] O. Watts, J. Yamagishi, and S. King. Letter-based speech synthesis. *7th ISCA Speech Synthesis Workshop*, pp. 317–322, 2010.

[37] S. Sitaram, A. Parlikar, G. K. Anumanchipalli, and A. W. Black. Universal grapheme-based speech synthesis. *Interspeech 2015*, pp. 3360–3364, 2015.

[38] O. Watts. Unsupervised learning for text-to-speech synthesis. *University of Edinburgh*, 2012.

[39] T. Qian, K. Hollingshead, S. Yoon, K. Kim, and R. Sproat. A python toolkit for universal transliteration. *The seventh international conference on Language Resources and Evaluation*, pp. 2897–2901, 2010.

[40] K. Sawada, S. Takaki, , K. Hashimoto, K. Oura, and K. Tokuda. Overview of NITECH HMM-based text-to-speech system for Blizzard Challenge 2014. *Blizzard Challenge 2014 Workshop*, 2014.

[41] K. Sawada, K. Hashimoto, K. Oura, and K. Tokuda. The NITECH HMM-based text-to-speech system for the Blizzard Challenge 2015. *Blizzard Challenge 2015 Workshop*, 2015.

[42] H. Zen, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Hidden semi-Markov model based speech synthesis. *8th International Conference on Spoken Language Processing*, pp. 1185–1180, 2004.

[43] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura. Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis. *Eurospeech 1999*, pp. 2347–2350, 1999.

[44] 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.

[45] 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.

[46] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura. Speech parameter generation algorithms for HMM-based speech synthesis. *2000 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 936–939, 2000.

[47] 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.

[48] CMU pronouncing dictionary. http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

[49] D. B. Paul and J. M. Baker. The design for the wall street journal-based CSR corpus. *The workshop on Speech and Natural Language*, pp. 357–362, 1992.

[50] J. Garofolo, L. Lamel, W. Fisher, J. Fiscus, D. Pallett, N. Dahlgren, and V. Zue. TIMIT: acoustic-phonetic continuous speech corpus. *Linguistic Data Consortium*, 1993.

[51] HTK. http://htk.eng.cam.ac.uk/.

[52] K. Vertanen. Baseline WSJ acoustic models for HTK and Sphinx: Training recipes and recognition experiments. *Cavendish Laboratory*, 2006.

[53] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi. Multi-space probability distribution HMM. *IEICE Transactions on Information & Systems*, Vol. E85-D, No. 3, pp. 455–464, 2002.

[54] Sequitur G2P. http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html.

[55] H. Kawahara, I. Masuda-Katsuse, and A. Cheveigne. 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.

[56] K. Shinoda and T. Watanabe. MDL-based context-dependent subword modeling for speech recognition. *Acoustical Science and Technology*, Vol. 21, No. 2, pp. 76–86, 2000.

[57] T. Toda and K. Tokuda. Speech parameter generation algorithm considering global variance for HMM-based speech synthesis. *Interspeech 2005*, pp. 2801–2804, 2005.

[58] HTS. http://hts.sp.nitech.ac.jp/.

[59] T. Toda, A. W. Black, and K. Tokuda. Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory. *IEEE Transactions on Audio, Speech, and Language Processing*, Vol. 15, No. 8, pp. 2222–2235, 2007.

[60] K. Ito, M. Yamamoto, K. Takeda, T. Takezawa, T. Matsuoka, T. Kobayashi, K. Shikano, and S. Itahashi. JNAS: Japanese speech corpus for large vocabulary continuous speech recognition research. *The Journal of the Acoustic Society of Japan*, Vol. 20, No. 3, pp. 199–206, 1999.

[61] Technical Standardization Committee on Speech Input/Output Systems. Speech synthesis system performance evaluation methods. *Japan Electronic Industry Development Association, IT-4001*, 2003. (in Japanese).

[62] A. W. Black and K. Tokuda. The Blizzard Challenge – 2005: Evaluating corpus-based speech synthesis on common datasets. *Interspeech 2005*, pp. 77–80, 2005.

[63] Blizzard Challenge 2015. http://www.synsig.org/index.php/Blizzard_Challenge_2015.

[64] Festvox. http://festvox.org/bsv/x3528.html.

[65] International Phonetic Association. Handbook of the international phonetic association: A guide to the use of the international phonetic alphabet. *Cambridge University Press*, 1999.

[66] T. Schultz. GlobalPhone: A multilingual speech and text database developed at karlsruhe university. *7th International Conference of Spoken Language Processing*, 2002.

[67] S. A. Sirohey. Human face segmentation and identification. *Technical Report CAR-TR-695, Center for Automation Research, University of Maryland*, 1993.

[68] T. Wakahara C. C. Tappert, C. Y. Suen. The state of the art in online handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 8, pp. 787–808, 1990.

[69] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, Vol. 60, No. 2, pp. 91–110, 2004.

[70] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. *ECCV International Workshop on Statistical Learning in Computer Vision*, 2004.

[71] L. Wiskott, J.-M. Fellous, N. Kruger, and C. Malsburg. Face recognition by elastic bunch graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 7, pp. 775–779, 1997.

[72] C. Schmid H. Jegou, M. Douze. Hamming embedding and weak geometric consistency for large scale image search. *European Conference on Computer Vision*, 2008.

[73] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[74] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Conference on Neural Information Processing Systems*, pp. 1097–1105, 2012.

[75] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, Vol. 37, No. 2, pp. 183–233, 1999.

[76] B. P. Carlin and S. Chib. Bayesian model choice via Markov chain Monte Carlo. *Journal of the Royal Statistical Society*, Vol. 57, No. 3, pp. 473–484, 1995.

[77] K. Sawada, A. Tamamori, K. Hashimoto, Y. Nankaku, and K. Tokuda. A Bayesian approach to image recognition based on separable lattice hidden Markov models. *IEICE Transactions on Information and Systems*, Vol. E99-D, No. 12, pp. 3119–3131, 2016.

[78] K. Messer, J. Matas, J. Kittler, and K. Jonsson. XM2VTSDB: The extended M2VTS database. *International Conference on Audio and Video-based Biometric Person Authentication*, pp. 72–77, 1999.

[79] D. A. Reynolds. Comparison of background normalization methods for text-independent speaker verification. *European Conference on Speech Communication and Technology*, Vol. 2, pp. 963–966, 1997.

[80] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *ACM international conference on Multimedia*, pp. 675–678, 2014.

[81] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, Vol. 115, No. 3, pp. 211–252, 2015.

[82] L. R. Rabiner and B. H. Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pp. 4–16, 1986.

List of Publications

Journal papers

[1] **Kei Sawada**, Akira Tamamori, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, “A Bayesian approach to image recognition based on separable lattice hidden Markov models,” IEICE Transactions on Information and Systems, Vol. E99-D, No. 12, pp. 3119–3131, Dec. 2016.

[2] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, “Constructing text-to-speech systems for languages with unknown pronunciations,” Acoustical Science and Technology. (Accepted)

[3] **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, “A Bayesian framework for image recognition based on hidden Markov eigen-image models,” IEEJ Transactions on Electrical and Electronic Engineering. (Conditionally accepted)

International conference proceedings

[4] **Kei Sawada**, Akira Tamamori, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, “Face recognition based on separable lattice 2-D HMMs using variational bayesian method,” 2012 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2012), pp. 2205–2208, Mar. 2012.

[5] Shinji Takaki, **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda,

"Overview of NIT HMM-based speech synthesis system for Blizzard Challenge 2012," Blizzard Challenge 2012 Workshop, Sep. 2012.

[**6**] Shinji Takaki, **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda, "Overview of NITECH HMM-based speech synthesis system for Blizzard Challenge 2013," Blizzard Challenge 2013 Workshop, Sep. 2013.

[**7**] **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Image recognition based on hidden Markov eigen-image models using variational Bayesian method," Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC 2013), pp. 1–8, Oct. 2013.

[**8**] **Kei Sawada**, Shinji Takaki, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda, "Overview of NITECH HMM-based text-to-speech system for Blizzard Challenge 2014," Blizzard Challenge 2014 Workshop, Sep. 2014.

[**9**] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda, "The NITECH HMM-based text-to-speech system for the Blizzard Challenge 2015," Blizzard Challenge 2015 Workshop, Sep. 2015.

[**10**] **Kei Sawada**, Chiaki Asai, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda, "The NITech text-to-speech system for the Blizzard Challenge 2016," Blizzard Challenge 2016 Workshop, Sep. 2016.

[**11**] Yoshinari Tsuzuki, **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Image recognition based on discriminative models using features generated from separable lattice HMMs," 2017 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2017), pp. 2607–2611, Mar. 2017.

[**12**] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, and Keiichi Tokuda, "The NITech text-to-speech system for the Blizzard Challenge 2017," Blizzard Challenge 2017 Workshop, Aug. 2017.

[13] **Kei Sawada**, Keiichi Tokuda, Simon King, and Alan W Black, "The Blizzard Machine Learning Challenge 2017," 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU 2017), pp. 331–337, Dec. 2017.

Technical reports

[14] **Kei Sawada**, Akira Tamamori, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Face recognition based on separable lattice 2-D HMMs with variational Bayesian method," IEICE Technical Report, vol. 111, no. 317, PRMU2011-120, pp. 125–130, Nov. 2011.

[15] **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Image recognition based on hidden Markov eigen-image models with the variational Bayesian method," IEICE Technical Report, vol. 112, no. 441, PRMU2012-165, pp. 155–160, Feb. 2013.

[16] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Evaluation of text-to-speech system construction for unknown-pronunciation languages," IEICE Technical Report, vol. 115, no. 346, SP2015-80, pp. 93–98, Dec. 2015.

[17] Masato Sukegawa, **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Parameter sharing structures of separable lattice HMMs using mixture output distributions for image recognition," IEICE Technical Report, vol. 115, no. 456, PRMU2015-138, pp. 37–42, Feb. 2016.

[18] Yoshinari Tsuzuki, **Kei Sawada**, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "Image recognition based on discriminative models using features generated from separable lattice HMMs," IEICE Technical Report, vol. 116, no. 89, PRMU2016-36, pp. 7–12, Jun. 2016.

[19] Chiaki Asai, **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Designing linguistic features for expressive speech synthesis

using audiobooks," IEICE Technical Report, vol. 116, no. 414, SP2016-70, pp. 35–40, Jan. 2017.

Domestic conference proceedings

[20] **Kei Sawada**, Akira Tamamori, Kei Hashimoto, Yoshihiko Nankaku, and Keiichi Tokuda, "A training algorithm based on variational Bayesian method using deterministic annealing process for separable lattice 2-D HMMs," The 74th National Convention of IPSJ, vol. 2, pp. 409–410, Mar. 2012.

[21] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Investigation of text-to-speech system construction in unknown-pronunciation language," Acoustical Society of Japan 2015 Autumn Meeting, pp. 231–232, Sep. 2015.

[22] **Kei Sawada**, Kazuki Igami, Chiaki Asai, Yusuke Sato, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Automatic construction of training corpus using audiobooks for statistical parametric speech synthesis," Acoustical Society of Japan 2016 Spring Meeting, pp. 219–220, Mar. 2016.

[23] Yukiya Hono, **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, Keiichi Tokuda, Daisuke Kondo, and Daisuke Ishikawa, "Singing voice conversion using post data in music SNS," Acoustical Society of Japan 2017 Autumn Meeting, pp. 209–210, Sep. 2017.

[24] **Kei Sawada**, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda, "Overview of the NITech text-to-speech system for the Blizzard Challenge 2017," Acoustical Society of Japan 2017 Autumn Meeting, pp. 287–290, Sep. 2017.

[25] **Kei Sawada**, Keiichi Tokuda, Simon King, and Alan W Black, "Overview of the Blizzard Machine Learning Challenge 2017," Acoustical Society of Japan 2017 Spring Meeting, Mar. 2017.

Appendix A

Derivation of training algorithm for HMEMs

A.1 Derivation of VB EM algorithm for HMEMs

The VB posterior distribution $Q(\boldsymbol{x})$ in Eq. (4.25) can be written by a Gaussian distribution as follows:

$$Q(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}\,|\,\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})}, \hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})}), \tag{A.1}$$

where $\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})}$ and $\hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})}$ are the mean vector and full covariance matrix of the factor vector $\boldsymbol{x}$, respectively. The re-estimation formulae $\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})}$ and $\hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})}$ of the VB posterior distribution $Q(\boldsymbol{x})$ are derived as follows:

$$\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} = \hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})}\left\{\sum_{\boldsymbol{t}}\sum_{\boldsymbol{k}}\sum_{d=1}^{D}\langle z_{\boldsymbol{t},\boldsymbol{k}}\rangle_{Q(\boldsymbol{z})}\left[\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d}\hat{\eta}_{\boldsymbol{k}}\hat{\nu}_{\boldsymbol{k},d}^{-1}\left(o_{\boldsymbol{t},d}-\hat{\gamma}_{\boldsymbol{k},d}\right)-\hat{\boldsymbol{u}}_{\boldsymbol{k}}\right]\right\}, \tag{A.2}$$

$$\hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})} = \left[\boldsymbol{I}+\sum_{\boldsymbol{k}}\sum_{d=1}^{D}N_{\boldsymbol{k}}\left(\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d}\hat{\eta}_{\boldsymbol{k}}\hat{\nu}_{\boldsymbol{k},d}^{-1}\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d}^{\top}+\hat{\boldsymbol{\Upsilon}}_{\boldsymbol{k}}\right)\right]^{-1}, \tag{A.3}$$

where $o_{\boldsymbol{t},d}$ is $\boldsymbol{o}_{\boldsymbol{t}} = \{o_{\boldsymbol{t},d}\,|\,d=1,2,\ldots,D\}$ and $\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d}$, $\hat{\gamma}_{\boldsymbol{k},d}$, $\hat{\boldsymbol{\Upsilon}}_{\boldsymbol{k}}$, and $\hat{\boldsymbol{u}}_{\boldsymbol{k}}$ are defined as:

$$\hat{\boldsymbol{h}}_{\boldsymbol{k},d} = \left[\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d}^{\top}\ \hat{\gamma}_{\boldsymbol{k},d}\right]^{\top}, \tag{A.4}$$

$$\hat{\boldsymbol{U}}_{\boldsymbol{k}}^{-1} = \begin{bmatrix}\hat{\boldsymbol{\Upsilon}}_{\boldsymbol{k}} & \hat{\boldsymbol{u}}_{\boldsymbol{k}}\\ \hat{\boldsymbol{u}}_{\boldsymbol{k}}^{\top} & \hat{u}_{\boldsymbol{k}}\end{bmatrix}. \tag{A.5}$$

The expectation value with respect to $Q(\boldsymbol{z}^{(m)})$ is computed by the following equations:

$$N_{\boldsymbol{k}} = \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})}, \tag{A.6}$$

$$\langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} = \langle z^{(1)}_{t^{(1)},k^{(1)}} \rangle_{Q(\boldsymbol{z}^{(1)})} \langle z^{(2)}_{t^{(2)},k^{(2)}} \rangle_{Q(\boldsymbol{z}^{(2)})}, \tag{A.7}$$

$$\langle z^{(m)}_{t^{(m)},k^{(m)}} \rangle_{Q(\boldsymbol{z}^{(m)})} = \sum_{\boldsymbol{z}^{(m)}} Q(\boldsymbol{z}^{(m)}) z^{(m)}_{t^{(m)},k^{(m)}}, \tag{A.8}$$

$$N^{(m)}_{k^{(m)},\bar{k}^{(m)}} = \sum_{t^{(m)}=2}^{T^{(m)}} \langle z^{(m)}_{t^{(m)}-1,k^{(m)}} z^{(m)}_{t^{(m)},\bar{k}^{(m)}} \rangle_{Q(\boldsymbol{z}^{(m)})}, \tag{A.9}$$

$$\langle z^{(m)}_{t^{(m)}-1,k^{(m)}} z^{(m)}_{t^{(m)},\bar{k}^{(m)}} \rangle_{Q(\boldsymbol{z}^{(m)})} = \sum_{\boldsymbol{z}^{(m)}} Q(\boldsymbol{z}^{(m)}) z^{(m)}_{t^{(m)}-1,k^{(m)}} z^{(m)}_{t^{(m)},\bar{k}^{(m)}}, \tag{A.10}$$

where $\langle \cdot \rangle_{Q(\cdot)}$ denotes the expectation with respect to the posterior distribution $Q(\cdot)$ and $z^{(m)}_{t^{(m)},k^{(m)}}$ is the Kronecker delta function:

$$z^{(m)}_{t^{(m)},k^{(m)}} = \begin{cases} 0 & (z^{(m)}_{t^{(m)}} \neq k^{(m)}) \\ 1 & (z^{(m)}_{t^{(m)}} = k^{(m)}) \end{cases}. \tag{A.11}$$

The VB posterior distribution $Q(\boldsymbol{z}^{(m)})$ in Eq. (4.26) can be represented as follows:

$$\begin{aligned} Q(\boldsymbol{z}^{(m)}) \propto \exp & \left[\sum_{k^{(m)}=1}^{K^{(m)}} z^{(m)}_{1,k^{(m)}} \langle \ln \pi^{(m)}_{k^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} \right] \\ \times \exp & \left[\sum_{t^{(m)}=2}^{T^{(m)}} \sum_{k^{(m)}=1}^{K^{(m)}} \sum_{\bar{k}^{(m)}=1}^{K^{(m)}} z^{(m)}_{t^{(m)}-1,k^{(m)}} z^{(m)}_{t^{(m)},\bar{k}^{(m)}} \langle \ln a^{(m)}_{k^{(m)},\bar{k}^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} \right] \\ \times \exp & \left[\sum_{\boldsymbol{t}} \sum_{k^{(m)}=1}^{K^{(m)}} \sum_{k^{(\bar{m})}=1}^{K^{(\bar{m})}} z^{(m)}_{t^{(m)},k^{(m)}} \langle z^{(\bar{m})}_{t^{(\bar{m})},k^{(\bar{m})}} \rangle_{Q(\boldsymbol{z}^{(\bar{m})})} \langle \ln \boldsymbol{b}_{\boldsymbol{k}}(\boldsymbol{o}_{\boldsymbol{t}}) \rangle_{Q(\boldsymbol{x})Q(\boldsymbol{\Lambda})} \right]. \end{aligned} \tag{A.12}$$

The expectation value with respect to $Q(\boldsymbol{x})$ and $Q(\boldsymbol{\Lambda})$ are derived as:

$$\langle \log \pi^{(m)}_{k^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} = \mathit{\Psi}\left(\hat{\phi}^{(m)}_{k^{(m)}}\right) - \mathit{\Psi}\left(\sum_{k'^{(m)}=1}^{K^{(m)}} \hat{\phi}^{(m)}_{k'^{(m)}}\right), \tag{A.13}$$

$$\langle \log a^{(m)}_{k^{(m)},\bar{k}^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} = \mathit{\Psi}\left(\hat{\alpha}^{(m)}_{k^{(m)},\bar{k}^{(m)}}\right) - \mathit{\Psi}\left(\sum_{k'^{(m)}=1}^{K^{(m)}} \hat{\alpha}^{(m)}_{k^{(m)}k'^{(m)}}\right), \tag{A.14}$$

$$\begin{aligned}\langle \ln \boldsymbol{b}_{\boldsymbol{k}}(\boldsymbol{o}_{\boldsymbol{t}}) \rangle_{Q(\boldsymbol{x})Q(\boldsymbol{\Lambda})} = \sum_{d=1}^{D} \Bigg[& \ln \mathcal{N}(o_{\boldsymbol{t},d} \,|\, \hat{\boldsymbol{h}}^{\top}_{\boldsymbol{k},d} \langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})}, \hat{\eta}^{-1}_{\boldsymbol{k},d} \hat{\nu}_{\boldsymbol{k},d}) \\ & -\frac{1}{2} \ln \hat{\eta}_{\boldsymbol{k},d} + \frac{1}{2} \mathit{\Psi}(\hat{\eta}_{\boldsymbol{k},d}) - \frac{1}{2} \mathrm{Tr} \Big\{ \hat{\boldsymbol{h}}^{\top}_{\boldsymbol{k},d} \left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\top} \right\rangle_{Q(\boldsymbol{x})} \hat{\boldsymbol{h}}_{\boldsymbol{k},d} \\ & -\hat{\boldsymbol{h}}^{\top}_{\boldsymbol{k},d} \langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})} \langle \tilde{\boldsymbol{x}} \rangle^{\top}_{Q(\boldsymbol{x})} \hat{\boldsymbol{h}}_{\boldsymbol{k},d} + \hat{\boldsymbol{U}}^{-1}_{\boldsymbol{k}} \left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\top} \right\rangle_{Q(\boldsymbol{x})} \Big\} \Bigg], \end{aligned} \tag{A.15}$$

where $\mathit{\Psi}(\cdot)$ is a digamma function. The expectation values $\langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})}$ and $\left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\top} \right\rangle_{Q(\boldsymbol{x})}$ with respect to $Q(\boldsymbol{x})$ can be calculated by using Eqs. (A.2) and (A.3):

$$\langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})} = \left[\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})\top} \;\; 1 \right]^{\top}, \tag{A.16}$$

$$\left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\top} \right\rangle_{Q(\boldsymbol{x})} = \begin{bmatrix} \hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})} + \hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} \hat{\boldsymbol{\mu}}^{(\boldsymbol{x})\top} & \hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} \\ \hat{\boldsymbol{\mu}}^{(\boldsymbol{x})\top} & 1 \end{bmatrix}. \tag{A.17}$$

Since HMEMs assume independence of horizontal and vertical state sequences, the VB posterior distribution $Q(\boldsymbol{z}^{(m)})$ in Eq. (A.12) has a Markovian structure as the likelihood function of an standard one-dimensional HMM. Therefore, Eqs. (A.8) and (A.10) can be computed efficiently by the forward-backward algorithm [82] in section 2.2.

The VB posterior distribution $Q(\boldsymbol{\Lambda})$ in Eq. (4.27) can be written by Dirichlet and Gauss-gamma distributions:

$$\begin{aligned} Q(\boldsymbol{\Lambda}) = \prod_{m=1}^{2} & \left[\mathcal{D}(\boldsymbol{\pi}^{(m)} \,|\, \hat{\boldsymbol{\phi}}^{(m)}) \prod_{k^{(m)}=1}^{K^{(m)}} \mathcal{D}(\boldsymbol{a}^{(m)}_{k^{(m)}} \,|\, \hat{\boldsymbol{\alpha}}^{(m)}_{k^{(m)}}) \right] \\ & \times \prod_{\boldsymbol{k}} \prod_{d=1}^{D} \mathcal{N}(\tilde{\boldsymbol{w}}_{\boldsymbol{k},d} \,|\, \hat{\boldsymbol{h}}_{\boldsymbol{k},d}, \hat{\boldsymbol{U}}^{-1}_{\boldsymbol{k}} \sigma^2_{\boldsymbol{k},d}) \mathcal{G}((\sigma^2_{\boldsymbol{k},d})^{-1} \,|\, \hat{\eta}_{\boldsymbol{k}}, \hat{\nu}_{\boldsymbol{k},d}). \end{aligned} \tag{A.18}$$

The posterior distribution of model parameters can be updated by statistics of the training data as follows:

$$\begin{aligned}
\hat{\phi}^{(m)}_{k^{(m)}} &= \phi^{(m)}_{k^{(m)}} + \langle z^{(m)}_{1,k^{(m)}} \rangle_{Q(\boldsymbol{z}^{(m)})}, && \text{(A.19)}\\
\hat{\alpha}^{(m)}_{k^{(m)},\bar{k}^{(m)}} &= \alpha^{(m)}_{k^{(m)},\bar{k}^{(m)}} + N^{(m)}_{k^{(m)},\bar{k}^{(m)}}, && \text{(A.20)}\\
\hat{\boldsymbol{h}}_{\boldsymbol{k},d} &= \hat{\boldsymbol{U}}_{\boldsymbol{k}}^{-1}\left(\boldsymbol{U}_{\boldsymbol{k}}\boldsymbol{h}_{\boldsymbol{k},d} + \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} o_{\boldsymbol{t},d} \langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})}\right), && \text{(A.21)}\\
\hat{\boldsymbol{U}}_{\boldsymbol{k}} &= \boldsymbol{U}_{\boldsymbol{k}} + N_{\boldsymbol{k}} \left\langle \tilde{\boldsymbol{x}}\tilde{\boldsymbol{x}}^{\top} \right\rangle_{Q(\boldsymbol{x})}, && \text{(A.22)}\\
\hat{\eta}_{\boldsymbol{k}} &= \eta_{\boldsymbol{k}} + \frac{1}{2}N_{\boldsymbol{k}}, && \text{(A.23)}\\
\hat{\nu}_{\boldsymbol{k},d} &= \nu_{\boldsymbol{k},d} + \frac{1}{2}\boldsymbol{h}^{\top}_{\boldsymbol{k},d}\boldsymbol{U}_{\boldsymbol{k}}\boldsymbol{h}_{\boldsymbol{k},d} - \frac{1}{2}\hat{\boldsymbol{h}}^{\top}_{\boldsymbol{k},d}\hat{\boldsymbol{U}}_{\boldsymbol{k}}\hat{\boldsymbol{h}}_{\boldsymbol{k},d} + \frac{1}{2}\sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} o_{\boldsymbol{t},d} o_{\boldsymbol{t},d}. && \text{(A.24)}
\end{aligned}$$

When PPCA structure, i.e., the covariance matrix of noise vector is tied in feature dimensions and HMM states, is used, the re-estimation formula of the gamma distribution is as follows:

$$\begin{aligned}
\hat{\eta} &= \eta + \frac{1}{2}D\sum_{\boldsymbol{k}} N_{\boldsymbol{k}}, && \text{(A.25)}\\
\hat{\nu} &= \nu + \frac{1}{2}\sum_{\boldsymbol{k}}\sum_{d=1}^{D}\boldsymbol{h}^{\top}_{\boldsymbol{k},d}\boldsymbol{U}_{\boldsymbol{k}}\boldsymbol{h}_{\boldsymbol{k},d} - \frac{1}{2}\sum_{\boldsymbol{k}}\sum_{d=1}^{D}\hat{\boldsymbol{h}}^{\top}_{\boldsymbol{k},d}\hat{\boldsymbol{U}}_{\boldsymbol{k}}\hat{\boldsymbol{h}}_{\boldsymbol{k},d}\\
&\quad + \frac{1}{2}\sum_{\boldsymbol{t}}\sum_{\boldsymbol{k}}\sum_{d=1}^{D} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} o_{\boldsymbol{t},d} o_{\boldsymbol{t},d}. && \text{(A.26)}
\end{aligned}$$

A.2 Derivation of Hyper-parameters based on prior data

The hyper-parameters based on prior data $\boldsymbol{o}^{(\text{prior})}$ are given as:

$$
\begin{aligned}
\phi^{(m)}_{k^{(m)}} &= 1 + \tau \langle z^{(m)}_{1,k^{(m)}} \rangle^{(\text{prior})}_{Q(\boldsymbol{z}^{(m)})}, && \text{(A.27)}\\
\alpha^{(m)}_{k^{(m)},\bar{k}^{(m)}} &= 1 + \tau N^{(m)(\text{prior})}_{k^{(m)},\bar{k}^{(m)}}, && \text{(A.28)}\\
\boldsymbol{h}_{\boldsymbol{k},d} &= \boldsymbol{U}_{\boldsymbol{k}}^{-1} \tau \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle^{(\text{prior})}_{Q(\boldsymbol{z})} \, o^{(\text{prior})}_{\boldsymbol{t},d} \, \langle \tilde{\boldsymbol{x}} \rangle^{(\text{prior})}_{Q(\boldsymbol{x})}, && \text{(A.29)}\\
\boldsymbol{U}_{\boldsymbol{k}} &= \tau N^{(\text{prior})}_{\boldsymbol{k}} \left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^{\top} \right\rangle^{(\text{prior})}_{Q(\boldsymbol{x})}, && \text{(A.30)}\\
\eta_{\boldsymbol{k}} &= 1 + \frac{1}{2} \tau N^{(\text{prior})}_{\boldsymbol{k}}, && \text{(A.31)}\\
\nu_{\boldsymbol{k},d} &= \frac{1}{2} \tau \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle^{(\text{prior})}_{Q(\boldsymbol{z})} \, o^{(\text{prior})}_{\boldsymbol{t},d} o^{(\text{prior})}_{\boldsymbol{t},d} - \frac{1}{2} \boldsymbol{h}^{\top}_{\boldsymbol{k},d} \boldsymbol{U}_{\boldsymbol{k}} \boldsymbol{h}_{\boldsymbol{k},d}, && \text{(A.32)}
\end{aligned}
$$

where $\cdot^{(\text{prior})}$ denotes statistics of prior data $\boldsymbol{o}^{(\text{prior})}$ and τ is the tuning parameter for prior distributions $P(\boldsymbol{\Lambda})$.

A.3 Derivation of VB DAEM algorithm for HMEMs

In the VB DAEM algorithm, the temperature parameters β_l are attached to the re-estimation formulas. The VB posterior distribution $Q(\boldsymbol{x})$ in Eq. (4.38) can be updated as follows:

$$
\begin{aligned}
\hat{\boldsymbol{\mu}}^{(\boldsymbol{x})} &= \hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})} \left\{ \beta_1 \sum_{\boldsymbol{t}} \sum_{\boldsymbol{k}} \sum_{d=1}^{D} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} \left[\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d} \hat{\eta}_{\boldsymbol{k}} \hat{\nu}^{-1}_{\boldsymbol{k},d} (o_{\boldsymbol{t},d} - \hat{\gamma}_{\boldsymbol{k},d}) - \hat{\boldsymbol{u}}_{\boldsymbol{k}} \right] \right\}, && \text{(A.33)}\\
\hat{\boldsymbol{\Sigma}}^{(\boldsymbol{x})} &= \left[\beta_2 \boldsymbol{I} + \beta_1 \sum_{\boldsymbol{k}} \sum_{d=1}^{D} N_{\boldsymbol{k}} (\hat{\boldsymbol{\omega}}_{\boldsymbol{k},d} \hat{\eta}_{\boldsymbol{k}} \hat{\nu}^{-1}_{\boldsymbol{k},d} \hat{\boldsymbol{\omega}}^{\top}_{\boldsymbol{k},d} + \hat{\boldsymbol{\Upsilon}}_{\boldsymbol{k}}) \right]^{-1}. && \text{(A.34)}
\end{aligned}
$$

The VB posterior distribution $Q(\boldsymbol{z}^{(m)})$ in Eq. (4.39) can be represented as follows:

$$
\begin{aligned}
Q(\boldsymbol{z}^{(m)}) \propto \exp & \left[\beta_3 \sum_{k^{(m)}=1}^{K^{(m)}} z^{(m)}_{1,k^{(m)}} \langle \ln \pi^{(m)}_{k^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} \right] \\
\times \exp & \left[\beta_3 \sum_{t^{(m)}=2}^{T^{(m)}} \sum_{k^{(m)}=1}^{K^{(m)}} \sum_{\bar{k}^{(m)}=1}^{K^{(m)}} z^{(m)}_{t^{(m)}-1,k^{(m)}} z^{(m)}_{t^{(m)},\bar{k}^{(m)}} \langle \ln a^{(m)}_{k^{(m)},\bar{k}^{(m)}} \rangle_{Q(\boldsymbol{\Lambda})} \right] \\
\times \exp & \left[\beta_1 \sum_{\boldsymbol{t}} \sum_{k^{(m)}=1}^{K^{(m)}} \sum_{k^{(\bar{m})}=1}^{K^{(\bar{m})}} z^{(m)}_{t^{(m)},k^{(m)}} \langle z^{(\bar{m})}_{t^{(\bar{m})},k^{(\bar{m})}} \rangle_{Q(\boldsymbol{z}^{(\bar{m})})} \langle \ln \boldsymbol{b}_{\boldsymbol{k}}(\boldsymbol{o}_{\boldsymbol{t}}) \rangle_{Q(\boldsymbol{x})Q(\boldsymbol{\Lambda})} \right]. \quad \text{(A.35)}
\end{aligned}
$$

The VB posterior distribution $Q(\boldsymbol{z}^{(m)})$ can be applied to the forward-backward algorithm. Therefore, Eqs. (A.8) and (A.10) considering the temperature parameters can be obtained efficiently.

The VB posterior distribution $Q(\boldsymbol{\Lambda})$ in Eq. (4.40) can be updated as follows:

$$
\begin{aligned}
\hat{\phi}^{(m)}_{k^{(m)}} &= \beta_4(\phi^{(m)}_{k^{(m)}} - 1) + 1 + \beta_3 \langle z^{(m)}_{1,k^{(m)}} \rangle_{Q(\boldsymbol{z}^{(m)})}, && \text{(A.36)} \\
\hat{\alpha}^{(m)}_{k^{(m)},\bar{k}^{(m)}} &= \beta_4(\alpha^{(m)}_{k^{(m)},\bar{k}^{(m)}} - 1) + 1 + \beta_3 N^{(m)}_{k^{(m)},\bar{k}^{(m)}}, && \text{(A.37)} \\
\hat{\boldsymbol{h}}_{\boldsymbol{k},d} &= \hat{\boldsymbol{U}}_{\boldsymbol{k}}^{-1} \left(\beta_4 \boldsymbol{U}_{\boldsymbol{k}} \boldsymbol{h}_{\boldsymbol{k},d} + \beta_1 \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} o_{\boldsymbol{t},d} \langle \tilde{\boldsymbol{x}} \rangle_{Q(\boldsymbol{x})} \right), && \text{(A.38)} \\
\hat{\boldsymbol{U}}_{\boldsymbol{k}} &= \beta_4 \boldsymbol{U}_{\boldsymbol{k}} + \beta_1 N_{\boldsymbol{k}} \left\langle \tilde{\boldsymbol{x}} \tilde{\boldsymbol{x}}^\top \right\rangle_{Q(\boldsymbol{x})}, && \text{(A.39)} \\
\hat{\eta}_{\boldsymbol{k}} &= \beta_4(\eta_{\boldsymbol{k}} - 1) + 1 + \frac{1}{2} \beta_1 N_{\boldsymbol{k}}, && \text{(A.40)} \\
\hat{\nu}_{\boldsymbol{k},d} &= \beta_4 \nu_{\boldsymbol{k},d} + \frac{1}{2} \beta_4 \boldsymbol{h}^\top_{\boldsymbol{k},d} \boldsymbol{U}_{\boldsymbol{k}} \boldsymbol{h}_{\boldsymbol{k},d} - \frac{1}{2} \hat{\boldsymbol{h}}^\top_{\boldsymbol{k},d} \hat{\boldsymbol{U}}_{\boldsymbol{k}} \hat{\boldsymbol{h}}_{\boldsymbol{k},d} \\
&\quad + \frac{1}{2} \beta_1 \sum_{\boldsymbol{t}} \langle z_{\boldsymbol{t},\boldsymbol{k}} \rangle_{Q(\boldsymbol{z})} o_{\boldsymbol{t},d} o_{\boldsymbol{t},d}. && \text{(A.41)}
\end{aligned}
$$