2 Learning Algorithms

This chapter addresses the background knowledge about machine learning and deep learning, which is used as a foundation for statistical speaker recognition. The paradigms based on statistical inference and neural networks are presented. In particular, we address the basics of expectation-maximization (EM) algorithms, variational EM algorithms, and general approximate inference algorithms. Bayesian learning is presented to deal with regularization in model-based speaker recognition.

2.1 Fundamentals of Statistical Learning

In general, the speech signal in speaker recognition is collected as a set of observation vectors. The underlying factors including channels, noises, and environments, etc., are unknown and treated as latent variables when training the speaker recognition model. Hierarchical latent variables or features can be explored to characterize the structural information in latent variable or feature space. In machine learning community, the latent space models are constructed in different ways with different interpretations. Machine learning provides a wide range of model-based approaches for speaker recognition. A model-based approach aims to incorporate the physical phenomena, measurements, uncertainties, and noises in the form of mathematical models. Such an approach is developed in a unified manner through different algorithms, examples, applications, and case studies. Mainstream methods are based on the statistical models. There are two popular categories of statistical methods in modern machine learning literature. One is the probabilistic models and the other is the neural network models that have been developing widely in various speaker recognition systems. A brief introduction to these two broad categories is addressed in what follows.

2.1.1 Probabilistic Models

The first category of statistical learning methods is grouped into probabilistic models or latent variable models that aim to relate a set of observable variables $\mathcal{X}$ to a set of latent variables $\mathcal{Z}$ based on a number of probability distributions [1]. Under a specification of model structure, the model parameters $\mathbf{\Lambda}$ of probability functions are estimated by maximizing the likelihood function with respect to model parameters $\mathbf{\Lambda}$. Basically, probabilistic model is configured as a *top-down* structure or representation. Starting

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

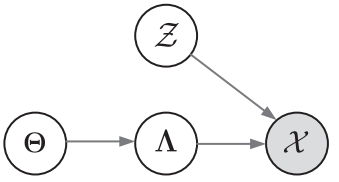

Figure 2.1 A hierarchical Bayesian representation for a latent variable model with hyperparameters $\mathbf{\Theta}$, parameters $\mathbf{\Lambda}$, latent variables $\mathcal{Z}$, and observed variables $\mathcal{X}$. Shaded node means observations. Unshaded nodes means latent variables.

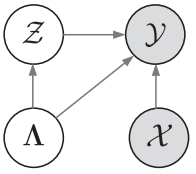

Figure 2.2 A supervised probabilistic model with parameters $\mathbf{\Lambda}$, latent variables $\mathcal{Z}$, and observed samples $\mathcal{X}$ and observed labels $\mathcal{Y}$.

from the hyperparameters $\mathbf{\Theta}$, model parameters are drawn or represented by a prior density $p(\mathbf{\Lambda}|\mathbf{\Theta})$. Having the model parameters $\mathbf{\Lambda}$, the speech training samples $\mathcal{X}$ are generated by a likelihood function $p(\mathcal{X}|\mathbf{\Lambda})$, which is marginalized over discrete latent variables by

$$p(\mathcal{X}|\mathbf{\Lambda}) = \sum_{\mathcal{Z}} p(\mathcal{X}, \mathcal{Z}|\mathbf{\Lambda}) \tag{2.1}$$

or over continuous latent variable by

$$p(\mathcal{X}|\mathbf{\Lambda}) = \int p(\mathcal{X}, \mathcal{Z}|\mathbf{\Lambda}) d\mathcal{Z}. \tag{2.2}$$

Probabilistic representation of data generation is intuitive with the graphical representation as shown in Figure 2.1. It is straightforward to build a supervised regression or classification model by directly maximizing the *conditional likelihood* $p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})$ where the training samples consist of observation samples $\mathcal{X}$ as well as their regression or class labels $\mathcal{Y}$. Figure 2.2 illustrates a latent variable model for supervised training. The conditional likelihood with discrete latent variables is expressed by

$$\begin{aligned} p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}) &= \sum_{\mathcal{Z}} p(\mathcal{Y}, \mathcal{Z}|\mathcal{X}, \mathbf{\Lambda}) \\ &= \sum_{\mathcal{Z}} p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}).p(\mathcal{Z}|\mathbf{\Lambda}) \end{aligned} \tag{2.3}$$

The supervised, unsupervised, and semi-supervised models can be flexibly performed and constructed by optimizing the corresponding likelihood functions in individual or hybrid style. The uncertainties of observations, parameters, and hyperparameters are easily incorporated in the resulting solution based on the probability functions. Domain

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

knowledge can be represented as constraints in the likelihood function, which essentially turns the maximum-likelihood problem into a constrained optimization problem.

For real-world applications, the probabilistic models are complicated with different latent variables in various model structures. There are a number of approximate inference algorithms that are available for solving the optimization problems in probabilistic models. However, the approximate inference or decoding algorithm is usually difficult to derive. Direct optimization over a likelihood function is prone to be intractable. Alternatively, indirect optimization over a lower bound of the likelihood function (or called the evidence lower bound [ELBO]) is implemented as an analytical solution. There are many latent variable models in various information systems. The latent variable models in speaker recognition, addressed in this book, include the Gaussian mixture model (GMM), factor analysis (FA), probabilistic linear discriminant analysis (PLDA), joint factor analysis (JFA), and mixture of PLDA, which will be addressed in Sections 3.1, 3.3, 3.5, 3.7, and 5.4.1, respectively.

2.1.2 Neural Networks

Deep neural networks (DNNs) have successfully boosted the performance of numerous information processing systems in many domains. The key to the success of DNNs is originated from the deep structured and hierarchical learning, which mimic the information processing functions of human's neural system. Multiple layers of nonlinear processing units are introduced to represent very complicated mapping between input samples and output targets. High-level abstraction is extracted to judge the target values corresponding to raw input samples. For example, the deeper layers in Figure 2.3 are capable of identifying the high-level semantic meaning of the input image and use the meaning for predicting the type of action in the image.

Table 2.1 illustrates the conceptual differences between the paradigms of probabilistic models and neural network models. Different from probabilistic models, neural networks are constructed as a bottom-up structure. The observed samples are received and propagated in a layer-wise model that is known as the multilayer perceptron (MLP) [20]. MLP is established as a distributed computer. The computation style in neural networks is consistent in different layers. In general, a neural network is seen as a black box and is basically hard to analyze and interpret. Neural networks can be considered

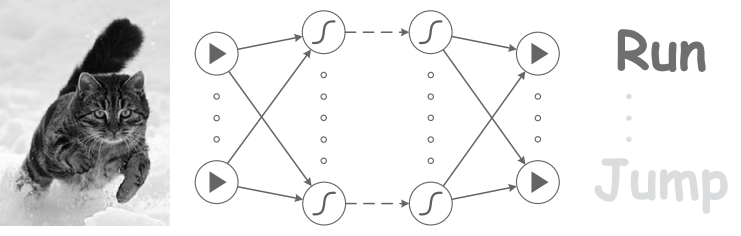

Figure 2.3 A deep neural network is constructed to retrieve the semantic meaning from an input image.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

Table 2.1 Probabilistic models versus neural networks from different machine learning perspectives.

	Probabilistic model	Neural network
Structure	Top-down	Bottom-up
Representation	Intuitive	Distributed
Interpretation	Easier	Harder
Supervised/unsupervised learning	Easier	Harder
Incorporation of domain knowledge	Easy	Hard
Incorporation of constraint	Easy	Hard
Incorporation of uncertainty	Easy	Hard
Learning algorithm	Many algorithms	Back-propagation
Inference/decode process	Harder	Easier
Evaluation on	ELBO	End performance

as deterministic learning machines where the uncertainty of model parameters is disregarded. A fixed style of computation model is implemented so that it is difficult to incorporate domain knowledge and constraint in the optimization procedure. However, neural networks are easy to implement and infer. The learning objective is based on the error backpropagation algorithm, which optimizes the end performance by propagating the gradients from the output layer back to the input layer. Only the computations of gradients and their propagations are required. Computation procedure is simple and intuitive.

A desirable machine learning algorithm for speaker recognition should preserve good properties from both paradigms. Our goal is to pursue modern machine learning models and optimization procedures that are explainable and intuitive. The details of deep neural networks will be discussed in Chapter 4. In what follows, we introduce the fundamentals of model inference that will be used in the construction of probabilistic models for speaker recognition.

2.2 Expectation-Maximization Algorithm

This subsection addresses the general principle of a maximum likelihood (ML) estimation of probabilistic models and the detailed procedure of how the Jensen's inequality is used to derive the expectation-maximization (EM) algorithm [8]. The EM algorithm provides an efficient approach to tackling the incomplete data problem in a ML estimation; it also provides a general solution to the estimation of model parameters based on the ML principle.

2.2.1 Maximum Likelihood

We assume that the spectral matrix of a speech signal $\mathcal{X}$ is observed, which is generated by a distribution $p(\mathcal{X}|\mathbf{\Lambda})$ parameterized by $\mathbf{\Lambda}$ under a specialized model. The optimal

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

ML parameters $\mathbf{\Lambda}_{\text{ML}}$ of a speaker recognition model are estimated by maximizing the accumulated likelihood function from T frames of signal spectra $\mathcal{X} = \{\mathbf{x}_t\}_{t=1}^{T}$, i.e.,

$$\mathbf{\Lambda}_{\text{ML}} = \underset{\mathbf{\Lambda}}{\text{argmax}}\, p(\mathcal{X}|\mathbf{\Lambda}). \tag{2.4}$$

Basically, the missing data problem happens in the estimation in Eq. 2.4 owing to the absence of latent variables $\mathcal{Z}$ in the representation. We could not directly work on such an optimization. To deal with this problem in model inference, the complete data $\{\mathcal{X}, \mathcal{Z}\}$, consisting of observation data $\mathcal{X}$ and latent variables $\mathcal{Z}$, are introduced in expression of a ML estimation in a form of

$$\mathbf{\Lambda}_{\text{ML}} = \underset{\mathbf{\Lambda}}{\text{argmax}} \sum_{\mathcal{Z}} p(\mathcal{X}, \mathcal{Z}|\mathbf{\Lambda}). \tag{2.5}$$

This optimization considers discrete latent variables $\mathcal{Z}$, which are integrated by summation.

2.2.2 Iterative Procedure

In general, the exact solution to global optimum in optimization problem of Eq. 2.4 is intractable because of an incomplete data problem. To cope with this problem, the expectation-maximization (EM) algorithm [8] was developed to find an analytical solution to ML estimation. This algorithm is implemented by alternatively running the expectation step (also called E-step) and the maximization step (M-step) and iteratively finding the convergence to local optimum for ML estimate of model parameters $\mathbf{\Lambda}_{\text{ML}}$. In this ML estimation, the first step (E-step) is performed by building an auxiliary function $Q(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)})$ or calculating an expectation function of the log likelihood function with new parameters $\mathbf{\Lambda}$ conditional on the parameters $\mathbf{\Lambda}^{(\tau)}$ at previous iteration τ

$$\begin{aligned} Q(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) &= \mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X}, \mathcal{Z}|\mathbf{\Lambda})|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}] \\ &= \sum_{\mathcal{Z}} p(\mathcal{Z}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}) \log p(\mathcal{X}, \mathcal{Z}|\mathbf{\Lambda}). \end{aligned} \tag{2.6}$$

The second step (M-step) is to estimate the updated parameters $\mathbf{\Lambda}^{(\tau+1)}$ at iteration $\tau + 1$ by maximizing the auxiliary function based on

$$\mathbf{\Lambda}^{(\tau+1)} = \underset{\mathbf{\Lambda}}{\text{argmax}}\, Q(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}). \tag{2.7}$$

In this iterative procedure, new parameters $\mathbf{\Lambda}^{(\tau+1)}$ are then viewed as the old parameters when moving to next EM iteration $\tau + 2$ for estimating the newly updated parameters $\mathbf{\Lambda}^{(\tau+2)}$. Such an iterative estimation only assures a local optimum while the global optimum solution is not guaranteed. Instead of directly maximizing the likelihood in Eq. 2.5, the EM algorithm indirectly maximizes the auxiliary function and theoretically proves a nondecreasing auxiliary function through a number of EM iterations [8]. This property is attractive and useful in order to run debugging over

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

the likelihood values in successive EM iterations in the implementation procedure. Nevertheless, the increase of auxiliary function may be rapidly saturated in the EM algorithm. But, a well-tuned setting of initialization of model parameters $\mathbf{\Lambda}^{(0)}$ is helpful to globally seek the optimal parameters.

It is important to investigate how the iterative optimization over auxiliary function $Q(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)})$ is performed to continuously increase the log likelihood $\log p(\mathcal{X}|\mathbf{\Lambda})$. Here, $\mathcal{Y} \triangleq \{\mathcal{X}, \mathcal{Z}\}$ denotes the complete data. Log likelihood function is denoted by $L(\mathbf{\Lambda}) \triangleq \log p(\mathcal{X}|\mathbf{\Lambda})$. We would like to investigate the increase of likelihood function $p(\mathcal{X}|\mathbf{\Lambda})$ by means of evaluating the increase of log likelihood function $L(\mathbf{\Lambda})$ since logarithm log is a monotonically increasing function. To do so, we first have the equation

$$p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}) = \frac{p(\mathcal{Y}, \mathcal{X}|\mathbf{\Lambda})}{p(\mathcal{X}|\mathbf{\Lambda})} = \frac{p(\mathcal{X}|\mathcal{Y}, \mathbf{\Lambda})p(\mathcal{Y}|\mathbf{\Lambda})}{p(\mathcal{X}|\mathbf{\Lambda})} = \frac{p(\mathcal{Y}|\mathbf{\Lambda})}{p(\mathcal{X}|\mathbf{\Lambda})}, \tag{2.8}$$

which is obtained by using $p(\mathcal{X}|\mathcal{Y}, \mathbf{\Lambda}) = p(\mathcal{X}|\mathcal{X}, \mathcal{Z}, \mathbf{\Lambda}) = 1$. Having this equation, we manipulate it by taking logarithm and then taking expectation over both sides of Eq. 2.8 with respect to $p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})$ to yield

$$\begin{aligned} L(\mathbf{\Lambda}) &= \log p(\mathcal{Y}|\mathbf{\Lambda}) - \log p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}) && \text{(2.9a)} \\ &= \mathbb{E}_{\mathcal{Y}}[\log p(\mathcal{Y}|\mathbf{\Lambda})|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}] - \mathbb{E}_{\mathcal{Y}}[\log p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}] && \text{(2.9b)} \\ &= Q(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) - H(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) && \text{(2.9c)} \end{aligned}$$

where $H(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)})$ is defined by

$$H(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) \triangleq \mathbb{E}_{\mathcal{Y}}[\log p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}]. \tag{2.10}$$

Here, the RHS of Eq. 2.9c is equal to $L(\mathbf{\Lambda})$ because

$$\mathbb{E}_{\mathcal{Y}}[L(\mathbf{\Lambda})] = \int p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})L(\mathbf{\Lambda})d\mathcal{Y} = L(\mathbf{\Lambda}). \tag{2.11}$$

By considering the negative logarithm as a convex function, the Jensen's inequality can be applied to derive

$$\begin{aligned} &H(\mathbf{\Lambda}^{(\tau)}|\mathbf{\Lambda}^{(\tau)}) - H(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) \\ &\quad= \mathbb{E}_{\mathcal{Y}}\left[\log \frac{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})}{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})}\middle| \mathcal{X}, \mathbf{\Lambda}^{(\tau)}\right] \\ &\quad= \int p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})\left(-\log \frac{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})}{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})}\right) d\mathcal{Y} \\ &\quad= \mathcal{D}_{\mathrm{KL}}(p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}) \| p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})) \\ &\quad\geq -\log\left(\int p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})\frac{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda})}{p(\mathcal{Y}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})} d\mathcal{Y}\right) = 0 \end{aligned} \tag{2.12}$$

where $\mathcal{D}_{\mathrm{KL}}(\cdot\|\cdot)$ means the Kullback–Leibler divergence [21]. Substituting Eq. 2.12 into Eq. 2.9c yields

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

$$\begin{aligned} L(\boldsymbol{\Lambda}) &= Q(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) - H(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) \\ &\geq Q(\boldsymbol{\Lambda}^{(\tau)}|\boldsymbol{\Lambda}^{(\tau)}) - H(\boldsymbol{\Lambda}^{(\tau)}|\boldsymbol{\Lambda}^{(\tau)}) = L(\boldsymbol{\Lambda}^{(\tau)}), \end{aligned} \tag{2.13}$$

which is derived by adopting the useful inequality

$$\begin{aligned} & Q(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) \geq Q(\boldsymbol{\Lambda}^{(\tau)}|\boldsymbol{\Lambda}^{(\tau)}) \\ & \Rightarrow L(\boldsymbol{\Lambda}) \geq L(\boldsymbol{\Lambda}^{(\tau)}) \\ & \Rightarrow p(\mathcal{X}|\boldsymbol{\Lambda}) \geq p(\mathcal{X}|\boldsymbol{\Lambda}^{(\tau)}). \end{aligned} \tag{2.14}$$

In summary, we increase the auxiliary function $Q(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})$, which equivalently increase the log likelihood function $L(\boldsymbol{\Lambda})$ or the likelihood function $p(\mathcal{X}|\boldsymbol{\Lambda})$. Auxiliary function is maximized to increase the likelihood function given by the updated parameters $\boldsymbol{\Lambda}^{(\tau)} \rightarrow \boldsymbol{\Lambda}^{(\tau+1)}$. However, we don't directly optimize the original likelihood function. Indirectly optimizing the auxiliary function will result in a local optimum in estimation of ML parameters.

2.2.3 Alternative Perspective

An alternative view of the EM algorithm is addressed by introducing a variational distribution $q(\mathcal{Z})$ for latent variables $\mathcal{Z}$. No matter what distribution $q(\mathcal{Z})$ is chosen, we can derive a general formula for decomposition of log likelihood in a form of

$$\log p(\mathcal{X}|\boldsymbol{\Lambda}) = \mathcal{D}_{\mathrm{KL}}(q\|p) + \mathcal{L}(q, \boldsymbol{\Lambda}). \tag{2.15}$$

This formula is obtained because

$$\begin{aligned} \mathcal{D}_{\mathrm{KL}}(q\|p) &= -\sum_{\mathcal{Z}} q(\mathcal{Z}) \log\left\{\frac{p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda})}{q(\mathcal{Z})}\right\} \\ &= -\mathbb{E}_q[\log p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda})] - \mathbb{H}_q[\mathcal{Z}] \end{aligned} \tag{2.16}$$

and

$$\begin{aligned} \mathcal{L}(q, \boldsymbol{\Lambda}) &= \sum_{\mathcal{Z}} q(\mathcal{Z}) \log\left\{\frac{p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\Lambda})}{q(\mathcal{Z})}\right\} \\ &= \mathbb{E}_q[\log p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\Lambda})] + \mathbb{H}_q[\mathcal{Z}], \end{aligned} \tag{2.17}$$

where the entropy of $q(\mathcal{Z})$ is defined by

$$\mathbb{H}_q[\mathcal{Z}] \triangleq -\sum_{\mathcal{Z}} q(\mathcal{Z}) \log q(\mathcal{Z}). \tag{2.18}$$

We have the following interpretation. The log likelihood $\log p(\mathcal{X}|\boldsymbol{\Lambda})$ is decomposed as a KL divergence between $q(\mathcal{Z})$ and posterior distribution $p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda})$ and a lower bound $\mathcal{L}(q, \boldsymbol{\Lambda})$ of the log likelihood function $\log p(\mathcal{X}|\boldsymbol{\Lambda})$. We call $\mathcal{L}(q, \boldsymbol{\Lambda})$ as the lower bound since KL term in Eq. 2.15 meets the property $\mathcal{D}_{\mathrm{KL}}(q\|p) \geq 0$. This property was also shown in Eq. 2.12. Figure 2.4(a) illustrates the relation among log likelihood function, KL divergence, and lower bound. ML estimation of model parameters $\boldsymbol{\Lambda}$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

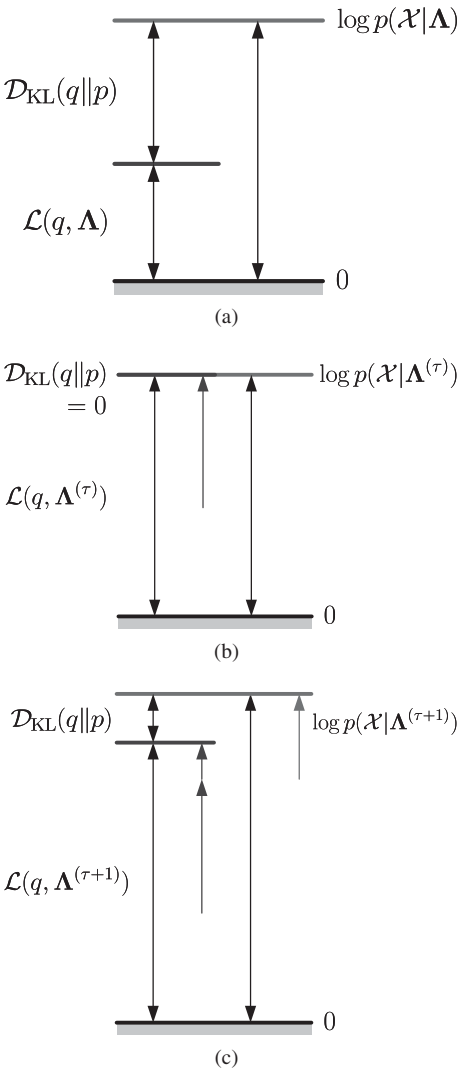

Figure 2.4 Illustration for (a) decomposing a log likelihood $\log p(\mathcal{X}|\mathbf{\Lambda})$ into a KL divergence $\mathcal{D}_{\mathrm{KL}}(q\|q)$ and a lower bound $\mathcal{L}(q, \mathbf{\Lambda})$, (b) updating the lower bound by setting KL divergence to zero with new distribution $q(\mathcal{Z}) = p(\mathcal{Z}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)})$, and (c) updating lower bound again by using new parameters $\mathbf{\Lambda}^{(\tau+1)}$ [1]. [Based on *Pattern Recognition and Machine Learning (Figure. 9.11–Figure. 9.13), by C. M. Bishop, 2006, Springer.*]

turns out as an indirect optimization via finding a variational distribution $q(\mathcal{Z})$. This indirect optimization is carried out by alternatively executing two steps to estimate the distribution of latent variables $q(\mathcal{Z})$ and the model parameters $\mathbf{\Lambda}$. The first step is to use the current estimate $\mathbf{\Lambda}^{(\tau)}$ to estimate the approximate distribution $q(\mathcal{Z})$ by

$$q(\mathcal{Z}) = p(\mathcal{Z}|\mathcal{X}, \mathbf{\Lambda}^{(\tau)}), \tag{2.19}$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

which is obtained by imposing $\mathcal{D}_{\mathrm{KL}}(q\|p) = 0$. Namely, the variational distribution is derived as the posterior distribution given by the old parameters $\boldsymbol{\Lambda}^{(\tau)}$ at iteration τ. As seen in Figure 2.4(b), the variational lower bound is improved accordingly. Then, in the second step, shown in Figure 2.4(c), we fix $q(\mathcal{Z})$ and maximize again the lower bound $\mathcal{L}(q, \boldsymbol{\Lambda})$ with respect to $\boldsymbol{\Lambda}$ to estimate new model parameters $\boldsymbol{\Lambda}^{(\tau+1)}$ at iteration $\tau + 1$. It is noted that this lower bound with variational distribution $q(\mathcal{Z})$ in Eq. 2.19 is exactly the same as the auxiliary function for the E-step in EM algorithm as shown below[1]

$$\begin{aligned}\mathcal{L}(q, \boldsymbol{\Lambda}) &= \mathbb{E}_q[\log p(\mathcal{X}|\mathcal{Z}, \boldsymbol{\Lambda}) + \log p(\mathcal{Z}|\boldsymbol{\Lambda})] + \text{const} \\ &= \mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\Lambda})|\mathcal{X}, \boldsymbol{\Lambda}^{(\tau)}] + \text{const} \\ &= Q(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) + \text{const}.\end{aligned} \tag{2.20}$$

Here, the terms independent of $\boldsymbol{\Lambda}$ are merged in a constant. Given the parameters $\boldsymbol{\Lambda}^{(\tau+1)}$, the distribution of latent variables is further updated at the next EM iteration as $\boldsymbol{\Lambda}^{(\tau+2)} \leftarrow \boldsymbol{\Lambda}^{(\tau+1)}$. A series of parameter updating are obtained by

$$\boldsymbol{\Lambda}^{(0)} \rightarrow \boldsymbol{\Lambda}^{(1)} \rightarrow \boldsymbol{\Lambda}^{(2)} \rightarrow \cdots \tag{2.21}$$

This updating assures that the lower bound $\mathcal{L}(q, \boldsymbol{\Lambda})$ is sequentially elevated and the log likelihood function $\log p(\mathcal{X}|\boldsymbol{\Lambda})$ is accordingly improved. Therefore, finding $q(\mathcal{Z})$ in Eq. 2.19 is equivalent to calculating the auxiliary function in E-step. Maximizing the lower bound $\mathcal{L}(q, \boldsymbol{\Lambda})$ with respect to model parameters $\boldsymbol{\Lambda}$ corresponds to fulfill the M-step in EM iterative updating. Originally, the log likelihood $\log p(\mathcal{X}|\boldsymbol{\Lambda})$ under latent variable model with $\mathcal{Z}$ is seen a non-convex or non-concave function. However, the inference based on the lower bound or expectation function in Eqs. 2.6 and 2.20 under Gaussian observation distribution becomes a concave optimization as illustrated in Figure 2.5. In this figure, the continuously maximizing the lower bound or the iterative updating of model parameters from $\boldsymbol{\Lambda}^{(\tau)}$ to $\boldsymbol{\Lambda}^{(\tau+1)}$ via EM steps assures the optimum in a ML estimate $\boldsymbol{\Lambda}_{\mathrm{ML}}$. Owing to a nondecreasing lower bound or auxiliary function, we guarantee the convergence of model parameters within a few EM steps.

2.2.4 Maximum *A Posteriori*

From a Bayesian perspective, it is meaningful to extend the optimization over the posterior distribution $p(\boldsymbol{\Lambda}|\mathcal{X})$ and address how to seek the MAP estimate $\boldsymbol{\Lambda}_{\mathrm{MAP}}$ by maximizing the posterior distribution of model parameters $\boldsymbol{\Lambda}$ by giving observations $\mathcal{X}$. The discrete latent variables $\mathcal{Z}$ are introduced in the following MAP estimation to estimate the mode $\boldsymbol{\Lambda}_{\mathrm{MAP}}$ of posterior distribution from the collection of training data $\mathcal{X}$

$$\begin{aligned}\boldsymbol{\Lambda}_{\mathrm{MAP}} &= \underset{\boldsymbol{\Lambda}}{\operatorname{argmax}}\, p(\boldsymbol{\Lambda}|\mathcal{X}) \\ &= \underset{\boldsymbol{\Lambda}}{\operatorname{argmax}} \sum_{\mathcal{Z}} p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\Lambda}) p(\boldsymbol{\Lambda}).\end{aligned} \tag{2.22}$$

[1] Note that in Eq. 2.9c, we take the expectation with respect to the posterior $p(\mathcal{Y}|\mathcal{X}, \boldsymbol{\Lambda}^{(r)})$, which is equivalent to taking expectation with respect to $p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda}^{(r)})$ as $\mathcal{Y} = \{\mathcal{X}, \mathcal{Z}\}$.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

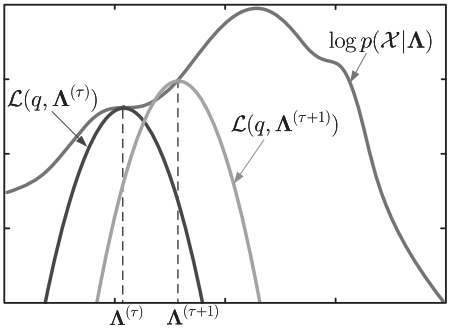

Figure 2.5 Illustration of updating the lower bound of log likelihood function $\mathcal{L}(q, \mathbf{\Lambda})$ in EM iteration from τ to $\tau + 1$. Lower bound is seen as an iterative log Gaussian approximation to the original log likelihood function $\log p(\mathcal{X}|\mathbf{\Lambda})$. [Based on *Pattern Recognition and Machine Learning (Figure. 9.14), by C. M. Bishop, 2006, Springer.*]

$$\underbrace{p(\mathbf{\Lambda}|\mathcal{X})}_{\text{Posterior}} = \frac{\overbrace{p(\mathcal{X}|\mathbf{\Lambda})}^{\text{Likelihood}}\overbrace{p(\mathbf{\Lambda})}^{\text{Prior}}}{\underbrace{\int p(\mathcal{X}|\mathbf{\Lambda})p(\mathbf{\Lambda})d\mathbf{\Lambda}}_{\text{Evidence}}}$$

Figure 2.6 Bayes rule for illustrating the relation among the posterior distribution, likelihood function, prior distribution, and evidence function.

In Eq. 2.22, the summation is taken due to the discrete latent variables $\mathcal{Z}$. Figure 2.6 shows the relation among posterior distribution, likelihood function, prior distribution, and evidence function, which is obtained by the Bayes theorem. We don't really need to compute the integral in evidence function for the MAP estimation because the evidence function $p(\mathcal{X}) = \int p(\mathcal{X}|\mathbf{\Lambda})p(\mathbf{\Lambda})d\mathbf{\Lambda}$ is independent of $\mathbf{\Lambda}$. The MAP estimation is an extension of the ML estimation by additionally considering the prior distribution that acted as a regularization term to reshape the estimation of model parameters. It is obvious that MAP provides advantage over ML for a richer parameter estimation. Importantly, if a conjugate prior distribution is chosen and combined with the likelihood function in a form of exponential distribution, a MAP estimation comparably searches the mode of the corresponding posterior distribution, which is formulated as the same distribution but with different parameters as the prior distribution. Conjugate prior is crucial in a MAP estimation. If we don't adopt the conjugate prior, the analytical posterior distribution does not exist for finding the mode or optimum of posterior distribution.

Here, we address an EM algorithm for the MAP estimation, which is an extension from EM based on a ML estimation. The incomplete data problem in a MAP estimation is tackled via EM steps to derive an analytical solution to local optimum $\mathbf{\Lambda}_{\text{MAP}}$. The resulting MAP-EM algorithm is formulated as E-step and M-step in an iterative learning procedure that updates new estimate $\mathbf{\Lambda}$ from the old parameters $\mathbf{\Lambda}^{(\tau)}$ at iteration τ. In the first step (E-step), we calculate the posterior auxiliary function, which is an expectation over posterior distribution

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

$$
\begin{aligned}
R(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) &= \mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X},\mathcal{Z},\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}] \\
&= \underbrace{\mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X},\mathcal{Z}|\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}]}_{Q(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})} + \log p(\boldsymbol{\Lambda}).
\end{aligned} \tag{2.23}
$$

In Eq. 2.23, the prior term $\log p(\boldsymbol{\Lambda})$ is independent of $\mathcal{Z}$ and is accordingly outside the expectation function. Obviously, different from the ML estimation, there is an additional factor $\log p(\boldsymbol{\Lambda})$ in the MAP estimation due to the prior density of model parameters. In the second step (M-step), we maximize the posterior expectation function to estimate the new model parameters $\boldsymbol{\Lambda}^{(\tau)} \rightarrow \boldsymbol{\Lambda}^{(\tau+1)}$ according to

$$
\boldsymbol{\Lambda}^{(\tau+1)} = \underset{\boldsymbol{\Lambda}}{\operatorname{argmax}}\, R(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}). \tag{2.24}
$$

The parameters are updated by $\boldsymbol{\Lambda}^{(\tau)} \rightarrow \boldsymbol{\Lambda}^{(\tau+1)}$ at iteration $\tau+1$ and then treated as the old parameters for estimation of new parameters at iteration $\tau+2$ in the next EM iteration.

We may further investigate the optimization of expectation function $R(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})$, which can result in local optimum of $p(\boldsymbol{\Lambda}|\mathcal{X})$ or $p(\mathcal{X}|\boldsymbol{\Lambda})p(\boldsymbol{\Lambda})$. To do so, we first have the definition of logarithm of joint distribution $L_{\text{MAP}}(\boldsymbol{\Lambda}) \triangleq \log\left(p(\mathcal{X}|\boldsymbol{\Lambda})p(\boldsymbol{\Lambda})\right)$ and use the relation

$$
p(\mathcal{X}|\boldsymbol{\Lambda}) = \frac{p(\mathcal{X},\mathcal{Z}|\boldsymbol{\Lambda})}{p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Lambda})}, \tag{2.25}
$$

to rewrite the expression as

$$
L_{\text{MAP}}(\boldsymbol{\Lambda}) = \log p(\mathcal{X},\mathcal{Z}|\boldsymbol{\Lambda}) - \log p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Lambda}) + \log p(\boldsymbol{\Lambda}). \tag{2.26}
$$

By referring Eq. 2.9c and taking expectation of both sides of Eq. 2.26 with respect to posterior distribution $p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)})$, we yield

$$
\begin{aligned}
L_{\text{MAP}}(\boldsymbol{\Lambda}) &= \mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X},\mathcal{Z}|\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}] \\
&\quad - \mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}] + \log p(\boldsymbol{\Lambda}) \\
&= \underbrace{\mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{X},\mathcal{Z}|\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}] + \log p(\boldsymbol{\Lambda})}_{=R(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})} \\
&\quad - \underbrace{\mathbb{E}_{\mathcal{Z}}[\log p(\mathcal{Z}|\mathcal{X},\boldsymbol{\Lambda})|\mathcal{X},\boldsymbol{\Lambda}^{(\tau)}]}_{\triangleq H(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})}.
\end{aligned} \tag{2.27}
$$

The terms $L_{\text{MAP}}(\boldsymbol{\Lambda})$ and $\log p(\boldsymbol{\Lambda})$ are independent of $\mathcal{Z}$ and are unchanged in Eq. 2.27. The posterior auxiliary function is decomposed as

$$
R(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) = L_{\text{MAP}}(\boldsymbol{\Lambda}) + H(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}). \tag{2.28}
$$

Because it has been shown in Eq. 2.12 that $H(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)})$ is continuously decreasing, i.e.,

$$
H(\boldsymbol{\Lambda}|\boldsymbol{\Lambda}^{(\tau)}) \leq H(\boldsymbol{\Lambda}^{(\tau)}|\boldsymbol{\Lambda}^{(\tau)}), \tag{2.29}
$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

we can derive

$$\begin{aligned} & R(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)}) \geq R(\mathbf{\Lambda}^{(\tau)}|\mathbf{\Lambda}^{(\tau)}) \\ \Rightarrow\ & L_{\text{MAP}}(\mathbf{\Lambda}) \geq L_{\text{MAP}}(\mathbf{\Lambda}^{(\tau)}) \\ \Rightarrow\ & p(\mathbf{\Lambda}|\mathcal{X}) \geq p(\mathbf{\Lambda}^{(\tau)}|\mathcal{X}). \end{aligned} \tag{2.30}$$

Namely, the increasing posterior auxiliary function $R(\mathbf{\Lambda}|\mathbf{\Lambda}^{(\tau)})$ will lead to an increasing logarithm of posterior distribution $L(\mathbf{\Lambda})$ or equivalently an increasing posterior distribution $p(\mathbf{\Lambda}|\mathcal{X})$. An EM algorithm is completed for a MAP estimation, which attains a converged local optimum for MAP parameters.

2.3 Approximate Inference

Variational Bayesian (VB) inference has been originated in machine learning literature since the late 1990s [22, 23]. VB theory has become a popular tool for approximate Bayesian inference in the presence of a latent variable model by implementing the EM-like algorithm. This section would like to construct a general latent variable model from a set of training data $\mathcal{X} = \{\mathbf{x}_t\}_{t=1}^{T}$ based on the set of variables including latent variables $\mathcal{Z}$, parameters $\mathbf{\Lambda}$, and hyperparameters $\mathbf{\Psi}$. All such variables are used to build an assumed topology $\mathcal{T}$ in learning representation. In this learning process, a fundamental issue in VB algorithm is to infer the joint posterior distribution

$$p(\mathcal{Z}|\mathcal{X}) \qquad \text{where } \mathcal{Z} = \{Z_j\}_{j=1}^{J}, \tag{2.31}$$

where all variables are included in the inference problem. Given the posterior distribution $p(\mathcal{Z}|\mathcal{X})$, the auxiliary functions in Eq. 2.23 are calculated to fulfill the E-step of an EM algorithm for both a ML and a MAP estimation procedures, respectively. Nevertheless, a number of latent variables do exist in probabilistic speaker recognition systems. These variables are coupled each other in calculation of posterior distribution. For instance, in a joint factor analysis model shown in Eq. 3.165, the speaker factor $\mathbf{y}(s)$ and the channel factor $\mathbf{x}_h(s)$ are correlated in their posterior distributions.

However, the analytical solution to exact calculation of posterior distribution $p(\mathcal{Z}|\mathcal{X})$ does not exist. The variational method is applied to carry out an approximate inference to handle this issue. Using the variational inference, an approximate or variational posterior distribution $q(\mathcal{Z})$ is factorized and viewed as a proxy to the original true distribution $p(\mathcal{Z}|\mathcal{X})$. Figure 2.7 illustrates the idea of approximate inference that analytically uses the variational distribution $q(\mathcal{Z})$ to approximate true posterior $p(\mathcal{Z}|\mathcal{X})$.

In the sequel, the underlying concept of approximating true posterior $p(\mathcal{Z}|\mathcal{X})$ using variational distribution $q(\mathcal{Z})$ is addressed. We formulate the variational Bayesian expectation-maximization VB-EM algorithm to implement a ML or MAP estimation where multiple latent variables are coupled in original posterior distribution. For notation simplicity, we formulate the approximation in the presence of continuous latent variables. The marginalization of probabilistic distribution is performed by integrating latent variables $\int d\mathcal{Z}$ instead of summing latent variables $\sum_{\mathcal{Z}}$.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

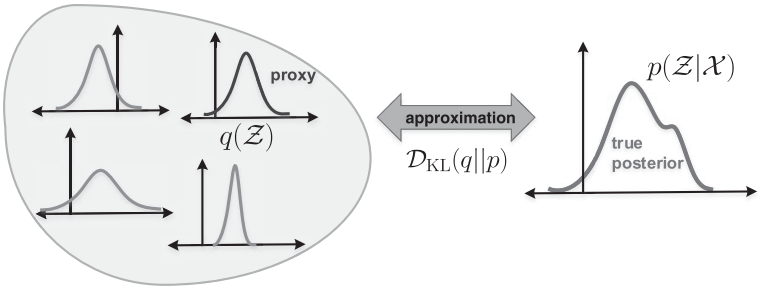

Figure 2.7 Illustration of finding an approximate Gaussian distribution $q(\mathcal{Z})$ that is factorizable and optimally close to the true posterior $p(\mathcal{Z}|\mathcal{X})$.

2.3.1 Variational Distribution

Approximate inference needs to approximate the true posterior distribution $p(\mathcal{Z}|\mathcal{X})$ by using a variational distribution $q(\mathcal{Z})$ based on a distance metric for two distributions. In Section 2.2.3, a Kullback–Leibler (KL) divergence [21] was adopted as a measure of distance between two distance in a ML–EM algorithm. In this section, a KL divergence is used to estimate the variational distribution. A KL divergence between $q(\mathcal{Z})$ and $p(\mathcal{Z}|\mathcal{X})$ is defined and decomposed by

$$\begin{aligned}\mathcal{D}_{\mathrm{KL}}(q(\mathcal{Z})\|p(\mathcal{Z}|\mathcal{X})) &\triangleq \int q(\mathcal{Z})\log\frac{q(\mathcal{Z})}{p(\mathcal{Z}|\mathcal{X})}d\mathcal{Z}\\ &= -\mathbb{E}_q[\log p(\mathcal{Z}|\mathcal{X})] - \mathbb{H}_q[\mathcal{Z}]\\ &= \int q(\mathcal{Z})\log\frac{q(\mathcal{Z})}{\frac{p(\mathcal{X},\mathcal{Z})}{p(\mathcal{X})}}d\mathcal{Z}\\ &= \log p(\mathcal{X}) - \underbrace{\int q(\mathcal{Z})\log\frac{p(\mathcal{X},\mathcal{Z})}{q(\mathcal{Z})}d\mathcal{Z}}_{\triangleq\mathcal{L}(q(\mathcal{Z}))},\end{aligned} \tag{2.32}$$

where

$$\begin{aligned}\mathcal{L}(q(\mathcal{Z})) &\triangleq \int q(\mathcal{Z})\log\frac{p(\mathcal{X},\mathcal{Z})}{q(\mathcal{Z})}d\mathcal{Z}\\ &= \mathbb{E}_q[\log p(\mathcal{X},\mathcal{Z})] + \mathbb{H}_q[\mathcal{Z}].\end{aligned} \tag{2.33}$$

Here, $\mathcal{L}(q(\mathcal{Z}))$ is seen as the the variational lower bound of $\log p(\mathcal{X})$, which is also known as the evidence lower bound (ELBO) $\mathcal{L}(q(\mathcal{Z}))$. It is because KL divergence is nonnegative so as to obtain $\log p(\mathcal{X}) \geq \mathcal{L}(q(\mathcal{Z}))$. This result is similar to what we have addressed in Eqs. 2.15–2.17. The variational distribution $q(\mathcal{Z})$, which is closest to the original posterior distribution $p(\mathcal{Z}|\mathcal{X})$, is obtained by minimizing the KL divergence between them or equivalently maximizing the variational lower bound $\mathcal{L}(q(Z|\mathcal{X}))$. This is because the evidence function $\log p(\mathcal{X})$ is independent of $\mathcal{Z}$. Therefore, we have

$$\widehat{q}(\mathcal{Z}) = \underset{q(\mathcal{Z})}{\operatorname{argmax}}\,\mathcal{L}(q(\mathcal{Z})). \tag{2.34}$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

In the implementation, an analytical solution to Eq. 2.34 is derived under the constraint that the variational posterior distribution $q(\mathcal{Z})$ is factorizable. We therefore derive the variational inference procedure where a surrogate of original posterior $p(\mathcal{Z}|\mathcal{X})$, here an alternative posterior based on the variational distribution $q(\mathcal{Z})$, is estimated to find a ML $\mathbf{\Lambda}_{\mathrm{ML}}$ or a MAP model parameters $\mathbf{\Lambda}_{\mathrm{MAP}}$. The resulting variational Bayesian (VB) inference is also known as the factorized variational inference, since the approximate inference is realized by the constraint of factorization.

2.3.2 Factorized Distribution

According to the factorized variational inference, the variational distribution is assumed to be conditionally independent over J latent variables

$$q(\mathcal{Z}) = \prod_{j=1}^{J} q(Z_j). \tag{2.35}$$

Notably, the factorization is *not* imposed in true posterior $p(\mathcal{Z}|\mathcal{X})$. We preserve the original posterior and arrange the posterior of a target latent variable $p(Z_i|\mathcal{X})$ in a form of marginal distribution of $p(\mathcal{Z}|\mathcal{X})$ over all Z_j except Z_i

$$\begin{aligned} p(Z_i|\mathcal{X}) &= \int \cdots \int p(\mathcal{Z}|\mathcal{X}) \prod_{j \neq i}^{J} dZ_j \\ &\triangleq \int p(\mathcal{Z}|\mathcal{X}) dZ_{\setminus i}. \end{aligned} \tag{2.36}$$

Here, $Z_{\setminus i}$ is defined as the complementary set of Z_i. Eq. 2.36 is then used to calculate the following KL divergence between $q(Z_i)$ and $p(Z_i|\mathcal{X})$

$$\begin{aligned} \mathcal{D}_{\mathrm{KL}}(q(Z_i) \| p(Z_i|\mathcal{X})) &= \int q(Z_i) \log \frac{q(Z_i)}{\int p(\mathcal{Z}|\mathcal{X}) dZ_{\setminus i}} dZ_i \\ &= \int q(Z_i) \log \frac{q(Z_i)}{\int \frac{p(\mathcal{X},\mathcal{Z})}{p(\mathcal{X})} dZ_{\setminus i}} dZ_i \\ &= \log p(\mathcal{X}) - \int q(Z_i) \log \frac{\int p(\mathcal{X},\mathcal{Z}) dZ_{\setminus i}}{q(Z_i)} dZ_i. \end{aligned} \tag{2.37}$$

By considering the negative logarithm in the second term of Eq. 2.37 as a convex function, similar to Eq. 2.12, the Jensen's inequality is applied to yield

$$\begin{aligned} &\int q(Z_i) \log \left(\frac{\int p(\mathcal{X},\mathcal{Z}) dZ_{\setminus i}}{q(Z_i)} \right) dZ_i \\ &= \int q(Z_i) \log \left(\int q(Z_{\setminus i}) \frac{p(\mathcal{X},\mathcal{Z})}{q(\mathcal{Z})} dZ_{\setminus i} \right) dZ_i \\ &\geq \int q(Z_i) \int q(Z_{\setminus i}) \log \left(\frac{p(\mathcal{X},\mathcal{Z})}{q(\mathcal{Z})} \right) dZ_{\setminus i} dZ_i \\ &= \int q(\mathcal{Z}) \log \left(\frac{p(\mathcal{X},\mathcal{Z})}{q(\mathcal{Z})} \right) d\mathcal{Z} = \mathcal{L}(q(\mathcal{Z})). \end{aligned} \tag{2.38}$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

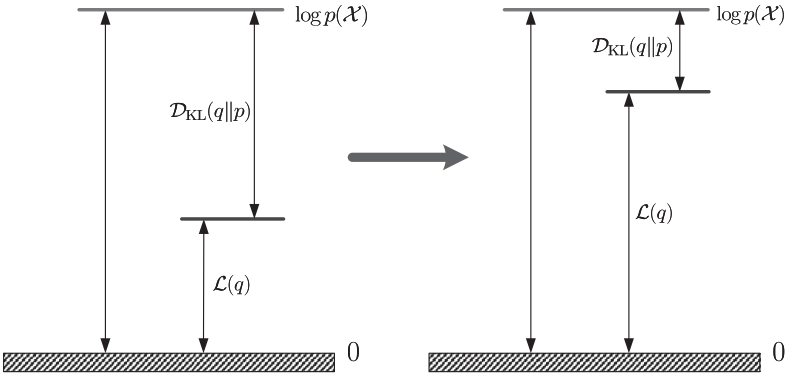

Figure 2.8 Illustration of minimizing KL divergence $\mathcal{D}_{\mathrm{KL}}(q\,\|\,p)$ via maximization of variational lower bound $\mathcal{L}(q)$ where $\log\, p(\mathcal{X})$ is fixed.

We therefore obtain the inequality

$$\mathcal{D}_{\mathrm{KL}}(q(Z_i)\|\, p(Z_i|\mathcal{X})) \leq \log\, p(\mathcal{X}) - \mathcal{L}(q(\mathcal{Z})). \tag{2.39}$$

Correspondingly, we minimize the KL divergence $\mathcal{D}_{\mathrm{KL}}(q(Z_i)\|\, p(Z_i|\mathcal{X}))$ or maximize the variational lower bound to derive the variational distribution for individual latent variable $q(Z_i)$ as follows:

$$\widehat{q}(Z_i) = \underset{q(Z_i)}{\operatorname{argmax}}\, \mathcal{L}(q(\mathcal{Z})). \tag{2.40}$$

The joint posterior distribution $\widehat{q}(\mathcal{Z})$ is then obtained by Eq. 2.35 by using J variational posterior distributions $\{\widehat{q}(Z_i)\}_{i=1}^{J}$ found by Eq. 2.40. An illustration of maximizing the variational lower bound $\mathcal{L}(q(\mathcal{Z}))$ or minimizing the KL divergence $\mathcal{D}_{\mathrm{KL}}(q(\mathcal{Z})\|\, p(\mathcal{Z}|\mathcal{X}))$ for estimation of variational distribution $q(\mathcal{X})$ is shown in Figure 2.8.

We can formulate the optimization problem in a form of

$$\begin{aligned} &\max_{q(Z_i)} \;\; \mathbb{E}_q[\log\, p(\mathcal{X},\mathcal{Z})] + \mathbb{H}_q[\mathcal{Z}] \\ &\text{subject to} \;\; \int_{\mathcal{Z}} q(d\mathcal{Z}) = 1. \end{aligned} \tag{2.41}$$

In this optimization, we may arrange the variational lower bound in Eq. 2.33 as

$$\mathcal{L}(q(\mathcal{Z})) = \int \prod_j q_j \left\{\log\, p(\mathcal{X},\mathcal{Z}) - \sum\nolimits_j \log q_j\right\} d\mathcal{Z} \tag{2.42a}$$

$$= \int q_i \underbrace{\int \log\, p(\mathcal{X},\mathcal{Z}) \prod_{j\neq i} q_j dZ_j\, dZ_i}_{\triangleq \log\, \widetilde{p}(\mathcal{X},Z_i)} - \int q_i \log q_i dZ_i + \text{const} \tag{2.42b}$$

$$= -\mathcal{D}_{\mathrm{KL}}(q(Z_i)\|\, \widetilde{p}(\mathcal{X},Z_i)) + \text{const}. \tag{2.42c}$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

In Eq. 2.42, we simply denote $q(Z_i)$ as q_i and define the term

$$\log \widetilde{p}(\mathcal{X}, Z_i) = \mathbb{E}_{\mathcal{Z}_{\setminus i}}[\log p(\mathcal{X}, \mathcal{Z})] + \text{const.} \tag{2.43}$$

The second term of Eq. 2.42(b) is obtained by noting that

$$\begin{aligned}
&\int \prod\nolimits_j q_j \left(\sum\nolimits_j \log q_j\right) d\mathcal{Z} \\
&= \int \left(\prod\nolimits_j q_j\right) \left(\log q_i + \sum\nolimits_{j\neq i} \log q_j\right) d\mathcal{Z} \\
&= \int \left(\prod\nolimits_j q_j\right) \log q_i d\mathcal{Z} + \int \left(\prod\nolimits_j q_j\right) \left(\sum\nolimits_{j\neq i} \log q_j\right) d\mathcal{Z} \\
&= \left(\prod\nolimits_{j\neq i} \int q_j dZ_j\right) \int q_i \log q_i dZ_i + \int \left(\prod\nolimits_{j\neq i} q_j\right) q_i \left(\sum\nolimits_{j\neq i} \log q_j\right) d\mathcal{Z} \\
&= \int q_i \log q_i dZ_i + \int \left(\prod\nolimits_{j\neq i} q_j\right) \left(\sum\nolimits_{j\neq i} \log q_j\right) dZ_{\setminus i} \int q_i dZ_i \\
&= \int q_i \log q_i dZ_i + \text{const independent of } q_i,
\end{aligned} \tag{2.44}$$

where we have used the property of distribution $\int q_i dZ_i = 1$. The factorized variational distribution is finally obtained by

$$\widehat{q}(Z_i) \propto \exp\left(\mathbb{E}_{\mathcal{Z}_{\setminus i}}\left[\log p(\mathcal{X}, \mathcal{Z})\right]\right). \tag{2.45}$$

More specifically, the exact solution to general form of the factorized distribution is expressed by

$$\widehat{q}(Z_i) = \frac{\exp\left(\mathbb{E}_{\mathcal{Z}_{\setminus i}}\left[\log p(\mathcal{X}, \mathcal{Z})\right]\right)}{\int \exp\left(\mathbb{E}_{\mathcal{Z}_{\setminus i}}\left[\log p(\mathcal{X}, \mathcal{Z})\right]\right) dZ_i} \tag{2.46}$$

where the normalization constant is included. From Eq. 2.45, we can see that the approximate posterior distribution of a target latent variable Z_i is inferred through calculating the joint distribution of training data $\mathcal{X}$ and all latent variables $\mathcal{Z}$. The logarithm of the derived variational distribution of a target variable Z_i is obtained by calculating the logarithm of the joint distribution over all visible and hidden variables and taking the expectation over all the other hidden variables $\mathcal{Z}_{\setminus i}$. Although the variational distribution is assumed to be factorizable, the learned target posterior distribution $\widehat{q}(Z_i)$ and the other posterior distributions $q(\mathcal{Z}_{\setminus i}) = \prod_{j\neq i}^{J} q(Z_j)$ are correlated each other. This finding can be seen from Eq. 2.45. As a result, starting from the initial variational distributions, we iteratively perform optimization for factorized distributions of all $q(Z_i)$.

2.3.3 EM versus VB-EM Algorithms

In general, we may follow the alternative perspective of the EM algorithm, as addressed in Section 2.2.3, to carry out the estimation of factorized posterior distributions $\{\widehat{q}(Z_j)\}_{j=1}^{J}$. This is comparable to fulfill an expectation step (E-step) similar to Eq. 2.19 in the standard EM algorithm. The key difference of VB-EM algorithm compared to EM algorithm is an additional factorization step due to the coupling of

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

various latent variables in true posterior distribution $p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda}^{(\tau)})$. Using the VB-EM algorithm, the factorized distribution $q(Z_i) = \widetilde{p}(\mathcal{X}, Z_i)$ does not lead to the result $\mathcal{D}_{\mathrm{KL}}(q(\mathcal{Z})\|\, p(\mathcal{Z}|\mathcal{X})) = 0$ in Eq. 2.32. But, using EM algorithm, the variational distribution $q(\mathcal{Z}) = p(\mathcal{Z}|\mathcal{X}, \boldsymbol{\Lambda}^{(\tau)})$ is imposed to force $\mathcal{D}_{\mathrm{KL}}(q(\mathcal{Z})\|\, p(\mathcal{Z}|\mathcal{X})) = 0$ in Eq. 2.15. Figures 2.8 and 2.4(a)–(b) also show the difference between the EM and VB-EM algorithms. Such a difference is mainly because of the factorization of variational distribution that is required in VB-EM algorithm in the presence of various latent variables.

In addition to inferring the factorized variational distribution, it is crucial to address the general formula of VB-EM algorithm for a ML or MAP estimation. In this case, the model parameters in a latent variable model are merged in the variational lower bound $\widetilde{\mathcal{L}}(q(\mathcal{Z}), \boldsymbol{\Lambda})$ which is similar to Eq. 2.17 in the EM algorithm. Obviously, the variational lower bound of a VB-EM algorithm $\widetilde{\mathcal{L}}(q, \boldsymbol{\Lambda})$ is different from that of an EM algorithm $\mathcal{L}(q, \boldsymbol{\Lambda})$. To tell the difference, we denote the variational lower bound of a VB-EM inference as $\widetilde{\mathcal{L}}(\cdot)$ while the bound of an EM inference is denoted by $\mathcal{L}(\cdot)$. Similar to an EM algorithm, there are also two steps in a VB-EM algorithm. The first step (called the VB-E step) is to estimate the variational distribution of each individual latent variable Z_i at new iteration $\tau + 1$ by maximizing the variational lower bound

$$q^{(\tau+1)}(Z_i) = \underset{q(Z_i)}{\operatorname{argmax}}\, \widetilde{\mathcal{L}}(q(\mathcal{Z}), \boldsymbol{\Lambda}^{(\tau)}), \tag{2.47}$$

where $\{Z_i\}_{i=1}^{J}$. Given the updated variational distribution

$$q^{(\tau+1)}(\mathcal{Z}) = \prod_{j=1}^{J} q^{(\tau+1)}(Z_j), \tag{2.48}$$

the second step (also called the VB-M step) is to estimate the ML or MAP parameters by maximizing the new lower bound

$$\boldsymbol{\Lambda}^{(\tau+1)} = \underset{\boldsymbol{\Lambda}}{\operatorname{argmax}}\, \widetilde{\mathcal{L}}(q^{(\tau+1)}(\mathcal{Z}), \boldsymbol{\Lambda}). \tag{2.49}$$

The updating of the ML or MAP parameters $\boldsymbol{\Lambda}^{(0)} \rightarrow \boldsymbol{\Lambda}^{(1)} \rightarrow \boldsymbol{\Lambda}^{(2)} \cdots$ based on the VB-EM steps does improve the variational lower bound $\widetilde{\mathcal{L}}(q(\mathcal{Z}), \boldsymbol{\Lambda})$ or increase the corresponding likelihood function $p(\mathcal{X}|\boldsymbol{\Lambda})$ or posterior distribution $p(\boldsymbol{\Lambda}|\mathcal{X})$. Without loss of generality, the learning process using the VB-EM algorithm is here named as the variational Bayesian learning.

2.4 Sampling Methods

Exact inference in most latent variable models is intractable. As a result, it is necessary to approximate the posterior distribution of the latent variables in these models. In addition to variational Bayesian inference, Markov chain Monte Carlo [24, 25] is an alternative implementation of the full Bayesian treatment for practical solutions.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

MCMC is popular as a stochastic approximation, which is different from the deterministic approximation based on variational Bayesian (VB) as addressed in Section 2.3. It is interesting to make a comparison between VB inference and MCMC inference. In general, the approximate inference using VB theory aims to approximate the true posterior distribution based on the factorized variational distribution over a set of latent variables. VB inference is easy to scale up for large systems. However, on the other hand, MCMC inference involves the numerical computation for sampling process, which is computationally demanding. This is different from finding analytical solutions to solving the integrals and expectations in VB inference. Moreover, MCMC can freely use any distributions in mind, which provides an avenue to highly flexible models for a wide range of applications. The advanced speaker recognition systems are seen as the complicated probabilistic models with a number of latent variables.

A basic issue in MCMC inference is to calculate the expectation for a latent variable function $f(\mathcal{Z})$ by taking into account the probability distribution $p(\mathcal{Z})$. The individual latent variables in $\mathcal{Z}$ may be discrete or continuous and may contain some factors or parameters that are estimated for the construction of a probabilistic speaker recognition model. In case of continuous latent variables in $\mathcal{Z}$, the expectation function is evaluated by taking an integral

$$\mathbb{E}_{\mathcal{Z}}[f(\mathcal{Z})] = \int f(\mathcal{Z})p(\mathcal{Z})d\mathcal{Z}. \tag{2.50}$$

If $\mathcal{Z}$'s are discrete, the integral is replaced by a summation. The reason of resorting to sampling methods is that such expectations are too complicated to be evaluated analytically. That is, the sampling methods avoid the requirement of an analytical solution for Eq. 2.50. Instead, what we need is to obtain a set of samples $\{\mathcal{Z}^{(l)}, l = 1, \ldots, L\}$ that are drawn independently from the distribution $p(\mathcal{Z})$. The integral is then approximated by finding an ensemble mean of function $f(\mathbf{\Lambda})$ given by these samples $\mathcal{Z}^{(l)}$

$$\widehat{f} = \frac{1}{L}\sum_{l=1}^{L} f(\mathcal{Z}^{(l)}). \tag{2.51}$$

Because the samples $\mathcal{Z}^{(l)}$ are collected from the distribution $p(\mathcal{Z})$, the estimator $\widehat{f}$ acts as a good approximation to the true expectation in Eq. 2.50. Roughly speaking, 10–20 independent samples are sufficient to approximate the expectation function. However, in real implementation, we may not draw the samples $\{\mathcal{Z}^{(l)}\}_{l=1}^{L}$ independently. The number of effective samples may be much less than the number of total samples L. This implies that a larger sample size $L' > L$ is needed to ensure a good approximation of the true expectation.

To apply Eq. 2.51, we need to sample from $p(\mathcal{Z})$. If $p(\mathcal{Z})$ is not a simple standard distribution, drawing samples from such a distribution is usually difficult. Therefore, we may use the technique of *importance sampling* in which the samples are drawn from a *proposal distribution* $q(\mathcal{Z})$ instead of original distribution $p(\mathcal{Z})$. The idea is that sampling from proposal distribution $q(\mathcal{Z})$ is much easier than sampling from original

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

distribution $p(\mathcal{Z})$. The expectation in Eq. 2.50 is accordingly calculated by means of a finite sum over samples $\{\mathcal{Z}^{(l)}\}_{l=1}^{L}$ drawn from $q(\mathcal{Z})$

$$\begin{aligned}\mathbb{E}_{\mathcal{Z}}[f(\mathcal{Z})] &= \int f(\mathcal{Z})\frac{p(\mathcal{Z})}{q(\mathcal{Z})}q(\mathcal{Z})d\mathcal{Z} \\ &\simeq \frac{1}{L}\sum_{l=1}^{L}\frac{p(\mathcal{Z}^{(l)})}{q(\mathcal{Z}^{(l)})}f(\mathcal{Z}^{(l)}) \\ &= \frac{1}{L}\sum_{l=1}^{L} r^{(l)} f(\mathcal{Z}^{(l)}),\end{aligned} \tag{2.52}$$

where $r^{(l)} = p(\mathcal{Z}^{(l)})/q(\mathcal{Z}^{(l)})$ is the so-called importance weight. This importance weight is seen as the correction for the bias due to sampling based on the approximate proposal distribution rather than the original true distribution.

2.4.1 Markov Chain Monte Carlo

The strategy of importance sampling suffers from the severe limitation of high dimensional space when evaluating the expectation function. In this subsection, we discuss a sampling framework called the Markov chain Monte Carlo (MCMC) that not only is able to sample from a large class of distributions but also has the desirable property of scaling well with the dimension of the sample space for large applications.

In the context of latent variable models, a first-order Markov chain is employed to express the probability function in the presence of a series of latent variables $\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(\tau)}$ in different states, iterations, or learning epochs τ such that the *conditional independence*

$$p(\mathcal{Z}^{(\tau+1)}|\mathcal{Z}^{(1)}, \ldots, \mathcal{Z}^{(\tau)}) = p(\mathcal{Z}^{(\tau+1)}|\mathcal{Z}^{(\tau)}) \tag{2.53}$$

holds for different states $\tau \in \{1, \ldots, T-1\}$. Starting from the initial sample or state $p(\mathcal{Z}^{(0)})$, the Markov chain is applied and operated with the transition probability

$$T(\mathcal{Z}^{(\tau)}, \mathcal{Z}^{(\tau+1)}) \triangleq p(\mathcal{Z}^{(\tau+1)}|\mathcal{Z}^{(\tau)}). \tag{2.54}$$

If the transition probabilities $T(\mathcal{Z}^{(\tau)}, \mathcal{Z}^{(\tau+1)})$ are unchanged for all τ, a *homogeneous* Markov chain is implemented. This means that the evolution of Markov chain depends on the current state and a fixed transition matrix. For a homogeneous Markov chain with transition probability $T(\mathcal{Z}', \mathcal{Z})$, a sufficient condition for the distribution $p^{\star}(\mathcal{Z})$ to be invariant is that the following reversibility condition is met:

$$p^{\star}(\mathcal{Z}')T(\mathcal{Z}', \mathcal{Z}) = p^{\star}(\mathcal{Z})T(\mathcal{Z}, \mathcal{Z}').$$

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

Summing over all $\mathcal{Z}'$ on both size of this equation, we have

$$\begin{aligned}\sum_{\mathcal{Z}'} p^\star(\mathcal{Z}')p(\mathcal{Z}|\mathcal{Z}') &= \sum_{\mathcal{Z}'} p^\star(\mathcal{Z})p(\mathcal{Z}'|\mathcal{Z}) \\ &= p^\star(\mathcal{Z})\sum_{\mathcal{Z}'} p(\mathcal{Z}'|\mathcal{Z}) \\ &= p^\star(\mathcal{Z}),\end{aligned}$$

which enables us to write the marginal probability of $\mathcal{Z}^{(\tau+1)}$ at epoch $\tau+1$ as follows:

$$p(\mathcal{Z}^{(\tau+1)}) = \sum_{\mathcal{Z}^{(\tau)}} p(\mathcal{Z}^{(\tau+1)}|\mathcal{Z}^{(\tau)})p(\mathcal{Z}^{(\tau)}). \tag{2.55}$$

The idea of MCMC is to construct a Markov chain so that samples obtained through the chain mimic the samples drawn from the target distribution $p(\mathcal{Z})$. The chain is constructed such that it spends more time on the important regions in the sample space and that the desired distribution $p^\star(\mathcal{Z})$ is invariant. It is also necessary for the distribution $p(\mathcal{Z}^{(\tau)})$ to converge to the required invariant distribution $p^\star(\mathcal{Z})$ when $\tau \to \infty$, irrespective of the choice of initial distribution $p(\mathcal{Z}^{(0)})$. Such an invariant distribution is also called the *equilibrium* distribution.

In Section 2.2.2, we outline the general EM algorithm without stating how to evaluate the posterior distribution $p(\mathcal{Z}|\mathcal{X}, \mathbf{\Lambda})$. In case the posterior cannot be written in an analytical form, we may use MCMC to draw samples through a Markov chain and replace $\log p(\mathcal{Z}|\mathcal{X}, \mathbf{\Lambda})$ with the average log-probability obtained from the drawn samples [26, Figure. 13].

2.4.2 Gibbs Sampling

Gibbs sampling [25, 27] is a realization of MCMC algorithm that has been widely applied for generating a sequence of observations that follow a specified distribution. This realization is seen as a special variant of Metropolis–Hastings algorithm [28]. Suppose we are given the distribution of a set of latent variables $p(\mathcal{Z}) = p(Z_1, \ldots, Z_J)$ and the distribution of initial state of a Markov chain $p(\mathcal{Z}^{(0)})$. In each step of sampling procedure, an arbitrary latent variable, say Z_i, has a value drawn from $p(Z_i|\mathcal{Z}_{\setminus i})$, where $\mathcal{Z}_{\setminus i} = \{Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_J\}$. We repeat the sampling procedure where different variables in Z are cycled in random or in a specific order. The distribution $p(\mathcal{Z})$ is sampled in a Gibbs sampling procedure. This distribution is invariant at each learning epoch or in the whole Markov chain. Correspondingly, the marginal distribution $p(\mathcal{Z}_{\setminus i})$ is invariant and the conditional distribution $p(Z_i|\mathcal{Z}_{\setminus i})$ is corrected at each sampling step i. The Gibbs sampling for a latent variable model with J variables running by T steps is performed according to the following algorithm

- Initialize $\{Z_i^{(1)} : i = 1, \ldots, J\}$
- For $\tau = 1, \ldots, T$:
  - Sample $Z_1^{(\tau+1)} \sim p(Z_1|Z_2^{(\tau)}, Z_3^{(\tau)}, \ldots, Z_J^{(\tau)})$.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

– Sample $Z_2^{(\tau+1)} \sim p(Z_2|Z_1^{(\tau+1)}, Z_3^{(\tau)}, \ldots, Z_J^{(\tau)})$.

$\vdots$

– Sample $Z_i^{(\tau+1)} \sim p(Z_i|Z_1^{(\tau+1)}, \ldots, Z_{i-1}^{(\tau+1)}, Z_{i+1}^{(\tau)}, \ldots, Z_J^{(\tau)})$.

$\vdots$

– Sample $Z_J^{(\tau+1)} \sim p(Z_J|Z_1^{(\tau+1)}, Z_2^{(\tau+1)}, \ldots, Z_{J-1}^{(\tau+1)})$.

We can adopt the Metropolis–Hastings algorithm to sample a target variable Z_i by fixing the rest of variables $\mathcal{Z}_{\setminus i}$ and adopting the transition probability from $\mathcal{Z}$ to $\mathcal{Z}^\star$ in Gibbs sampling, which is obtained by

$$q_i(\mathcal{Z}^\star|\mathcal{Z}) = p(Z_i^\star|\mathcal{Z}_{\setminus i}). \tag{2.56}$$

It is because that the rest of variables is unchanged by a sampling step, then $(\mathcal{Z}^\star)_{\setminus i} = \mathcal{Z}_{\setminus i}$. According to the property $p(\mathcal{Z}) = p(Z_i|\mathcal{Z}_{\setminus i})p(\mathcal{Z}_{\setminus i})$, we can determine the acceptance probability in Metropolis–Hastings algorithm obtained by

$$\begin{aligned} A_i(\mathcal{Z}^\star, \mathcal{Z}) &= \frac{p(\mathcal{Z}^\star)q_i(\mathcal{Z}|\mathcal{Z}^\star)}{p(\mathcal{Z})q_i(\mathcal{Z}^\star|\mathcal{Z})} \\ &= \frac{p(Z_i^\star|(\mathcal{Z}^\star)_{\setminus i})p((\mathcal{Z}^\star)_{\setminus i})p(Z_i|(\mathcal{Z}^\star)_{\setminus i})}{p(Z_i|\mathcal{Z}_{\setminus i})p(\mathcal{Z}_{\setminus i})p(Z_i^\star|\mathcal{Z}_{\setminus i})} = 1, \end{aligned} \tag{2.57}$$

which means that we can always accept the steps of the Gibbs sampling in the Metropolis–Hastings algorithm. Eq. 2.57 is obtained by applying $(\mathcal{Z}^\star)_{\setminus i} = \mathcal{Z}_{\setminus i}$.

2.5 Bayesian Learning

A challenge in machine learning and big data analytics is that data are collected from heterogeneous environments or under adverse conditions. The collected data are usually noisy, non-labeled, non-aligned, mismatched, and ill-posed. Bayesian inference and statistics play an essential role in dealing with uncertainty modeling and probabilistic representation in machine learning, including classification, regression, supervised learning, and unsupervised learning, to name a few. However, probabilistic models may be improperly assumed, overestimated, or underestimated. It is important to address the issue of model regularization in construction of speaker recognition system. Bayesian solutions to model regularization in different information systems have been popular in the last decade. Sections 2.3 and 2.4 have surveyed the approximate Bayesian inference and learning based on variational inference and sampling methods, respectively. This section started with a general description of model regularization (Section 2.5.1) and then presented how Bayesian learning is developed to tackle the regularization issue in speaker recognition (Section 2.5.2). In general, model regularization is known as a process that incorporates additional information to reflect model uncertainty or randomness that can deal with the ill-posed or overfitting problem. The uncertainty or randomness can be characterized by a prior distribution $p(\mathbf{\Lambda})$ or even a prior process for finding a drawing process for prior distributions [29]. The prior process can be Dirichlet

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

process, beta process, Gaussian process, etc., which is the distribution over distribution functions [30].

2.5.1 Model Regularization

Speaker recognition models based on GMM, PLDA, SVM, and DNN are basically learned or adapted from a collection of training utterances or an online test utterance with learning strategy either in supervised mode or in unsupervised or semi-supervised mode. The goal of building a learning machine for speaker recognition is to learn a regularized model that is robust and can sufficiently generalize to recognize an unheard speaker utterance. Our ultimate goal is to achieve a desirable prediction performance under a learning machine. It is inevitable that the overfitting or underfitting problem may happen because the assumed model may not faithfully reflect the conditions in speaker utterances. The number of training utterances may be too large or too small. It is important to characterize the stochastic property in speaker utterances $\mathcal{X}$, in model assumption, and in parameter estimation during the construction of speaker model.

We therefore need a tool or framework to sufficiently represent or factorize speech signal, and faithfully characterize the mapping between collected data and assumed model. This tool should be flexible to probe speech signals and understand the production of real-world speaker utterances in the presence of various scenarios and environments. Using this tool, a scalable model topology can be learned or adapted with different size of training utterances. The learned representation can be robust under the mismatched or ill-posed conditions in training and test utterances. The noise interference and channel mismatch can be accommodated in the model to implement a noise-robust and channel-aware speaker recognition system. We need a general tool or theory to analyze, represent, and understand the speaker utterances in adverse environments. This tool can handle the issue of model selection and provide an efficient calculation or selection for optimal hyperparameter or regularization parameters for an assumed model without requirement of additional validation data. The optimization does not only consider the likelihood function or the goodness-of-fit measure but also the regularization term for model penalty. Accordingly, the modeling tool based on Bayesian learning is adopted. Occam's razor [31] is taken into account for model construction and hyperparameter selection. Bayesian theory provides a complete solution, platform, tool, or infrastructure to cope with various issues in model regularization. In what follows, we summarize how Bayesian learning can work for speaker recognition [30]

- tackle the overfitting or underfitting problem
- characterize the randomness for probabilistic model construction
- achieve the robust model under mismatched and ill-posed environments
- choose the model topology and hyperparameters
- be insensitive to noise contamination and microphone variations
- be scalable for large systems or applications
- be adaptive, flexible, and autonomous

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press

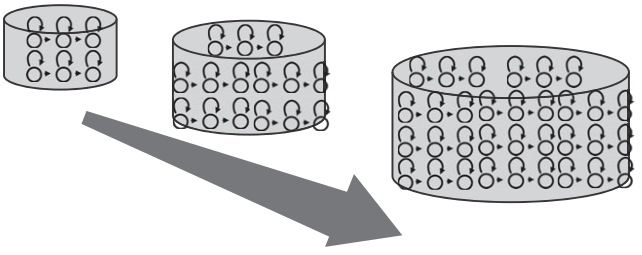

Figure 2.9 Illustration of a Bayesian speaker model where the model complexity is scalable by the amount of training utterances.

Figure 2.9 shows the concept that the Bayesian speaker model is flexible and scalable in the presence of changing amounts of speaker utterances.

2.5.2 Bayesian Speaker Recognition

The real-world speaker recognition encounters a number of challenging issues that could be tackled by Bayesian approaches. First, the learning processes for speaker recognition are usually operated in an unsupervised fashion. The target information is basically none or very little during the training phase. The prediction performance under unsupervised learning for unknown test utterances is sometimes unreliable and difficult. Second, the number of factors or Gaussian components is unknown in factor analysis models or Gaussian mixture models. Determining the model size is an unsupervised learning problem, and the ability to determine the model size from heterogeneous data is crucial because a suitable model structure provides meaningful and reliable predictions for future data. Too complicated models will be unstable or over-expressive with weak generalization. Third, how to build a balanced solution or how to tune a tradeoff between overdetermined and under-determined systems is a critical issue in speaker recognition. Fourth, the enrollment utterances in speaker recognition system are usually very short. It is challenging to verify the right speaker using limited data. In addition, speaker recognition performance is substantially affected by the inter- and intra-speaker variabilities. The robustness to these variabilities is essential to achieve desirable performance. Finally, the variabilities in channels and noise are also severe in real-world speaker recognition. How to build a representative model for these variabilities is important.

To deal with the abovementioned issues, Bayesian perspectives and Bayesian methods are introduced and implemented. From this perspective, we need to explore the latent space containing latent variables to characterize speaker identities. Bayesian learning plays a natural role to carry out latent variable models for speaker recognition. Based on Bayesian approaches, the model regularization issue is handled and the uncertainty modeling is performed. Approximate Bayesian inference methods are feasible to realize these ideas. Using these ideas, the prediction performance on future data will be improved. The robustness of the resulting model will be enhanced.

https://doi.org/10.1017/9781108552332.003 Published online by Cambridge University Press