3 Machine Learning Models

This chapter provides detailed descriptions and derivations of various machine learning models for speaker verification. Instead of discussing these models in chronological order, we will start from the fundamental models, such as Gaussian mixture models and support vector machines and progressively move to the more complicated ones that are built on these fundamental models. We will leave the recent development in DNN-based models to the next two chapters.

3.1 Gaussian Mixture Models

A Gaussian mixture model (GMM) is a weighted sum of Gaussian distributions. In the context of speech and speaker recognition, a GMM can be used to represent the distribution of acoustic vectors $\mathbf{o}$'s:[1]

$$p(\mathbf{o}|\Lambda) = \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \tag{3.1}$$

where $\Lambda = \{\pi_c, \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c\}_{c=1}^{C}$ contains the parameters of the GMM. In Eq. 3.1, π_c, $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ are the weight, mean vector, and covariance matrix of the cth mixture whose density has the form

$$\mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) = \frac{1}{(2\pi)^{\frac{D}{2}} |\boldsymbol{\Sigma}_c|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{o}-\boldsymbol{\mu}_c)^{\mathsf{T}} \boldsymbol{\Sigma}_c^{-1} (\mathbf{o}-\boldsymbol{\mu}_c)\right\}.$$

The mixture weight π_c's can be considered as the priors of the mixtures and they should sum to 1.0, i.e., $\sum_{c=1}^{C} \pi_c = 1$. Figure 3.1 shows an example of a one-dimensional GMM with three mixtures.

To estimate Λ, we express the log-likelihood of a set of training data in terms of the parameters and find the parameters in Λ that lead to the maximum-likelihood of the training data. Specifically, given a set of T independent and identically distributed (iid) acoustic vectors $\mathcal{O} = \{\mathbf{o}_t; t = 1, \ldots, T\}$, the log-likelihood function (function of Λ) is given by

[1] In this book, we use lower case boldface letters to denote random vectors. Depending on the context, lower-case boldface letters can also represent the realization (i.e., observations) of the corresponding random vectors. Note that this is different from the usual probabilistic convention that random variables are represented by capital letters (e.g., X) and their realization (x), as in $P(X < x)$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

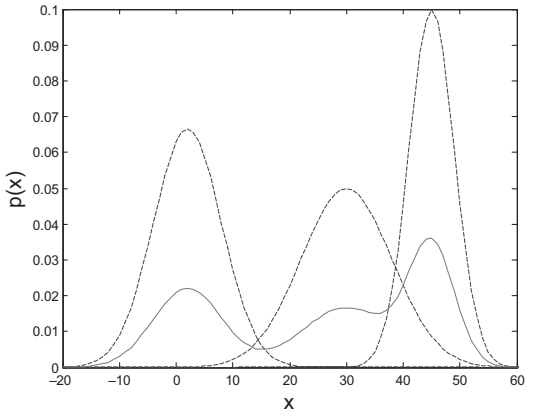


Figure 3.1 A one-D GMM with three mixture components. [Adapted from *Bayesian Speech and Language Processing (Figure 3.2), by S. Watanabe and J.T. Chien, 2015, Cambridge University Press.*]

$$\log p(\mathcal{O}|\Lambda) = \log \left\{ \prod_{t=1}^{T} \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \right\} = \sum_{t=1}^{T} \log \left\{ \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c) \right\}.$$

To find Λ that maximizes $\log p(\mathcal{O}|\Lambda)$, we may set $\frac{\partial \log p(\mathcal{O})}{\partial \Lambda} = 0$ and solve for Λ. But this method will not give a closed-form solution for Λ. The trouble is that the summation appears inside the logarithm. The problem, however, can be readily solved by using the EM algorithm to be explained next.

3.1.1 The EM Algorithm

The EM algorithm (see Section 2.2.2) is a popular iterative method for estimating the parameters of statistical models. It is also an elegant method for finding maximum-likelihood solutions for models with latent variables, such as GMMs. This is achieved by maximizing a log-likelihood function of the model parameters through two iterative steps: **E**xpectation and **M**aximization.

Denote the acoustic vectors from a large population as $\mathcal{O} = \{\mathbf{o}_t; t = 1, \ldots, T\}$. In GMM, for each data point $\mathbf{o}_t$, we do not know which Gaussian generates it. Therefore, the latent information is the Gaussian identity for each $\mathbf{o}_t$. Define

$$\mathcal{L} = \{\ell_{tc};\ t = 1, \ldots, T \text{ and } c = 1, \ldots, C\}$$

as the set of latent variables, where $\ell_{tc} = 1$ if $\mathbf{o}_t$ is generated by the cth Gaussian; otherwise $\ell_{tc} = 0$. Figure 3.2 shows a graphical representation of a GMM. This graphical model suggests that $\boldsymbol{o}_t$ depends on $\boldsymbol{\ell}_t = [\ell_{t1} \ \ldots \ \ell_{tC}]^{\mathsf{T}}$ so that the marginal likelihood $p(\mathbf{o}_t)$ can be obtained by marginalizing out the latent variable $\boldsymbol{\ell}_t$:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

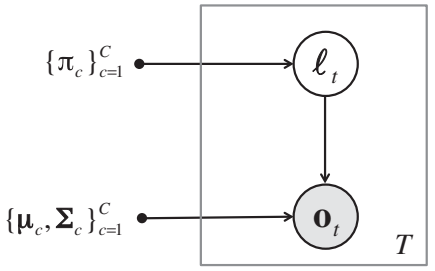


Figure 3.2 Graphical representation of a GMM. $\boldsymbol{\ell}_t = [\ell_{t1} \ \ldots \ \ell_{tC}]^\mathsf{T}$ and $\mathbf{o}_t$, with $t = 1, \ldots, T$, are the latent variables and observed vectors, respectively.

$$\begin{aligned} p(\mathbf{o}_t) &= \sum_{\boldsymbol{\ell}_t} p(\boldsymbol{\ell}_t) p(\mathbf{o}_t|\boldsymbol{\ell}_t) \\ &= \sum_{c=1}^{C} P(\ell_{tc} = 1) p(\mathbf{o}_t|\ell_{tc} = 1) \\ &= \sum_{c=1}^{C} \pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \end{aligned} \tag{3.2}$$

which has the same form as Eq. 3.1.

In the EM literatures, $\{\mathcal{O}, \mathcal{L}\}$ is called the *complete* data set, and $\mathcal{O}$ is the *incomplete* data set. In EM, we maximize $\log p(\mathcal{O}, \mathcal{L}|\Lambda)$ with respect to Λ instead of maximizing $\log p(\mathcal{O}|\Lambda)$. The important point is that maximizing the former is much more straightforward and will lead to closed-form solutions for each EM iteration, as will be shown below.

However, we actually do not know $\mathcal{L}$. So, we could not compute $\log p(\mathcal{O}, \mathcal{L}|\Lambda)$. Fortunately, we know its posterior distribution, i.e., $P(\mathcal{L}|\mathcal{O}, \Lambda)$, through the Bayes theorem. Specifically, for each $\mathbf{o}_t$, we compute the posterior probability:[2]

$$\begin{aligned} \gamma(\ell_{tc}) &\equiv P(\ell_{tc} = 1|\mathbf{o}_t, \Lambda) \\ &= \frac{P(\ell_{tc} = 1|\Lambda) p(\mathbf{o}_t|\ell_{tc} = 1, \Lambda)}{p(\mathbf{o}_t|\Lambda)} \\ &= \frac{\pi_c \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)}{\sum_{j=1}^{C} \pi_j \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \end{aligned} \tag{3.3}$$

In the speech and speaker recognition literatures, computing the posterior probabilities of mixture components is called *alignment*. Its aim is to determine how close a vector $\mathbf{o}_t$ is to the individual Gaussians, accounting for both the priors, means and covariances of the mixture components. Figure 3.3 illustrates the alignment process.

With the posteriors $\gamma(\ell_{tc})$, given the current estimate of the model parameters Λ^{old}, we can find its new estimate Λ by computing the expected value of $\log p(\mathcal{O}, \mathcal{L}|\Lambda)$ under

[2] We denote probabilities and probability mass functions of discrete random variables using capital letter P, and we denote the likelihoods and probability density functions of continuous random variables using lower case letter p.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

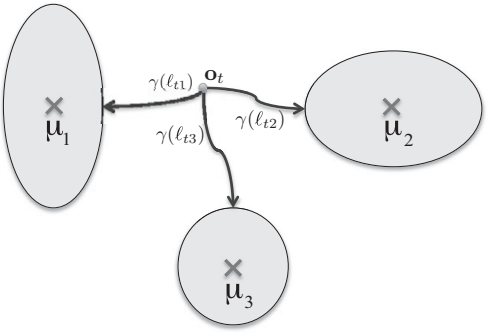


Figure 3.3 Aligning an acoustic vector $\mathbf{o}_t$ to a GMM with three mixture components.

the posterior distribution of $\mathcal{L}$:

$$\begin{aligned} Q(\Lambda|\Lambda^{\text{old}}) &= \mathbb{E}_{\mathcal{L}}\{\log p(\mathcal{O},\mathcal{L}|\Lambda)|\mathcal{O},\Lambda^{\text{old}}\} \\ &= \sum_{t=1}^{T}\sum_{c=1}^{C} P(\ell_{tc}=1|\mathbf{o}_t,\Lambda^{\text{old}})\log p(\mathbf{o}_t,\ell_{tc}=1|\Lambda) \\ &= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\log p(\mathbf{o}_t,\ell_{tc}=1|\Lambda) \\ &= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\log p(\mathbf{o}_t|\ell_{tc}=1,\Lambda)P(\ell_{tc}=1|\Lambda) \\ &= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\log\left[\mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c)\pi_c\right], \end{aligned} \tag{3.4}$$

where $Q(\Lambda|\Lambda^{\text{old}})$ is called the auxiliary function or simply Q-function (see Eq. 2.6). The E-step consists in computing $\gamma(\ell_{tc})$ for all training samples so that $Q(\Lambda|\Lambda^{\text{old}})$ can be expressed as a function of $\boldsymbol{\mu}_c$, $\boldsymbol{\Sigma}_c$, and π_c for $c = 1,\ldots,C$.

Then, in the M-step, we maximize $Q(\Lambda|\Lambda^{\text{old}})$ with respect to Λ by setting $\frac{\partial Q(\Lambda|\Lambda^{\text{old}})}{\partial\Lambda} = 0$ to obtain (see [32, Ch. 3]):

$$\boldsymbol{\mu}_c = \frac{\sum_{t=1}^{T}\gamma(\ell_{tc})\mathbf{o}_t}{\sum_{t=1}^{T}\gamma(\ell_{tc})} \tag{3.5a}$$

$$\boldsymbol{\Sigma}_c = \frac{\sum_{t=1}^{T}\gamma(\ell_{tc})(\mathbf{o}_t-\boldsymbol{\mu}_c)(\mathbf{o}_t-\boldsymbol{\mu}_c)^{\mathsf{T}}}{\sum_{t=1}^{T}\gamma(\ell_{tc})} \tag{3.5b}$$

$$\pi_c = \frac{1}{T}\sum_{t=1}^{T}\gamma(\ell_{tc}), \tag{3.5c}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $c = 1, \ldots, C$. Eq. 3.5a–Eq. 3.5c constitute the M-step of the EM algorithm. In practical implementation of the M-step, we compute the sufficient statistics:

$$n_c = \sum_{t=1}^{T} \gamma(\ell_{tc}) \tag{3.6a}$$

$$\mathbf{f}_c = \sum_{t=1}^{T} \gamma(\ell_{tc}) \mathbf{o}_t \tag{3.6b}$$

$$\mathbf{S}_c = \sum_{t=1}^{T} \gamma(\ell_{tc}) \mathbf{o}_t \mathbf{o}_t^{\mathsf{T}}, \tag{3.6c}$$

where $c = 1, \ldots, C$. Then, Eq. 3.5a–Eq. 3.5c become

$$\boldsymbol{\mu}_c = \frac{1}{n_c} \mathbf{f}_c \tag{3.7a}$$

$$\boldsymbol{\Sigma}_c = \frac{1}{n_c} \mathbf{S}_c - \boldsymbol{\mu}_c \boldsymbol{\mu}_c^{\mathsf{T}} \tag{3.7b}$$

$$\pi_c = \frac{1}{T} n_c. \tag{3.7c}$$

In summary, the EM algorithm iteratively performs the E- and M-steps until $Q(\Lambda|\Lambda^{\text{old}})$ no longer increases.

- **Initialization**: Randomly select C samples from $\mathcal{O}$ and assign them to $\{\boldsymbol{\mu}_c\}_{c=1}^{C}$; Set $\pi_c = \frac{1}{C}$ and $\boldsymbol{\Sigma}_c = \mathbf{I}$, where $c = 1, \ldots, C$.
- **E-Step**: Find the distribution of the latent (unobserved) variables, given the observed data and the current estimate of the parameters;
- **M-Step**: Re-estimate the parameters to maximize the likelihood of the observed data, under the assumption that the distribution found in the E-step is correct.

The iterative process guarantees to increases the true likelihood or leaves it unchanged (if a local maximum has already been reached).

3.1.2 Universal Background Models

If we use the speech of a large number of speakers to train a GMM using Eq. 3.3 and Eqs. 3.5a–3.5c, we obtain a universal background model (UBM), Λ^{ubm}, with density function

$$p(\mathbf{o}|\Lambda^{\text{ubm}}) = \sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}). \tag{3.8}$$

Specifically, the UBM is obtained by iteratively carried out the E- and M-steps as follows.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

- E-step: Compute the conditional distribution of mixture components:

$$\gamma(\ell_{tc}) \equiv \Pr(\text{Mixture} = c|\mathbf{o}_t) = \frac{\pi_c^{\text{ubm}}\mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})}{\sum_{c=1}^{C}\pi_c^{\text{ubm}}\mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})} \tag{3.9}$$

  where $c = 1, \ldots, C$.
- M-step: Update the model parameters:
  - Mixture weights: $\pi_c^{\text{ubm}} = \frac{1}{T}\sum_{t=1}^{T}\gamma(\ell_{tc})$
  - Mean vectors: $\boldsymbol{\mu}_c^{\text{ubm}} = \frac{\sum_{t=1}^{T}\gamma(\ell_{tc})\mathbf{o}_t}{\sum_{t=1}^{T}\gamma(\ell_{tc})}$
  - Covariance matrices: $\boldsymbol{\Sigma}_c^{\text{ubm}} = \frac{\sum_{t=1}^{T}\gamma(\ell_{tc})\mathbf{o}_t\mathbf{o}_t^T}{\sum_{t=1}^{T}\gamma(\ell_{tc})} - \boldsymbol{\mu}_c^{\text{ubm}}(\boldsymbol{\mu}_c^{\text{ubm}})^{\mathsf{T}}$

  where $c = 1, \ldots, C$.

In a GMM–UBM system, each of the genuine speakers (also called target speakers) has his/her own GMM, and the UBM serves as a reference for comparing likelihood ratios. More precisely, if the likelihood of a test utterance with respect to a genuine-speaker's GMM is larger than its likelihood with respect to the UBM, there is a high chance that the test utterance is spoken by the genuine speaker.

3.1.3 MAP Adaptation

If each target speaker has a large number of utterances for training his/her GMM, the EM algorithm in Section 3.1 can be directly applied. However, in practice, the amount of speech for each speaker is usually small. As a result, directly applying the EM algorithm will easily cause overfitting. A better solution is to apply the maximum *a posteriori* (MAP) adaptation in which target-speaker models are adapted from the UBM.

The MAP algorithm finds the parameters of target-speaker's GMM given UBM parameters $\Lambda^{\text{ubm}} = \{\pi_c^{\text{ubm}}, \boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}\}_{c=1}^{C}$ using the EM algorithm. The objective is to estimate the mode of the posterior of the model parameters:

$$\begin{aligned}\Lambda^{\text{map}} &= \underset{\Lambda}{\text{argmax}}\, p(\Lambda|\mathcal{O}) \\ &= \underset{\Lambda}{\text{argmax}}\, p(\mathcal{O}|\Lambda)p(\Lambda) \\ &= \underset{\Lambda}{\text{argmax}} \prod_{t=1}^{T} p(\mathbf{o}_t|\Lambda)p(\Lambda).\end{aligned} \tag{3.10}$$

Direct optimization of Eq. 3.10 is difficult. However, we may iteratively find the optimal solution via the EM algorithm. Similar to the EM algorithm for GMM in Section 3.1, instead of maximizing Eq. 3.10, we maximize the auxiliary function:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$
\begin{aligned}
Q(\Lambda|\Lambda^{\text{old}}) &= \mathbb{E}_{\mathcal{L}}\left\{\log\left[p(\mathcal{O},\mathcal{L}|\Lambda)p(\Lambda)\right]|\mathcal{O},\Lambda^{\text{old}}\right\} \\
&= \mathbb{E}_{\mathcal{L}}\left\{\log p(\mathcal{O},\mathcal{L}|\Lambda)|\mathcal{O},\Lambda^{\text{old}}\right\} + \mathbb{E}_{\mathcal{L}}\left\{\log p(\Lambda)|\mathcal{O},\Lambda^{\text{old}}\right\} \\
&= \sum_{t=1}^{T}\sum_{c=1}^{C} P(\ell_{tc}=1|\mathbf{o}_t,\Lambda^{\text{old}})\log p(\mathbf{o}_t,\ell_{tc}=1|\Lambda) + \log p(\Lambda) \\
&= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\log p(\mathbf{o}_t,\ell_{tc}=1|\Lambda) + \log p(\Lambda) \\
&= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\left[\log p(\mathbf{o}_t|\ell_{tc}=1,\Lambda) + \log P(\ell_{tc}=1|\Lambda)\right] + \log p(\Lambda) \\
&= \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\left[\log \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c) + \log \pi_c\right] + \log p(\Lambda), \qquad (3.11)
\end{aligned}
$$

If only the mean vectors are adapted, we have $\pi_c = \pi_c^{\text{ubm}}$ and $\boldsymbol{\Sigma}_c = \boldsymbol{\Sigma}_c^{\text{ubm}}$. Also, we only need to assume a prior over $\boldsymbol{\mu}_c$:

$$
p(\boldsymbol{\mu}_c) = \mathcal{N}(\boldsymbol{\mu}_c|\boldsymbol{\mu}_c^{\text{ubm}}, r^{-1}\boldsymbol{\Sigma}_c^{\text{ubm}}), \qquad (3.12)
$$

where r is called the relevant factor [7]. Figure 3.4 shows the graphical model of the Bayesian GMM when only the mean vectors $\{\boldsymbol{\mu}_c\}_{c=1}^{C}$ of the GMM are assumed random with prior distribution given by Eq. 3.12.

Dropping terms independent of $\boldsymbol{\mu}_c$, Eq. 3.11 can be written as:

$$
Q(\boldsymbol{\mu}|\boldsymbol{\mu}^{\text{ubm}}) = \sum_{t=1}^{T}\sum_{c=1}^{C} \gamma(\ell_{tc})\log \mathcal{N}(\mathbf{o}_t|\boldsymbol{\mu}_c,\boldsymbol{\Sigma}_c^{\text{ubm}}) + \log \mathcal{N}(\boldsymbol{\mu}_c|\boldsymbol{\mu}_c^{\text{ubm}}, r^{-1}\boldsymbol{\Sigma}_c^{\text{ubm}}). \quad (3.13)
$$

where $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^{\mathsf{T}} \ \ldots \ \boldsymbol{\mu}_C^{\mathsf{T}}]^{\mathsf{T}}$.

The E-step is similar to the E-step in training GMMs. Specifically, given T_s acoustic vectors $\mathcal{O}^{(s)} = \{\mathbf{o}_1,\ldots,\mathbf{o}_{T_s}\}$ from speaker s, we compute the sufficient statistics:

$$
n_c = \sum_{t=1}^{T_s}\gamma(\ell_{tc}) \quad \text{and} \quad E_c(\mathcal{O}^{(s)}) = \frac{1}{n_c}\sum_{t=1}^{T_s}\gamma(\ell_{tc})\mathbf{o}_t, \qquad (3.14)
$$

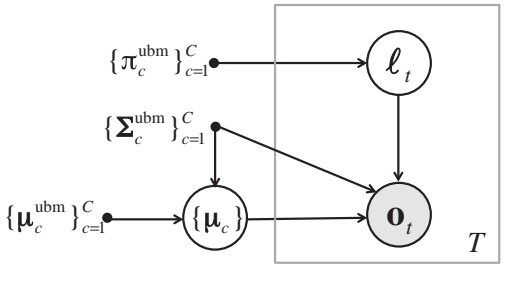


Figure 3.4 Graphical model of Bayesian GMMs with prior over the GMM's mean vectors given by Eq. 3.12.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\gamma(\ell_{tc})$ is computed as in Eq. 3.3. In the M-step, we differentiate $Q(\boldsymbol{\mu}|\boldsymbol{\mu}^{\text{old}})$ in Eq. 3.13 with respect to $\boldsymbol{\mu}_c$, which gives

$$\frac{\partial Q(\boldsymbol{\mu}|\boldsymbol{\mu}^{\text{ubm}})}{\partial \boldsymbol{\mu}_c} = \sum_{t=1}^{T_s} \gamma(\ell_{tc})(\boldsymbol{\Sigma}_c^{\text{ubm}})^{-1}(\mathbf{o}_t - \boldsymbol{\mu}_c) - r(\boldsymbol{\Sigma}_c^{\text{ubm}})^{-1}(\boldsymbol{\mu}_c - \boldsymbol{\mu}_c^{\text{ubm}}). \tag{3.15}$$

By setting Eq. 3.15 to $\mathbf{0}$, we obtain the adapted mean

$$\begin{aligned} \boldsymbol{\mu}_c &= \frac{\sum_t \gamma(\ell_{tc})\mathbf{o}_t}{\sum_t \gamma(\ell_{tc}) + r} + \frac{r\boldsymbol{\mu}_c^{\text{ubm}}}{\sum_t \gamma(\ell_{tc}) + r} \\ &= \alpha_c E_c(\mathcal{O}^{(s)}) + (1 - \alpha_c)\boldsymbol{\mu}_c^{\text{ubm}}, \end{aligned} \tag{3.16}$$

where

$$\alpha_c = \frac{n_c}{n_c + r}. \tag{3.17}$$

The relevance factor r is typically set to 16. Figure 3.5 shows the MAP adaptation process and Figure 3.6 illustrates a two-dimensional examples of the adaptation process. Note that in practice only the mean vectors will be adapted.

According to Eq. 3.14 and Eq. 3.17, $\alpha_c \to 1$ when $\mathcal{O}^{(s)}$ comprises lots of vectors (long utterances) and $\alpha_c \to 0$ otherwise. This means that $\boldsymbol{\mu}_c^{(s)}$ will be closed to the observed vectors from Speaker s when the utterance is long and will be similar to the cth Gaussian of the UBM when not many frames are aligned to the cth mixture. This property agrees with the Bayesian philosophy.

In Eq. 3.17, r determines the minimum number of frames aligning to mixture c to have the adaptation effect on $\boldsymbol{\mu}_c^{(s)}$. Specifically, when r is very small, say $r = 1$, a very small number of frames aligning to mixture c will be enough to cause the mean vector $\boldsymbol{\mu}_c^{(s)}$ to be adapted to $\mathcal{O}^{(s)}$. On the other hand, when r is large, say $r = 30$, a lot more frames aligning to mixture c are required to have any adaptation effect.

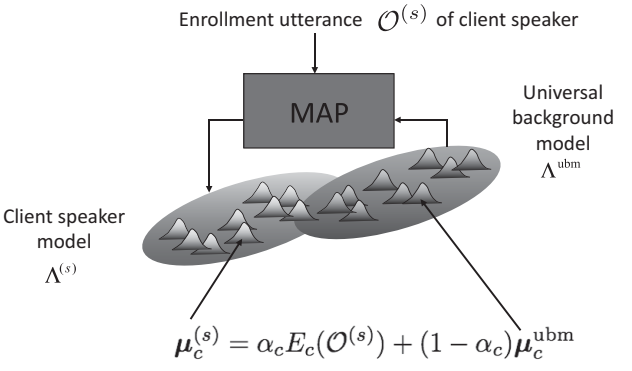


Figure 3.5 The MAP adaptation process.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

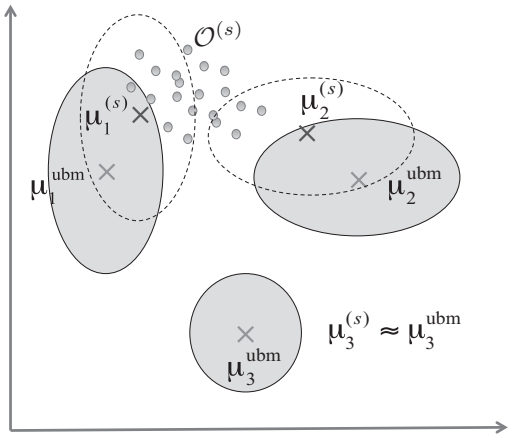


Figure 3.6 A two-dimensional example illustrating the adaptation of mixture components in a UBM. $\boldsymbol{\mu}_1^{\text{ubm}}$ and $\boldsymbol{\mu}_2^{\text{ubm}}$ will move toward the speaker-dependent samples $\mathcal{O}^{(s)}$ as the samples are close enough to these two Gaussians, whereas $\boldsymbol{\mu}_3^{\text{ubm}}$ will remain unchanged because the samples are too far away from it.

3.1.4 GMM–UBM Scoring

Given the acoustic vectors $\mathcal{O}^{(t)}$ from a test speaker and a claimed identity s, speaker verification can be formulated as a two-class hypothesis problem:

- H_0: $\mathcal{O}^{(t)}$ comes from the true speaker s
- H_1: $\mathcal{O}^{(t)}$ comes from an impostor

Verification score is a log-likelihood ratio:

$$S_{\text{GMM–UBM}}(\mathcal{O}^{(t)}|\Lambda^{(s)}, \Lambda^{\text{ubm}}) = \log p(\mathcal{O}^{(t)}|\Lambda^{(s)}) - \log p(\mathcal{O}^{(t)}|\Lambda^{\text{ubm}}), \tag{3.18}$$

where $\log p(\mathcal{O}^{(t)}|\Lambda^{(s)})$ is the log-likelihood of $\mathcal{O}^{(t)}$ given the speaker model $\Lambda^{(s)}$, which is given by

$$\log p(\mathcal{O}^{(t)}|\Lambda^{(s)}) = \sum_{\mathbf{o}\in\mathcal{O}^{(t)}} \log \sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{\text{ubm}}). \tag{3.19}$$

Note that only the mean vectors in the speaker model are speaker-dependent, i.e., $\Lambda^{(s)} = \{\pi_c^{\text{ubm}}, \boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{\text{ubm}}\}_{c=1}^{C}$. This is because only the means are adapted in practice. Figure 3.5 shows the scoring process.

A side benefit of MAP adaptation is that it keeps the correspondence between the mixture components of the target-speaker model $\Lambda^{(s)}$ and the UBM. This property allows for fast scoring when the GMM and the UBM cover a large region of the feature space so that only a few Gaussians contribute to the likelihood value for each acoustic vector $\mathbf{o}$. Figure 3.6 illustrates such situation in which the contribute of the third Gaussian (with mean $\boldsymbol{\mu}_3^{\text{ubm}}$) can be ignored for any test vectors far away from it. Therefore, we may express the log-likelihood ratio of a test utterance with acoustic vectors $\mathcal{O}^{(t)}$ as

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$
\begin{aligned}
&S_{\text{GMM–UBM}}(\mathcal{O}^{(t)}|\Lambda^{(s)}, \Lambda^{\text{ubm}}) = \log p(\mathcal{O}^{(t)}|\Lambda^{(s)}) - \log p(\mathcal{O}^{(t)}|\Lambda^{\text{ubm}}) \\
&= \sum_{\mathbf{o}\in\mathcal{O}^{(t)}} \left[\log \sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{\text{ubm}}) - \log \sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}) \right] \\
&\approx \sum_{\mathbf{o}\in\mathcal{O}^{(t)}} \left[\log \sum_{c\in\Omega} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{\text{ubm}}) - \log \sum_{c\in\Omega} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}) \right], \quad (3.20)
\end{aligned}
$$

where Ω is a set of indexes corresponding to the C' largest likelihoods in $\{\pi_c^{\text{ubm}} \mathcal{N}(\mathbf{o}|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})\}_{c=1}^{C}$. Typically, $C' = 5$. The third equation in Eq. 3.20 requires $C + C'$ Gaussian evaluations for each $\mathbf{o}$, whereas the second equation requires $2C$ Gaussian evaluations. When $C = 1024$ and $C' = 5$, substantial computation saving can be achieved.

3.2 Gaussian Mixture Model–Support Vector Machines

A drawback of GMM–UBM systems is that the GMMs and UBM are trained separately, which means that information that discriminates the target speakers from the background speakers cannot be fully utilized. In 2006, Campbell [14] proposed to turn the GMMs into vectors so that target-speakers' GMMs and background-speakers' GMMs can be treated as positive- and negative-class vectors for training discriminative classifiers, one for each target speaker. Because of the high dimensionality of the resulting vectors, linear support vector machines are a natural choice.

3.2.1 Support Vector Machines

To understand the concepts of support vector machines (SVMs), we need to ask ourself "what is a good decision boundary?" Consider a two-class linearly separable classification problem shown in Figure 3.7. There are plenty of methods to find a boundary that can separate the two classes. A naive way is to find a line that is perpendicular to the line joining the two centers and is of equal distance to the centers, as shown in Figure 3.7. Setting the boundary position based on the centers of the two classes means that all samples are considered equally important. But is the boundary in Figure 3.7 good? Obviously it is not.

In 1995, Vapnik [33] advocated finding a subset of vectors that are more relevant to the classification task and using these vectors to define the decision boundary. Moreover, the decision boundary should be as far away from the data of both classes as possible (of course, not infinitely far). These two criteria can be fulfilled by maximizing the margin shown in Figure 3.8. In the figure, relevant vectors highlighted by the big circles are called support vectors, which give rise to the name support vector machines.

Linearly Separable Problems

In Figure 3.8, the decision boundary is given by

$$\mathbf{w}^{\mathsf{T}}\mathbf{x} + b = 0.$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

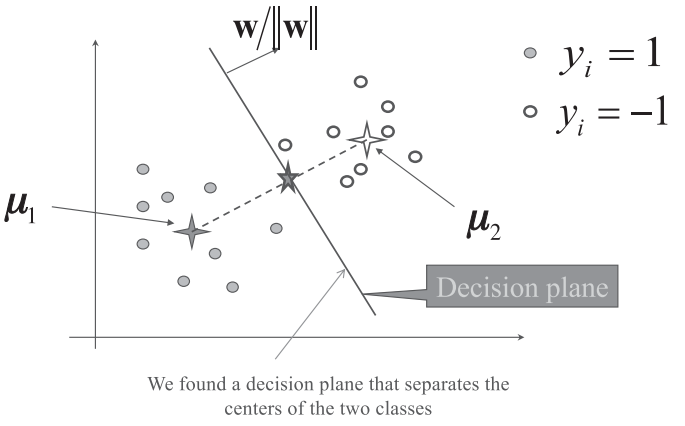


Figure 3.7 A naive way of finding a linear decision boundary that separates two classes. For two-dimensional problems, the decision plane is of equal distance to the two class-centers and is perpendicular to the line joining the centers. [Based on *Biometric Authentication: A Machine Learning Approach (Figure 4.1), by S.Y. Kung, M.W. Mak and S.H. Lin, 2005, Prentice Hall.*]

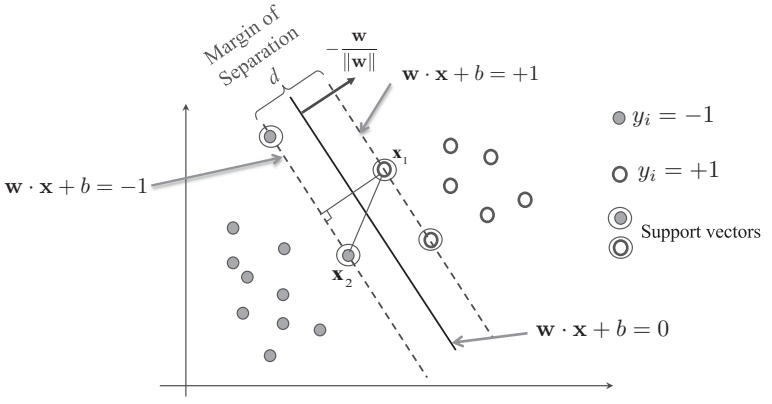


Figure 3.8 Finding a decision plane such that the margin between the two classes is the largest. [Based on *Biometric Authentication: A Machine Learning Approach (Figure 4.1), by S.Y. Kung, M.W. Mak and S.H. Lin, 2005, Prentice Hall.*]

The points $\mathbf{x}_1$ and $\mathbf{x}_2$ lie on the two lines parallel to the decision boundary, i.e.,

$$\begin{aligned} \mathbf{w}^{\mathsf{T}}\mathbf{x}_1 + b &= -1 \\ \mathbf{w}^{\mathsf{T}}\mathbf{x}_2 + b &= 1. \end{aligned} \tag{3.21}$$

The margin of separation d is the projection of $(\mathbf{x}_2 - \mathbf{x}_1)$ onto the direction perpendicular to the decision boundary, i.e.,

$$\begin{aligned} d &= \frac{\mathbf{w}^{\mathsf{T}}}{\|\mathbf{w}\|}(\mathbf{x}_2 - \mathbf{x}_1) \\ &= \frac{2}{\|\mathbf{w}\|} \qquad \text{(Using Eq. 3.21).} \end{aligned}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Therefore, maximizing d is equivalent to minimizing $\|\mathbf{w}\|^2$ subject to the constraint that no data points fall inside the margin of separation. This can be solved by using constrained optimization. Specifically, given a set of training data points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with class labels $\mathcal{Y} = \{y_1, \ldots, y_N\}$ where $y_i \in \{+1, -1\}$, the weight vector $\mathbf{w}$ can be obtained by solving the following *primal* problem:

$$\begin{aligned} &\text{Minimize } \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to } y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1, \quad \text{for } i = 1, \ldots, N. \end{aligned} \tag{3.22}$$

To solve this constrained optimization problem, we introduce Lagrange multipliers $\alpha_i \geq 0$ to form the Lagrangian function

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) - 1 \right), \tag{3.23}$$

where $\alpha = \{\alpha_1, \ldots, \alpha_N\}$. Setting the gradient of $L(\mathbf{w}, b, \alpha)$ with respect to $\mathbf{w}$ and b to zero, we have

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \text{ and } \sum_{i=1}^{N} \alpha_i y_i = 0. \tag{3.24}$$

Substituting Eq. 3.24 into Eq. 3.23, we have [32, Ch. 4]

$$L(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\mathsf{T} \mathbf{x}_j,$$

which is a function of α_i's instead of $\mathbf{w}$. As Eq. 3.24 suggests that if we know α_i's, we will know $\mathbf{w}$. Therefore, we may solve the primal problem by solving the following *dual* problem:[3]

$$\begin{aligned} &\text{maximize } L(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\mathsf{T} \mathbf{x}_j \\ &\text{subject to } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{N} \alpha_i y_i = 0. \end{aligned} \tag{3.25}$$

Note that because $L(\alpha)$ is quadratic in α_i, a global maximum of α_i can be found.

The solution of Eq. 3.25 comprises two kinds of Lagrange multipliers:

- $\alpha_i = 0$: The corresponding $\mathbf{x}_i$ are irrelevant to the classification task,
- $\alpha_i > 0$: The corresponding $\mathbf{x}_i$ are critical to the classification task,

where $\mathbf{x}_i$ for which $\alpha_i > 0$ are called *support vectors*. The parameter b can be computed by using the Karush-Kuhn-Tucker (KKT) condition [34], i.e., for any k such that $y_k = 1$ and $\alpha_k > 0$, we have

[3] Because Eq. 3.23 is convex in $\mathbf{w}$, the duality gap is zero. As a result, the primal and dual problems have the same solution.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$\alpha_k[y_k(\mathbf{w}^\mathsf{T}\mathbf{x}_k + b) - 1] = 0$$
$$\Longrightarrow b = 1 - \mathbf{w}^\mathsf{T}\mathbf{x}_k.$$

The SVM output is given by

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^\mathsf{T}\mathbf{x} + b \\ &= \sum_{i \in \mathcal{S}} \alpha_i y_i \mathbf{x}_i^\mathsf{T}\mathbf{x} + b, \end{aligned} \tag{3.26}$$

where $\mathcal{S}$ is the set of indexes for which $\alpha_k > 0$.

Linearly Non-Separable Problems

If the data patterns are not separable by a linear hyperplane (see Figure 3.9), a set of *slack* variables $\{\xi = \xi_1, \ldots, \xi_N\}$ is introduced with $\xi_i \geq 0$ such that the inequality constraints in Eq. 3.22 become

$$y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 - \xi_i \quad \forall i = 1, \ldots, N. \tag{3.27}$$

The slack variables $\{\xi_i\}_{i=1}^N$ allow some data to violate the constraints in Eq. 3.22. For example, in Figure 3.9, $\mathbf{x}_1$ and $\mathbf{x}_3$ violate the constraints in Eq. 3.22 and their slack variables ξ_1 and ξ_3 are nonzero. However, they satisfy the relaxed constraints in Eq. 3.27. The value of ξ_i indicates the degree of violation. For example, in Figure 3.9, $\mathbf{x}_3$ has a higher degree of violation than $\mathbf{x}_1$, and $\mathbf{x}_2$ does not violate the constraint in Eq. 3.22.

With the slack variables, the minimization problem becomes

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i, \quad \text{subject to} \quad y_i(\mathbf{w}^\mathsf{T}\mathbf{x}_i + b) \geq 1 - \xi_i, \tag{3.28}$$

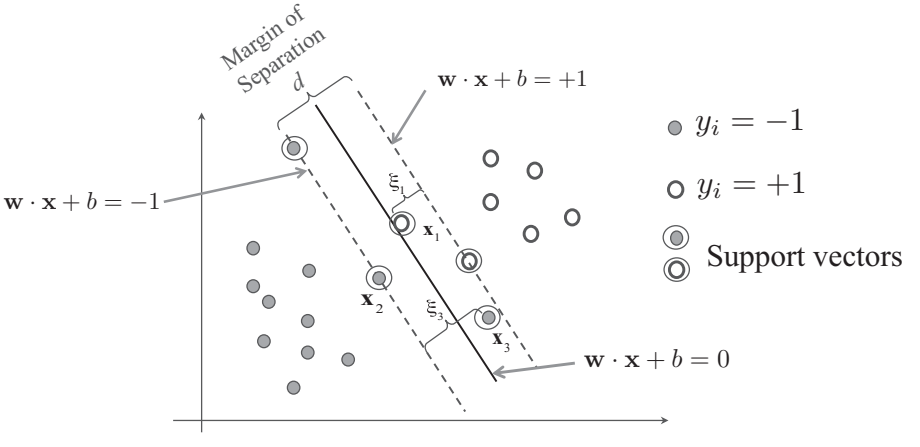


Figure 3.9 An example classification problem that cannot be perfectly separated by a linear hyperplane. The nonzero slack variables ξ_1 and ξ_3 allow $\mathbf{x}_1$ and $\mathbf{x}_3$ to violate the primal constraints in Eq. 3.22 so that the problem can still be solved by a linear SVM.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where C is a user-defined penalty parameter to penalize any violation of the safety margin for all training data. The new Lagrangian is

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i - \sum_{i=1}^{N} \alpha_i (y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i) - \sum_{i=1}^{N} \beta_i \xi_i, \tag{3.29}$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are, respectively, the Lagrange multipliers to ensure that $y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i$ and that $\xi_i \geq 0$. Differentiating $L(\mathbf{w}, b, \alpha)$ w.r.t. $\mathbf{w}$, b, and ξ_i and set the results to zero, we obtain Eq. 3.24 and $C = \alpha_i + \beta_i$. Substituting them into Eq. 3.29, we obtain the Wolfe dual:

$$\begin{aligned} &\text{maximize } L(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^{\mathsf{T}} \mathbf{x}_j \\ &\text{subject to } 0 \leq \alpha_i \leq C, i = 1, \ldots, N, \text{ and } \sum_{i=1}^{N} \alpha_i y_i = 0. \end{aligned} \tag{3.30}$$

The solution of Eq. 3.30 comprises three types of support vectors shown in Figure 3.10: (1) on the margin hyperplanes, (2) inside the margin of separation, and (3) outside the margin of separation but on the wrong side of the decision boundary. Note that for the nonsupport vectors and support vectors on the margin hyperplanes, their slack variables ξ_i are 0.

Nonlinear SVM

The earlier discussions cover large-margin classifiers with a linear decision boundary only. However, linear SVMs are not rich enough to solve complex problems. For exam-

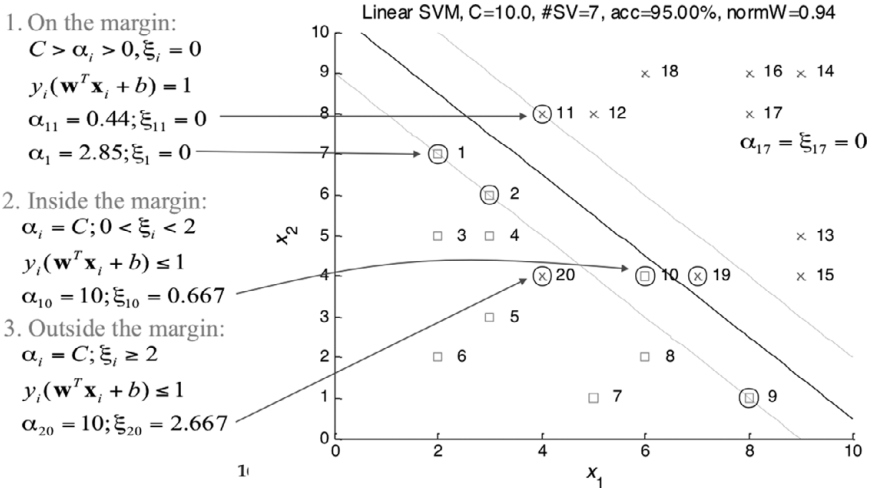


Figure 3.10 Three types of support vectors in a linear SVM with slack variables in its optimization constraints.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

ple, in Figure 3.10, no straight line can *perfectly* separate the two classes. Figure 3.11(a) shows another example in which two decision boundaries (thresholds) are required to separate the two classes. Again, a one-D linear SVM could not solve this problem because it can only provide one decision threshold.

In case the training data $\mathbf{x} \in \mathcal{X}$ are not linearly separable, we may use a nonlinear function $\boldsymbol{\phi}(\mathbf{x})$ to map the data from the input space to a new high-dimensional space (called feature space) where data become linearly separable. For example, in Figure 3.11, the nonlinear function is to perform the mapping:

$$\boldsymbol{\phi} : x \rightarrow [x \;\; x^2]^{\mathsf{T}}.$$

The decision boundary in Figure 3.11(b) is a straight line that can perfectly separate the two classes. Specifically, it can be written as

$$x^2 - c = [0 \;\; 1] \begin{bmatrix} x \\ x^2 \end{bmatrix} - c = 0$$

Or equivalently,

$$\mathbf{w}^{\mathsf{T}} \boldsymbol{\phi}(x) + b = 0, \tag{3.31}$$

where $\mathbf{w} = [0 \;\; 1]^{\mathsf{T}}$, $\boldsymbol{\phi}(x) = [x \;\; x^2]^{\mathsf{T}}$, and $b = -c$. Note that Eq. 3.31 is linear in $\boldsymbol{\phi}$, which means that by mapping x to $[x \;\; x^2]^{\mathsf{T}}$, a nonlinearly separable problem becomes linearly separable.

Figure 3.12(a) shows a two-dimensional example in which linear SVMs will not be able to perfectly separate the two classes. However, if we transform the input vectors $\mathbf{x} = [x_1 \;\; x_2]^{\mathsf{T}}$ by a nonlinear map:

$$\boldsymbol{\phi} : \mathbf{x} \rightarrow [x_1^2 \;\; \sqrt{2}x_1x_2 \;\; x_2^2]^{\mathsf{T}}, \tag{3.32}$$

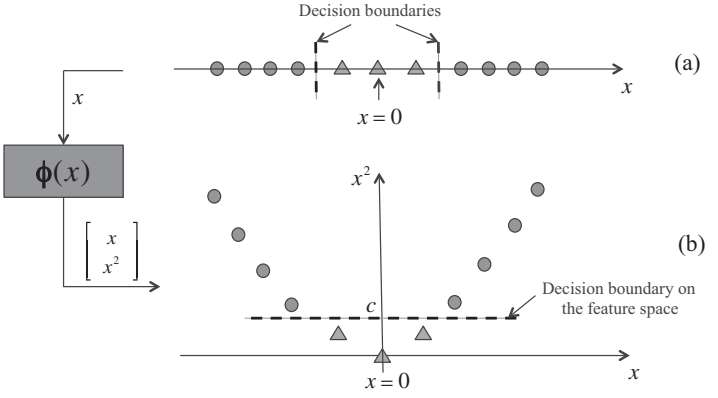


Figure 3.11 (a) One-dimensional example that cannot be solved by linear SVMs. (b) The same problem on a two-dimensional feature space in which linear SVMs can easily solve the problem.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

we will be able to use a linear SVM to separate the two classes in three-dimensional space, as shown in Figure 3.12(b). The linear SVM has the form

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}) + b \\ &= \mathbf{w}^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}) + b, \end{aligned}$$

where $\mathcal{S}$ is the set of support vector indexes and $\mathbf{w} = \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)$. Note that in this simple problem, the dot products $\boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}_j)$ for any $\mathbf{x}_i$ and $\mathbf{x}_j$ in the input space can be easily evaluated

$$\boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}_j) = x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{i2} x_{j1} x_{j2} + x_{i2}^2 x_{j2}^2 = (\mathbf{x}_i^\mathsf{T} \mathbf{x}_j)^2. \tag{3.33}$$

While the mapping in Eq. 3.32 can transform the nonlinearly separable problem in Figure 3.12 into a linearly separable one, its capability is rather limited, as demonstrated in the two-spiral problem in Figure 3.13(a) and 3.13(b). The failure of this mapping in the two-spiral problem suggests that it is necessary to increase the dimension of the feature space. To achieve this without increasing the degree of the polynomial, we may change Eq. 3.33 to

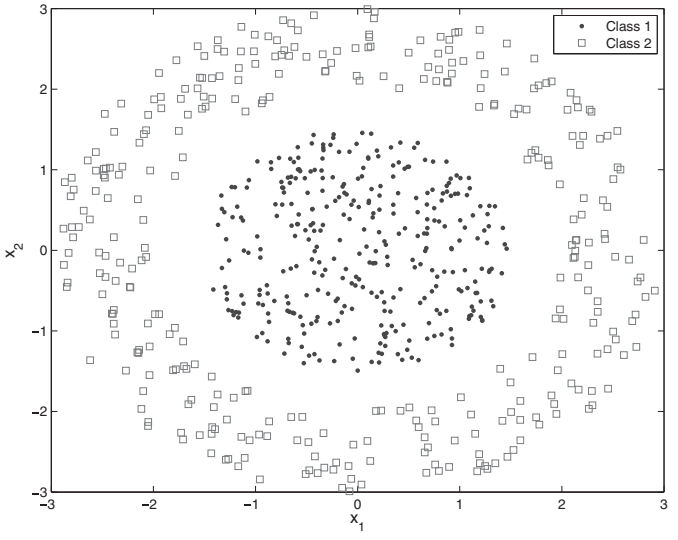
$$\boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}_j) = (1 + \mathbf{x}_i^\mathsf{T} \mathbf{x}_j)^2. \tag{3.34}$$

With the addition of a constant term, the dimension of $\boldsymbol{\phi}(\mathbf{x})$ in Eq. 3.34 increases to 6. This can be observed by writing $\boldsymbol{\phi}(\mathbf{u})^\mathsf{T} \boldsymbol{\phi}(\mathbf{v})$ for any vectors $\mathbf{u}$ and $\mathbf{v}$ in the input space as

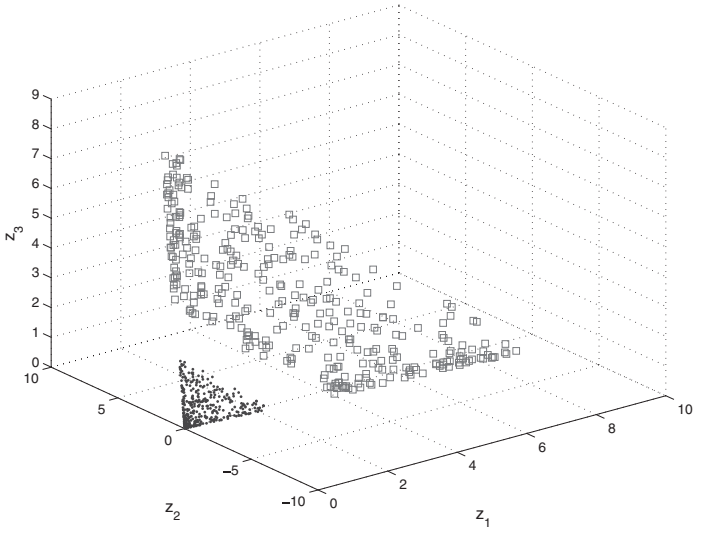
$$\begin{aligned} \boldsymbol{\phi}(\mathbf{u})^\mathsf{T} \boldsymbol{\phi}(\mathbf{v}) &= (1 + \mathbf{u}^\mathsf{T} \mathbf{v})^2 \\ &= \left(1 + [u_1 \;\; u_2] \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}\right)^2 \\ &= (1 + u_1 v_1 + u_2 v_2)(1 + u_1 v_1 + u_2 v_2) \\ &= 1 + 2u_1 v_1 + 2u_2 v_2 + 2u_1 v_1 u_2 v_2 + u_1^2 v_1^2 + u_2^2 v_2^2 \\ &= \begin{bmatrix} 1 & \sqrt{2}u_1 & \sqrt{2}u_2 & \sqrt{2}u_1 u_2 & u_1^2 & u_2^2 \end{bmatrix} \begin{bmatrix} 1 \\ \sqrt{2}v_1 \\ \sqrt{2}v_2 \\ \sqrt{2}v_1 v_2 \\ v_1^2 \\ v_2^2 \end{bmatrix}. \end{aligned}$$

Therefore, vector $\mathbf{x}$ is mapped to $\boldsymbol{\phi}(\boldsymbol{x}) = \begin{bmatrix} 1 & \sqrt{2}x_1 & \sqrt{2}x_2 & \sqrt{2}x_1 x_2 & x_1^2 & x_2^2 \end{bmatrix}^\mathsf{T}$, which is a six-dimensional vector. The decision boundary in the $\boldsymbol{\phi}$-space is linear because the output of the SVM can now be written as:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press



(a)



(b)

Figure 3.12 (a) A two-dimensional example that cannot be solved by linear SVMs. (b) By applying the mapping in Eq. 3.32, the same problem can be easily solved by a linear SVM on the three-dimensional space.

$$f(\mathbf{x}) = \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x})^{\mathsf{T}} \boldsymbol{\phi}(\mathbf{x}_i) + b.$$

Increasing the dimension to 6, however, is still not sufficient to solve the two-D spiral problem, as shown in Figure 3.13. This is because the decision boundary produced by a second-order polynomial is too smooth. The problem can be alleviated by increasing the order to three. For example, if we define a mapping $\boldsymbol{\phi}(\mathbf{x})$ such that

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

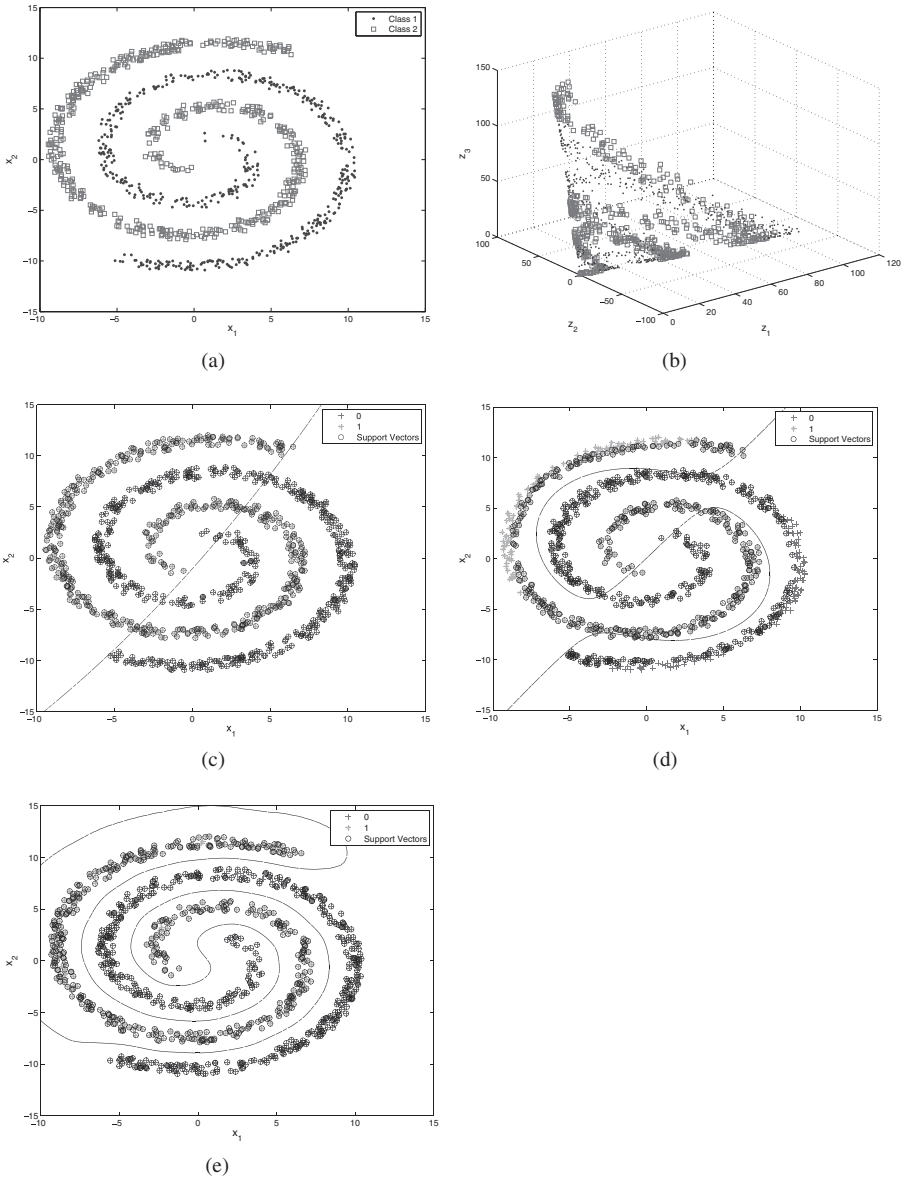


Figure 3.13 (a) A two-dimensional spiral example. (b) The mapping in Eq. 3.32 is not sufficient to convert the nonlinearly separable problem into a linearly separable one. (c) A decision boundary produced by a second-order polynomial SVM. (d) A decision boundary produced by a third-order polynomial SVM. (e) A decision boundary produced by an RBF-SVM with $\sigma = 1$. In (c)–(e), the penalty factor C was set to 1.

$$\boldsymbol{\phi}(\mathbf{x}_i)^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_j) = (1 + \mathbf{x}_i^{\mathsf{T}}\mathbf{x}_j)^3 \tag{3.35}$$

for any $\mathbf{x}_i$ and $\mathbf{x}_j$ in the input space, we have a better chance of solving (not perfectly) the two-spiral problem. This is because the polynomial in Eq. 3.35 has 10 terms,[4] meaning

[4] The number of terms in the polynomial expansion $\left(\sum_{i=1}^{k} x_i\right)^n$ is $\binom{n+k-1}{n}$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

that the transformed space has 10 dimensions. In such high dimensional space, the highly nonlinear two-spiral problem will become more linear. Figure 3.13(d) shows the decision boundary found by an SVM of the form

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}) + b \\ &= \sum_{i \in \mathcal{S}} \alpha_i y_i (1 + \mathbf{x}_i^\mathsf{T} \mathbf{x})^3 + b. \end{aligned}$$

Evidently, the SVM can classify most of the data points correctly.

When the polynomial degree increases, the dimension of $\boldsymbol{\phi}(\mathbf{x})$ also increases. For example, for $\mathbf{x} \in \mathbb{R}^2$ and third-order polynomial (Eq. 3.35), the dimension is 10. However, increasing the degree of polynomial does not necessarily increase the complexity of the decision boundary. This may only increase the curvature of the boundary at some locations in the input space, as demonstrated in Figure 3.13(d).

The dimension of $\boldsymbol{\phi}$-space increases rapidly with the input dimension. For example, if the input dimension is 100, the dimension of $\boldsymbol{\phi}(\mathbf{x})$ becomes 176,851 for third-order polynomial. This will be too expensive to evaluate the dot products in such high-dimensional space. Fortunately, we may evaluate the right-hand side of Eq. 3.35 instead of the dot product on the left-hand side. The former is much cheaper and can be easily generalized to polynomials of any degree d:

$$\boldsymbol{\phi}(\mathbf{x}_i)^\mathsf{T} \boldsymbol{\phi}(\mathbf{x}_j) = (1 + \mathbf{x}_i^\mathsf{T} \mathbf{x}_j)^d. \tag{3.36}$$

To emphasize the fact that it is not necessary to evaluate the dot products, it is common to write it as a kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (1 + \mathbf{x}_i^\mathsf{T} \mathbf{x}_j)^d.$$

In additional to the polynomial kernel, the radial basis function (RBF) kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left\{ -\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2} \right\}$$

is also commonly used. In this kernel, the parameter σ controls the kernel width, which in turn controls the curvature of the decision boundary. The smaller the value of σ, the sharper the curvature.

With this kernel notation, the decision function of an SVM can be written as

$$f(\mathbf{x}) = \sum_{i \in \mathcal{S}} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b.$$

where $K(\cdot, \cdot)$ can be either a linear kernel, or a polynomial kernel, or an RBF kernel. Note that the linear SVM in Eq. 3.26 uses the linear kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\mathsf{T} \mathbf{x}_j.$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

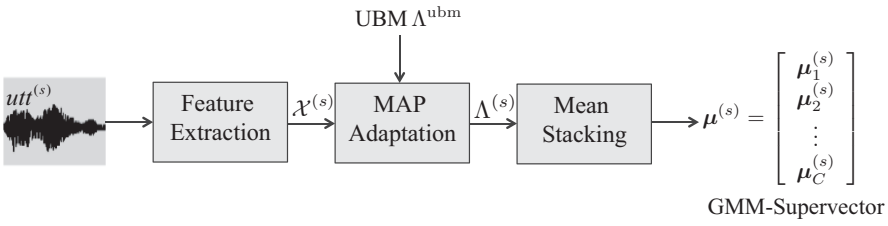


Figure 3.14 Extraction of a GMM-supervector from an utteracne.

3.2.2 GMM Supervectors

To apply SVMs for speaker verification, it is necessary to convert a variable-length utterance into a fixed-length vector, the so-called vectorization process. Campbell et al. [14] proposed to achieve such task by stacking the mean vectors of a MAP-adapted GMM as shown in Figure 3.14. Given the speech of a client speaker, MAP adaptation (Eq. 3.16) is applied to create his/her GMM model. Then, the mean vectors of the speaker model are stacked to form a supervector with dimension CF, where C is the number of Gaussians in the UBM and F is the dimension of the acoustic vectors. Typically, $C = 1024$ and $F = 60$, which result in supervectors with 61,440 dimensions.

Because of this high dimensionality, it is sensible to use linear SVMs to classify the supervectors. To incorporate the covariance matrices and the mixture coefficients of the UBM into the SVM, Campbell et al. suggest using a linear kernel of the form:[5]

$$K\left(\text{utt}^{(i)}, \text{utt}^{(j)}\right) = \sum_{c=1}^{C} \left(\sqrt{\pi_c}\boldsymbol{\Sigma}_c^{-\frac{1}{2}}\boldsymbol{\mu}_c^{(i)}\right)^{\mathsf{T}} \left(\sqrt{\pi_c}\boldsymbol{\Sigma}_c^{-\frac{1}{2}}\boldsymbol{\mu}_c^{(j)}\right), \tag{3.37}$$

where i and j index to the ith and jth utterances, respectively, and π_c, $\boldsymbol{\Sigma}_c$ and $\boldsymbol{\mu}_c$ are the weight, covariance matrix, and mean vector of the cth mixture, respectively. Eq. 3.37 can be written in a more compact form:

$$\begin{aligned} K\left(\text{utt}^{(i)}, \text{utt}^{(j)}\right) &= \left(\boldsymbol{\Omega}^{-\frac{1}{2}}\overrightarrow{\boldsymbol{\mu}}^{(i)}\right)^{\mathsf{T}} \left(\boldsymbol{\Omega}^{-\frac{1}{2}}\overrightarrow{\boldsymbol{\mu}}^{(j)}\right) \\ &\equiv K\left(\overrightarrow{\boldsymbol{\mu}}^{(i)}, \overrightarrow{\boldsymbol{\mu}}^{(j)}\right) \end{aligned} \tag{3.38}$$

where

$$\boldsymbol{\Omega} = \text{diag}\left\{\pi_1^{-1}\boldsymbol{\Sigma}_1, \ldots, \pi_C^{-1}\boldsymbol{\Sigma}_C\right\} \quad \text{and} \quad \overrightarrow{\boldsymbol{\mu}} = \left[\boldsymbol{\mu}_1^{\mathsf{T}}, \ldots, \boldsymbol{\mu}_C^{\mathsf{T}}\right]^{\mathsf{T}}. \tag{3.39}$$

In practice, $\boldsymbol{\Sigma}_c$'s are assumed to be diagonal.

For each target speaker, a set of target-speaker's supervectors are obtained from his/her enrollment utterances, one for each utterance. Then, a linear SVM is trained to discriminate his/her supervector(s) from a set of supervectors derived from a number of background speakers. After training, for target-speaker s, we have

[5] To avoid cluttering with symbols, the superscript "ubm" in $\boldsymbol{\Sigma}_c$ and π_c is omitted.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

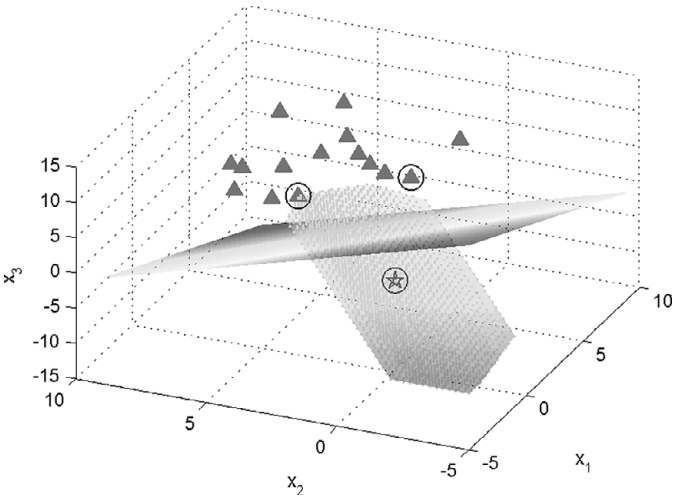


Figure 3.15 Illustration of a linear SVM in three-dimensional space when solving a data-imbalance problem in which the minority-class samples are represented by pentagrams and the majority-class samples are represented by filled triangles. The shaded area under the decision plane represents the possible locations of the minority-class samples. [Reprinted from *Acoustic vector resampling for GMMSVM-based speaker verification (Figure 1), M.W. Mak and W. Rao, Proceedings of Annual Conference of International Speech Communication Association, 2010, pp. 1449–1452, with permission of ISCA.*]

$$f^{(s)}(\vec{\mu}) = \sum_{i \in \mathcal{S}_s} \alpha_i^{(s)} K\left(\vec{\mu}^{(i)}, \vec{\mu}\right) - \sum_{i \in \mathcal{S}_{\text{bkg}}} \alpha_i^{(s)} K\left(\vec{\mu}^{(i)}, \vec{\mu}\right) + b^{(s)}, \tag{3.40}$$

where $\mathcal{S}_s$ and $\mathcal{S}_{\text{bkg}}$ are the support vector indexes corresponding to the target speaker and background speakers, respectively, and $\alpha_i^{(s)}$'s and $b^{(s)}$ are the Lagrange multipliers and the bias term of the target-speaker's SVM.

In practical situations, the number of target-speaker utterances is very small. This create a severe data-imbalance problem [35] in which the decision boundary is largely defined by the supervectors of the nontarget speakers. Another issue caused by data imbalance is that the SVM's decision boundary tends to skew toward the minority class [36, 37]. This will lead to a large number of false rejections unless the decision threshold has been adjusted to compensate for the bias. Figure 3.15 illustrates such situation. In the figure, there is a large region in the supervector space in which the target-speaker's supervector will not affect the orientation of the decision boundary.

The data-imbalance problem in GMM–SVM systems can be overcome by creating more supervectors from the utterance(s) of the target speakers [35, 38, 39]. Figure 3.16 illustrates the procedure, which is called utterance-partitioning with acoustic vector resampling (UP-VAR). The goal is to increase the number of sub-utterances without compromising their representation power. This is achieved by the following steps:

1. Randomly rearrange the sequence of acoustic vectors in an utterance;
2. Partition the acoustic vectors of an utterance into N segments;
3. Repeated Step 1 and Step 2 R times.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

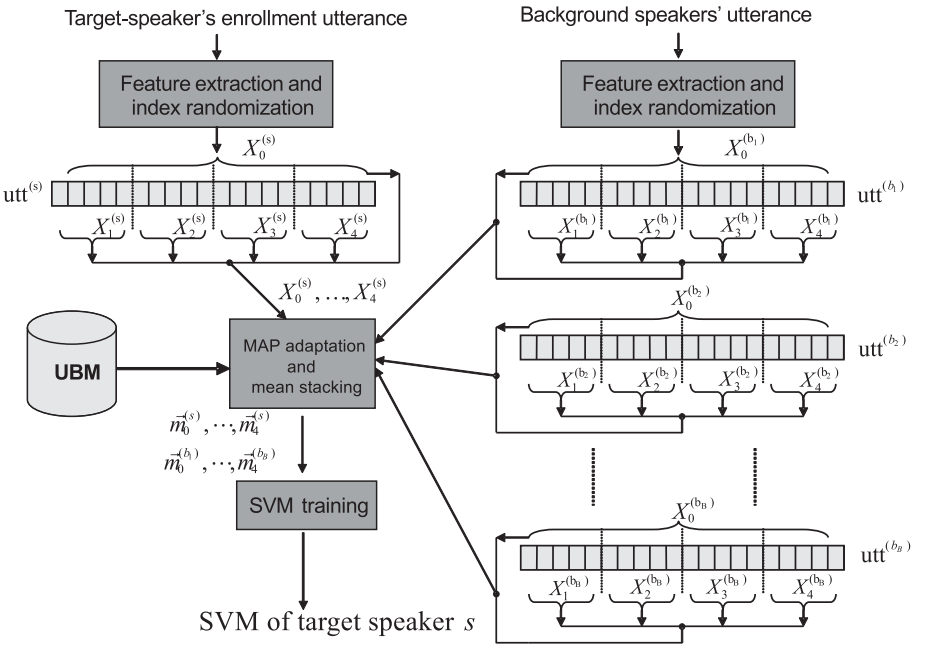


Figure 3.16 Illustration of the utterance partitioning with acoustic vector resampling (UP-AVR) process that increases the number of supervectors from a target speaker for training a speaker-dependent SVM. [Reprinted from *Utterance Partitioning with Acoustic Vector Resampling for GMM-SVM Speaker Verification (Figure 2), M.W. Mak and W. Rao, Speech Communication, vol. 53, no. 1, Jan. 2011, pp. 119–130, with permission of Elsevier.*]

By repeating Step 1 and Step 2 R times, we obtain $RN+1$ target-speaker's supervectors for training the speaker-dependent SVM.

3.2.3 GMM–SVM Scoring

During scoring, given a test utterance $\text{utt}^{(t)}$ and the SVM of a target-speaker s, we first apply MAP adaptation to the UBM to obtain the corresponding supervector $\boldsymbol{\mu}^{(t)}$ (see Figure 3.14). Then, we compute the GMM–SVM score as follows:

$$S_{\text{GMM–SVM}}(s,t) = \sum_{i\in\mathcal{S}_s} \alpha_i^{(s)} K\left(\overrightarrow{\boldsymbol{\mu}}^{(i)}, \overrightarrow{\boldsymbol{\mu}}^{(t)}\right) - \sum_{i\in\mathcal{S}_{\text{bkg}}} \alpha_i^{(s)} K\left(\overrightarrow{\boldsymbol{\mu}}^{(i)}, \overrightarrow{\boldsymbol{\mu}}^{(t)}\right) + b^{(s)}. \tag{3.41}$$

Therefore, GMM–SVM scoring amounts to finding the distance of the test supervector from the speaker-dependent hyperplane defined by the SVM of the target speaker.

Comparing the first line of Eq. 3.20 and Eq. 3.41, we can see the similarity between the GMM–UBM scoring and GMM–SVM scoring. In particular, both have two terms, one from the target speaker and one from the background speakers. However, there are also important differences. First, the two terms in GMM–UBM scoring are unweighted, meaning that they have equal contribution to the score. On the other hand, the two terms

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

in GMM–SVM are weighted by Lagrange multipliers, which are optimized to produce the best discrimination between the target speaker and the background speakers. Second, the speaker model $\Lambda^{(s)}$ and Λ^{ubm} in GMM–UBM are separately trained, whereas the Lagrange multipliers in GMM–SVM are jointly trained by the SVM optimizer. This means that for each speaker, the optimizer will automatically find a set of *important* utterances from his/her enrollment sessions and the background speakers. This speaker-dependent selection of utterances make the GMM–SVM systems more flexible and perform much better than GMM–UBM systems.

3.2.4 Nuisance Attribute Projection

Because the vectorization process in Figure 3.14 will collect both the speaker and nonspeaker (e.g., channel) statistics from the input utterance through Eq. 3.3 and Eq. 3.16, the GMM-supervector $\overrightarrow{\mu}$ will contain not only speaker but also other nonspeaker information. Therefore, it is important to remove such unwanted information before performing GMM–SVM scoring. One of the most promising approaches is the nuisance attribute projection (NAP) [40–42].

Basic Idea of NAP

The idea of NAP is to find a subspace within the GMM-supervector space in which all nonspeaker variabilities occur. The method requires a training set comprising multiple speakers and multiple recording sessions per speaker. Assume that we are given a training set comprising N GMM-supervectors $\mathcal{X} = \{\overrightarrow{\mu}^{(1)}, \ldots, \overrightarrow{\mu}^{(N)}\}$. Our goal is to find a subspace defined by the column vectors in $\boldsymbol{U}$ and define a projection matrix

$$\mathbf{P} \equiv \mathbf{I} - \mathbf{U}\mathbf{U}^{\mathsf{T}}, \tag{3.42}$$

where $\mathbf{I}$ is an identity matrix, such that the projected vectors

$$\tilde{\mu}^{(i)} \equiv \mathbf{P}\overrightarrow{\mu}^{(i)} = (\mathbf{I} - \mathbf{U}\mathbf{U}^{\mathsf{T}})\overrightarrow{\mu}^{(i)}, \quad i = 1, \ldots, N \tag{3.43}$$

retain most of the speaker information but with nonspeaker information suppressed. For $\tilde{\mu}^{(i)}$'s to retain most of the speaker information, the rank of $\mathbf{U}$ should be much lower than the dimension of $\overrightarrow{\mu}^{(i)}$'s.

Figure 3.17 illustrates the concept of suppressing session variability in NAP. In the figure, the superscript h is the session index, and all sessions (supervectors) from the same speaker s – indexed by (s, h) – lie on the line defined by the single column in $\mathbf{U}$. By applying Eq. 3.43, we obtain a projected vector $\tilde{\mu}^{(s)}$ independent of the session index h.

Figure 3.18 demonstrates the capability of NAP in a three-dimensional toy problem in which nonspeaker variability occurs along the x_1-axis, i.e., $\boldsymbol{U} = [1\ 0\ 0]^{\mathsf{T}}$. In the figure, the feature vectors $\mathbf{x}$'s come from two different speakers, one for each cluster. Because $\boldsymbol{U} = [1\ \ 0\ \ 0]^{\mathsf{T}}$ is of rank 1, $\mathbf{U}\mathbf{U}^{\mathsf{T}}\mathbf{x}$'s vary along a straight line and $(\mathbf{I} - \mathbf{U}\mathbf{U}^{\mathsf{T}})\mathbf{x}$'s lie on a plane perpendicular to the line. As shown in the figure, after NAP projection, there is no direction in which the two clusters (black ○) are indistinguishable.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

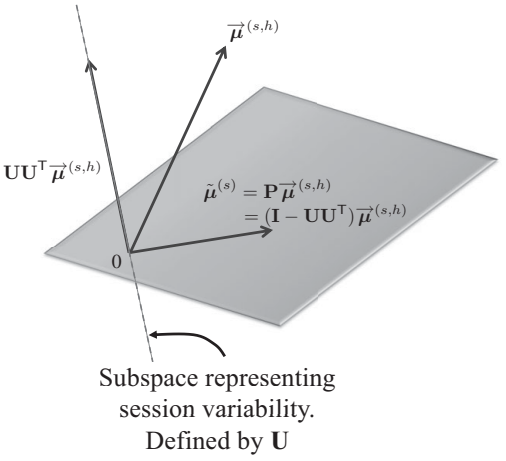


Figure 3.17 Suppressing session variability by nuisance attribute projection (NAP) when $\mathbf{U}$ is of rank 1. The superscripts s and h stand for speaker and session, respectively. The dashed line represents the nuisance direction and the shaped plane is orthogonal to it. All of the NAP-projected vectors lie on the shaped plane.

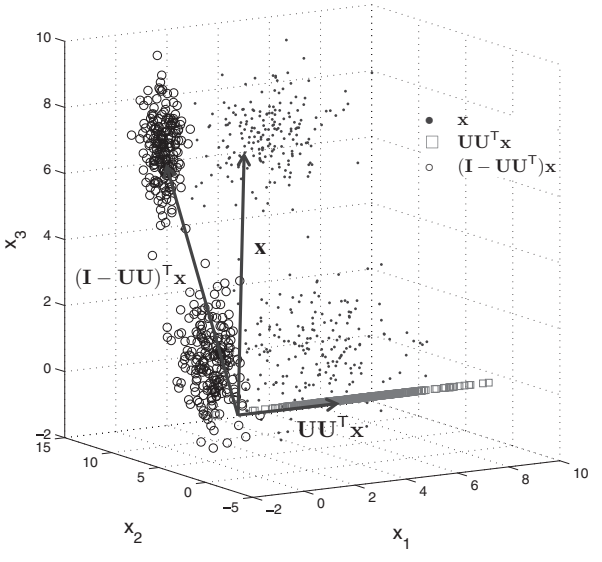


Figure 3.18 A three-dimensional example illustrating how NAP suppresses nonspeaker variability when $\mathbf{U} = [1\ 0\ 0]^{\mathsf{T}}$. Before the projection, the dots from two different speakers are indistinguishable along the x_1-axis, as indicated by the "□." After the projection, the black "○" of the two speakers lie on the plane perpendicular to the x_1-axis. As there is no variation along the x_1-axis after the projection, nonspeaker variability has been suppressed.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Objective Function of NAP

The subspace $\mathbf{U}$ can be found by minimizing the objective function:

$$\mathbf{U}^* = \underset{\mathbf{U}:\mathbf{U}^\mathsf{T}\mathbf{U}=\mathbf{I}}{\operatorname{argmin}} \frac{1}{2}\sum_{ij} w_{ij} \left\| \mathbf{P}\left(\overrightarrow{\boldsymbol{\mu}}^{(i)} - \overrightarrow{\boldsymbol{\mu}}^{(j)}\right)\right\|^2, \tag{3.44}$$

where $\mathbf{P} = \mathbf{I} - \mathbf{U}\mathbf{U}^\mathsf{T}$ and

$$w_{ij} = \begin{cases} 1 & \overrightarrow{\boldsymbol{\mu}}^{(i)} \text{ and } \overrightarrow{\boldsymbol{\mu}}^{(j)} \text{ belong to the same speaker} \\ 0 & \text{otherwise.} \end{cases} \tag{3.45}$$

Eq. 3.44 and Eq. 3.45 suggest that we pull the projected supervectors belonging to the same speaker together and do not care about the vector pairs coming from different speakers.

To find $\mathbf{U}^*$ in Eq. 3.44, we define the data matrix $\mathbf{X} \equiv [\overrightarrow{\boldsymbol{\mu}}^{(1)} \ \ldots \ \overrightarrow{\boldsymbol{\mu}}^{(N)}]$ and the vector difference

$$\overrightarrow{\mathbf{d}}_{ij} \equiv \overrightarrow{\boldsymbol{\mu}}^{(i)} - \overrightarrow{\boldsymbol{\mu}}^{(j)}.$$

Then, the weighted projected distance in Eq. 3.44 can be expressed as

$$\begin{aligned} d_{\text{NAP}} &= \frac{1}{2}\sum_{ij} w_{ij} \left(\mathbf{P}\overrightarrow{\mathbf{d}}_{ij}\right)^\mathsf{T} \left(\mathbf{P}\overrightarrow{\mathbf{d}}_{ij}\right) \\ &= \frac{1}{2}\sum_{ij} w_{ij}\, \overrightarrow{\mathbf{d}}_{ij}^\mathsf{T}\mathbf{P}^\mathsf{T}\mathbf{P}\overrightarrow{\mathbf{d}}_{ij} \\ &= \frac{1}{2}\sum_{ij} w_{ij}\, \overrightarrow{\mathbf{d}}_{ij}^\mathsf{T}(\mathbf{I} - \mathbf{U}\mathbf{U}^\mathsf{T})^\mathsf{T}(\mathbf{I} - \mathbf{U}\mathbf{U}^\mathsf{T})\overrightarrow{\mathbf{d}}_{ij} \\ &= \frac{1}{2}\sum_{ij} w_{ij}\, \overrightarrow{\mathbf{d}}_{ij}^\mathsf{T}\overrightarrow{\mathbf{d}}_{ij} - \frac{1}{2}\sum_{ij} w_{ij}\, \overrightarrow{\mathbf{d}}_{ij}^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\overrightarrow{\mathbf{d}}_{ij}, \end{aligned} \tag{3.46}$$

where we have used the constraint $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}$. Dropping terms independent of $\mathbf{U}$, we have

$$\begin{aligned} d'_{\text{NAP}} &= -\frac{1}{2}\sum_{ij} w_{ij} \left(\overrightarrow{\boldsymbol{\mu}}^{(i)} - \overrightarrow{\boldsymbol{\mu}}^{(j)}\right)^\mathsf{T} \mathbf{U}\mathbf{U}^\mathsf{T} \left(\overrightarrow{\boldsymbol{\mu}}^{(i)} - \overrightarrow{\boldsymbol{\mu}}^{(j)}\right) \\ &= -\frac{1}{2}\sum_{ij} w_{ij} (\overrightarrow{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(i)} - \frac{1}{2}\sum_{ij} w_{ij} (\overrightarrow{\boldsymbol{\mu}}^{(j)})^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\boldsymbol{\mu}^{(j)} \\ &\quad + \sum_{ij} w_{ij} (\overrightarrow{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(j)} \\ &= -\sum_{ij} w_{ij} (\overrightarrow{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(i)} + \sum_{ij} w_{ij} (\overrightarrow{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\mathbf{U}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(j)}, \end{aligned} \tag{3.47}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where we have used the property $w_{ij} = w_{ji}$. Using the identity $\mathbf{a}^\mathsf{T}\mathbf{B}\mathbf{B}^\mathsf{T}\mathbf{a} = \mathrm{Tr}\{\mathbf{B}^\mathsf{T}\mathbf{a}\mathbf{a}^\mathsf{T}\mathbf{B}\}$, where Tr stands for matrix trace, Eq. 3.47 can be written as

$$\begin{aligned} d'_{\mathrm{NAP}} &= -\mathrm{Tr}\left\{\sum_{ij} w_{ij}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(i)}(\overrightarrow{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\mathbf{U}\right\} + \mathrm{Tr}\left\{\sum_{ij} w_{ij}\mathbf{U}^\mathsf{T}\overrightarrow{\boldsymbol{\mu}}^{(i)}(\overrightarrow{\boldsymbol{\mu}}^{(j)})^\mathsf{T}\mathbf{U}\right\} \\ &= -\mathrm{Tr}\left\{\mathbf{U}^\mathsf{T}\mathbf{X}\mathrm{diag}(\mathbf{W}\mathbf{1})\mathbf{X}^\mathsf{T}\mathbf{U}\right\} + \mathrm{Tr}\left\{\mathbf{U}^\mathsf{T}\mathbf{X}\mathbf{W}\mathbf{X}^\mathsf{T}\mathbf{U}\right\} \\ &= \mathrm{Tr}\left\{\mathbf{U}^\mathsf{T}\mathbf{X}\left[\mathbf{W} - \mathrm{diag}(\mathbf{W}\mathbf{1})\right]\mathbf{X}^\mathsf{T}\mathbf{U}\right\}, \end{aligned} \tag{3.48}$$

where $\mathbf{W}$ is the weight matrix in Eq. 3.45, diag($\mathbf{a}$) means converting $\mathbf{a}$ into a diagonal matrix, and $\mathbf{1}$ is a vector of all 1's. It can be shown [43] that minimizing d'_{NAP} in Eq. 3.48 with the constraint $\mathbf{U}^\mathsf{T}\mathbf{U} = \mathbf{I}$ is equivalent to finding the first K eigenvectors with the smallest eigenvalues of

$$\mathbf{X}\left[\mathbf{W} - \mathrm{diag}(\mathbf{W}\mathbf{1})\right]\mathbf{X}^\mathsf{T}\mathbf{U} = \boldsymbol{\Lambda}\mathbf{U},$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\mathbf{X}\left[\mathbf{W} - \mathrm{diag}(\mathbf{W}\mathbf{1})\right]\mathbf{X}^\mathsf{T}$ in its diagonal. Once $\mathbf{U}$ has been found, we may apply Eq. 3.43 to all GMM-supervectors for both SVM training and GMM–SVM scoring.

Note that speakers with lots of sessions will dominate the objective function in Eq. 3.44. To balance the influence of individual speakers in the training set, we may change the weights in Eq. 3.45 to

$$w_{ij} = \begin{cases} \frac{1}{N_k^2} & \text{both } \overrightarrow{\boldsymbol{\mu}}^{(i)} \text{ and } \overrightarrow{\boldsymbol{\mu}}^{(j)} \text{ belong to Speaker } k \\ 0 & \text{otherwise,} \end{cases} \tag{3.49}$$

where N_k is the number of utterances from Speaker k.

Relationship with Within-Class Covariance Matrix

With the weights in Eq. 3.49, the distance in Eq. 3.46 can be obtained from the within-class covariance matrix. To see this, let's assume that we have N training utterances spoken by K speakers, and for the kth speaker, we have N_k training utterances whose indexes are stored in the set $\mathcal{C}_k$. Then, we can write Eq. 3.46 as follows:

$$\begin{aligned} d_{\mathrm{NAP}} &= \frac{1}{2}\sum_{k=1}^{K}\frac{1}{N_k^2}\sum_{i\in\mathcal{C}_k}\sum_{j\in\mathcal{C}_k}(\tilde{\boldsymbol{\mu}}^{(i)} - \tilde{\boldsymbol{\mu}}^{(j)})^\mathsf{T}(\tilde{\boldsymbol{\mu}}^{(i)} - \tilde{\boldsymbol{\mu}}^{(j)}) \\ &= \sum_{k=1}^{K}\frac{1}{N_k}\sum_{i\in\mathcal{C}_k}(\tilde{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\tilde{\boldsymbol{\mu}}^{(i)} - \sum_{k=1}^{K}\frac{1}{N_k^2}\sum_{i\in\mathcal{C}_k}\sum_{j\in\mathcal{C}_k}(\tilde{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\tilde{\boldsymbol{\mu}}^{(j)} \\ &= \sum_{k=1}^{K}\left[\frac{1}{N_k}\sum_{i\in\mathcal{C}_k}(\tilde{\boldsymbol{\mu}}^{(i)})^\mathsf{T}\tilde{\boldsymbol{\mu}}^{(i)} - \tilde{\mathbf{m}}_k^\mathsf{T}\tilde{\mathbf{m}}_k\right], \end{aligned} \tag{3.50}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\tilde{\mathbf{m}}_k = \frac{1}{N_k}\sum_{i\in\mathcal{C}_k}\tilde{\boldsymbol{\mu}}^{(i)}$. Note that the within-class covariance (WCCN) matrix is given by

$$\begin{aligned}\boldsymbol{\Sigma}_{\text{wccn}} &= \sum_{k=1}^{K}\frac{1}{N_k}\sum_{i\in\mathcal{C}_k}(\tilde{\boldsymbol{\mu}}^{(i)}-\tilde{\mathbf{m}}_k)(\tilde{\boldsymbol{\mu}}^{(i)}-\tilde{\mathbf{m}}_k)^{\mathsf{T}}\\ &= \sum_{k=1}^{K}\frac{1}{N_k}\left[\sum_{i\in\mathcal{C}_k}\tilde{\boldsymbol{\mu}}^{(i)}(\tilde{\boldsymbol{\mu}}^{(i)})^{\mathsf{T}}-\sum_{i\in\mathcal{C}_k}\tilde{\boldsymbol{\mu}}^{(i)}\tilde{\mathbf{m}}_k^{\mathsf{T}}-\sum_{i\in\mathcal{C}_k}\tilde{\mathbf{m}}_k(\tilde{\boldsymbol{\mu}}^{(i)})^{\mathsf{T}}+\sum_{i\in\mathcal{C}_k}\tilde{\mathbf{m}}_k\tilde{\mathbf{m}}_k^{\mathsf{T}}\right]\\ &= \sum_{k=1}^{K}\frac{1}{N_k}\left[\sum_{i\in\mathcal{C}_k}\tilde{\boldsymbol{\mu}}^{(i)}(\tilde{\boldsymbol{\mu}}^{(i)})^{\mathsf{T}}-2N_k\tilde{\mathbf{m}}_k\tilde{\mathbf{m}}_k^{\mathsf{T}}+N_k\tilde{\mathbf{m}}_k\tilde{\mathbf{m}}_k^{\mathsf{T}}\right]\\ &= \sum_{k=1}^{K}\frac{1}{N_k}\left[\sum_{i\in\mathcal{C}_k}\tilde{\boldsymbol{\mu}}^{(i)}(\tilde{\boldsymbol{\mu}}^{(i)})^{\mathsf{T}}-\tilde{\mathbf{m}}_k\tilde{\mathbf{m}}_k^{\mathsf{T}}\right].\end{aligned} \tag{3.51}$$

Comparing Eq. 3.51 with Eq. 3.50 and using the property of matrix trace, $\text{Tr}\{\mathbf{a}\mathbf{a}^{\mathsf{T}}\} = \mathbf{a}^{\mathsf{T}}\mathbf{a}$, we obtain

$$\text{Tr}\{\boldsymbol{\Sigma}_{\text{wccn}}\} = d_{\text{NAP}}. \tag{3.52}$$

This means that the objective function in Eq. 3.44 using the weights in Eq. 3.49 is equivalent to minimizing the trace of the within-class covariance matrix.

3.3 Factor Analysis

Although NAP can reduce the session variability, it does not define a speaker subspace to model inter-speaker variability. Moreover, it does not take the prior distribution of the supervectors into account when projecting out the nuisance information. These two limitations, however, can be readily addressed by factor analysis models.

Factor analysis (FA) is a statistical method for modeling the variability and covariance structure of observed variables through a smaller number of unobserved variables called *factors*. It was originally introduced by psychologists to find the underlying latent factors that account for the correlations among a set of observations or measures.

For instance, the exam scores of 10 different subjects of 1000 students may be explained by two latent factors (also called common factors): language ability and math ability. Each student has their own values for these two factors across all of the 10 subjects and his/her exam score for each subject is a linear weighted sum of these two factors plus the score mean and an error. For each subject, all students share the same weights, which are referred to as the factor loadings, for this subject. Specifically, denote $\mathbf{x}_i = [x_{i,1}\ \cdots\ x_{i,10}]^{\mathsf{T}}$ and $\mathbf{z}_i = [z_{i,1}\ \ z_{i,2}]^{\mathsf{T}}$ as the exam scores and common factors of the ith student, respectively. Also, denote $\mathbf{m} = [m_1\ \cdots\ m_{10}]^{\mathsf{T}}$ as the mean exam scores of these 10 subjects. Then, we have

$$x_{i,j} = m_j + v_{j,1}z_{i,1} + v_{j,2}z_{i,2} + \epsilon_{i,j},\quad i = 1,\ldots,1000 \text{ and } j = 1,\ldots,10, \tag{3.53}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

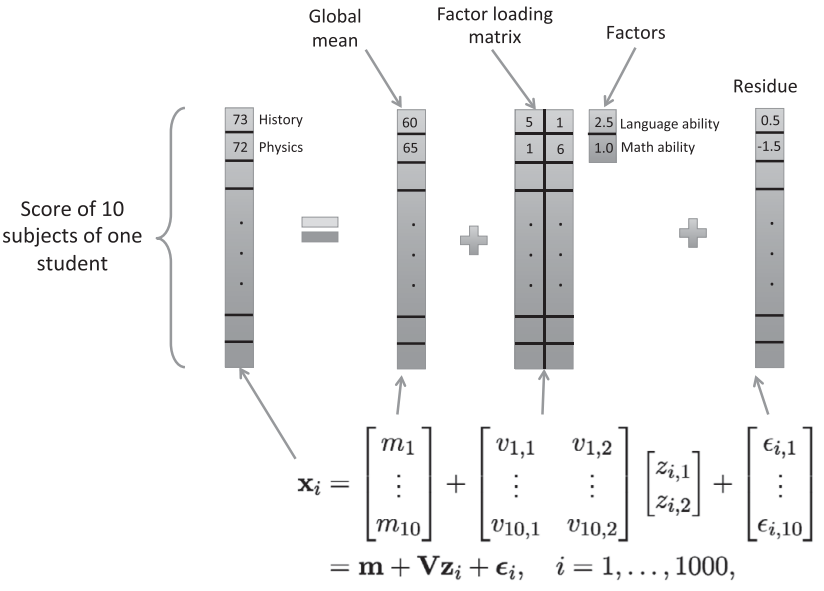


Figure 3.19 Illustrative example showing the idea of factor analysis. $\mathbf{x}_i$ represents the exam scores of ten subjects of a student who is strong in language but weak in math. The factor loading suggests that history demands language skill, whereas physics demands math skill.

where $v_{j,1}$ and $v_{j,2}$ are the factor loadings for the jth subject and $\epsilon_{i,j}$ is the error term. Eq. 3.53 can also be written as:

$$\begin{aligned} \mathbf{x}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{i,10} \end{bmatrix} &= \begin{bmatrix} m_1 \\ \vdots \\ m_{10} \end{bmatrix} + \begin{bmatrix} v_{1,1} & v_{1,2} \\ \vdots & \vdots \\ v_{10,1} & v_{10,2} \end{bmatrix} \begin{bmatrix} z_{i,1} \\ z_{i,2} \end{bmatrix} + \begin{bmatrix} \epsilon_{i,1} \\ \vdots \\ \epsilon_{i,10} \end{bmatrix} \\ &= \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, 1000, \end{aligned} \tag{3.54}$$

where $\mathbf{V}$ is a 10×2 matrix comprising the factor loadings of the 10 subjects.

Figure 3.19 shows the exam scores of history and physics of a student who is strong in language but weak in math (as reflected by the latent factors). The first two rows of the loading matrix suggest that history requires language skill, whereas physics requires math skill, because of the large *factor loadings*. Should the latent factors of this student be swapped (i.e., 1.0 for language ability and 2.5 for math ability), he/she will obtain very a high score in physics but very low score in history.

In the context of speaker recognition, one may consider the elements of a GMM-supervector as the observed variables and the coordinates in the subspace in which the supervector lives are the factors. To simplify the discussions, in this subsection, we start from the simple case where the UBM only have one Gaussian component (so that the supervector reduces to ordinary vectors) and then extend the simple case to the more general case where the UBM has C mixtures.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

3.3.1 Generative Model

Denote $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ as a set of R-dimensional vectors. In factor analysis, $\mathbf{x}_i$'s are assume to follow a linear-Gaussian model:

$$\mathbf{x}_i = \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, N, \tag{3.55}$$

where $\mathbf{m}$ is the global mean of vectors in $\mathcal{X}$, $\mathbf{V}$ is a low-rank $R \times D$ matrix, $\mathbf{z}_i$ is a D-dimensional latent vector with prior density $\mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_i$ is the residual noise following a Gaussian density with zero mean and covariance matrix $\boldsymbol{\Sigma}$. Figure 3.20 shows the graphical model of factor analysis.

Given the factor analysis model in Eq. 3.55 and the graphical model in Figure 3.20, it can be shown that the marginal density of $\mathbf{x}$ is given by

$$\begin{aligned} p(\mathbf{x}) &= \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \\ &= \int \mathcal{N}(\mathbf{x}|\mathbf{m} + \mathbf{V}\mathbf{z}, \boldsymbol{\Sigma})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})d\mathbf{z} \\ &= \mathcal{N}(\mathbf{x}|\mathbf{m}, \mathbf{V}\mathbf{V}^\mathsf{T} + \boldsymbol{\Sigma}). \end{aligned} \tag{3.56}$$

Eq. 3.56 can be obtained by convolution of Gaussian [44] or by noting that $p(\mathbf{x})$ is a Gaussian. For the latter, we take the expectation of $\mathbf{x}$ in Eq. 3.55 to obtain:

$$\mathbb{E}\{\mathbf{x}\} = \mathbf{m}.$$

Also, we take the expectation of $(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^\mathsf{T}$ in Eq. 3.55 to obtain:

$$\begin{aligned} \mathbb{E}\{(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^\mathsf{T}\} &= \mathbb{E}\{(\mathbf{V}\mathbf{z} + \boldsymbol{\epsilon})(\mathbf{V}\mathbf{z} + \boldsymbol{\epsilon})^\mathsf{T}\} \\ &= \mathbf{V}\mathbb{E}\{\mathbf{z}\mathbf{z}^\mathsf{T}\}\mathbf{V}^\mathsf{T} + \mathbb{E}\{\boldsymbol{\epsilon}\boldsymbol{\epsilon}^\mathsf{T}\} \\ &= \mathbf{V}\mathbf{I}\mathbf{V}^\mathsf{T} + \boldsymbol{\Sigma} \\ &= \mathbf{V}\mathbf{V}^\mathsf{T} + \boldsymbol{\Sigma}. \end{aligned} \tag{3.57}$$

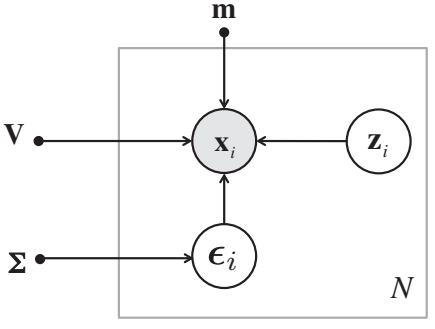


Figure 3.20 Graphical model of factor analysis.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Eq. 3.56 and Eq. 3.57 suggest that $\mathbf{x}$'s vary in a subspace of $\mathbb{R}^D$ with variability explained by the covariance matrix $\mathbf{V}\mathbf{V}^\mathsf{T}$. Any deviations away from this subspace are explained by $\boldsymbol{\Sigma}$.

3.3.2 EM Formulation

Because there is no closed-form solution for $\mathbf{V}$ and $\boldsymbol{\Sigma}$, they are estimated by the EM algorithm. Although each iteration of EM should be started with the E-step followed by the M-step, notationally, it is more convenient to describe the M-step first, assuming that the posterior expectations have already been computed in the E-step.

In the M-step, we maximize the expectation of the complete likelihood. Denote $\boldsymbol{\omega} = \{\mathbf{m}, \mathbf{V}, \boldsymbol{\Sigma}\}$ and $\boldsymbol{\omega}' = \{\mathbf{m}', \mathbf{V}', \boldsymbol{\Sigma}'\}$ as the old and new parameter sets, respectively. Also denote $\mathcal{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$ as the latent factors of the FA model in Eq. 3.55. The E- and M-steps iteratively maximize the expectation of the complete likelihood with respect to the latent variables (see Section 2.2.2):

$$\begin{aligned} Q(\boldsymbol{\omega}'|\boldsymbol{\omega}) &= \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}\{\log p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\omega}')|\mathcal{X}, \boldsymbol{\omega}\} \\ &= \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}\left\{\sum\nolimits_i \log\left[p\left(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\omega}'\right) p(\mathbf{z}_i)\right] \middle| \mathcal{X}, \boldsymbol{\omega}\right\} \\ &= \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}\left\{\sum\nolimits_i \log\left[\mathcal{N}\left(\mathbf{x}_i|\mathbf{m}' + \mathbf{V}'\mathbf{z}_i, \boldsymbol{\Sigma}'\right)\mathcal{N}(\mathbf{z}_i|\mathbf{0}, \mathbf{I})\right] \middle| \mathcal{X}, \boldsymbol{\omega}\right\}. \end{aligned} \tag{3.58}$$

To simplify notations, we drop the symbol ($'$) in Eq. 3.58, ignore the constant terms independent of the model parameters, and replace expectation notation $\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}\{\mathbf{z}|\mathbf{x}\}$ by $\langle \mathbf{z}|\mathbf{x}\rangle$, which results in

$$\begin{aligned} Q(\boldsymbol{\omega}) &= -\sum_i \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z}|\mathbf{x})}\left\{\frac{1}{2}\log|\boldsymbol{\Sigma}| + \frac{1}{2}(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{z}_i)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{z}_i)\middle|\mathcal{X}, \boldsymbol{\omega}\right\} \\ &= \sum_i\left[-\frac{1}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}(\mathbf{x}_i - \mathbf{m})^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m})\right] \\ &\quad + \sum_i(\mathbf{x}_i - \mathbf{m})^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\langle \mathbf{z}_i|\mathbf{x}_i\rangle - \frac{1}{2}\left[\sum_i\left\langle \mathbf{z}_i^\mathsf{T}\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\mathbf{z}_i|\mathbf{x}_i\right\rangle\right]. \end{aligned} \tag{3.59}$$

Using the following properties of matrix derivatives

$$\begin{aligned} \frac{\partial \mathbf{a}^\mathsf{T}\mathbf{X}\mathbf{b}}{\partial \mathbf{X}} &= \mathbf{a}\mathbf{b}^\mathsf{T} \\ \frac{\partial \mathbf{b}^\mathsf{T}\mathbf{X}^\mathsf{T}\mathbf{B}\mathbf{X}\mathbf{c}}{\partial \mathbf{X}} &= \mathbf{B}^\mathsf{T}\mathbf{X}\mathbf{b}\mathbf{c}^\mathsf{T} + \mathbf{B}\mathbf{X}\mathbf{c}\mathbf{b}^\mathsf{T} \\ \frac{\partial}{\partial \mathbf{A}}\log|\mathbf{A}^{-1}| &= -(\mathbf{A}^{-1})^\mathsf{T} \end{aligned}$$

we have

$$\frac{\partial Q}{\partial \mathbf{V}} = \sum_i \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m})\langle \mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\rangle - \sum_i \boldsymbol{\Sigma}^{-1}\mathbf{V}\left\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\right\rangle. \tag{3.60}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Setting $\frac{\partial Q}{\partial \mathbf{V}} = 0$, we have

$$\sum_i \mathbf{V}\left\langle \mathbf{z}_i \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \right\rangle = \sum_i (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \tag{3.61}$$

$$\mathbf{V} = \left[\sum_i (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i | \mathbf{x}_i \rangle^\mathsf{T} \right] \left[\sum_i \left\langle \mathbf{z}_i \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \right\rangle \right]^{-1}. \tag{3.62}$$

To find $\mathbf{\Sigma}$, we evaluate

$$\frac{\partial Q}{\partial \mathbf{\Sigma}^{-1}} = \frac{1}{2} \sum_i \left[\mathbf{\Sigma} - (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\mathsf{T} \right] + \sum_i (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \mathbf{V}^\mathsf{T} - \frac{1}{2} \sum_i \mathbf{V} \langle \mathbf{z}_i \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \mathbf{V}^\mathsf{T}.$$

Note that according to Eq. 3.61, we have

$$\frac{\partial Q}{\partial \mathbf{\Sigma}^{-1}} = \frac{1}{2} \sum_i \left[\mathbf{\Sigma} - (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\mathsf{T} \right] + \sum_i (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \mathbf{V}^\mathsf{T} - \frac{1}{2} \sum_i (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \mathbf{V}^\mathsf{T}.$$

Therefore, setting $\frac{\partial Q}{\partial \mathbf{\Sigma}^{-1}} = 0$ we have

$$\sum_i \mathbf{\Sigma} = \sum_i \left[(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\mathsf{T} - (\mathbf{x}_i - \mathbf{m}) \langle \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle \mathbf{V}^\mathsf{T} \right].$$

Rearranging, we have

$$\mathbf{\Sigma} = \frac{1}{N} \sum_i \left[(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^\mathsf{T} - \mathbf{V} \langle \mathbf{z}_i | \mathbf{x}_i \rangle (\mathbf{x}_i - \mathbf{m})^\mathsf{T} \right].$$

To compute $\mathbf{m}$, we evaluate

$$\frac{\partial Q}{\partial \mathbf{m}} = -\sum_i \left(\mathbf{\Sigma}^{-1} \mathbf{m} - \mathbf{\Sigma}^{-1} \mathbf{x}_i \right) + \sum_i \mathbf{\Sigma}^{-1} \mathbf{V} \langle \mathbf{z}_i | \mathbf{x}_i \rangle.$$

Setting $\frac{\partial Q}{\partial \mathbf{m}} = 0$, we have

$$\mathbf{m} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_i,$$

where we have used the property $\sum_i \langle \mathbf{z}_i | \mathbf{x}_i \rangle \approx \mathbf{0}$ when N is sufficiently large. This property arises from the assumption that the prior of $\mathbf{z}$ follows a Gaussian distribution, i.e., $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Alternatively, we may take the expectation of $\langle \mathbf{z}_i | \mathbf{x}_i \rangle$ in Eq. 3.65(a) to get this result.

In the E-step, we compute the posterior means $\langle \mathbf{z}_i | \mathbf{x}_i \rangle$ and posterior moments $\langle \mathbf{z}_i \mathbf{z}_i^\mathsf{T} | \mathbf{x}_i \rangle$. Let's express the following posterior density in terms of its likelihood and prior [45]:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$
\begin{aligned}
& p(\mathbf{z}_i|\mathbf{x}_i, \boldsymbol{\omega}) \\
& \propto p(\mathbf{x}_i|\mathbf{z}_i, \boldsymbol{\omega}) p(\mathbf{z}_i) \\
& \propto \exp\left\{-\frac{1}{2}(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{z}_i)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m} - \mathbf{V}\mathbf{z}_i) - \frac{1}{2}\mathbf{z}_i^\mathsf{T}\mathbf{z}_i\right\} \\
& \propto \exp\left\{\mathbf{z}_i^\mathsf{T}\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m}) - \frac{1}{2}\mathbf{z}_i^\mathsf{T}\left(\mathbf{I} + \mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\right)\mathbf{z}_i\right\}.
\end{aligned}
\tag{3.63}
$$

Consider the following property of Gaussian distribution with mean $\boldsymbol{\mu}_z$ and covariance $\mathbf{C}_z$

$$
\begin{aligned}
\mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_z, \mathbf{C}_z) &\propto \exp\left\{-\frac{1}{2}(\mathbf{z} - \boldsymbol{\mu}_z)^\mathsf{T}\mathbf{C}_z^{-1}(\mathbf{z} - \boldsymbol{\mu}_z)\right\} \\
&\propto \exp\left\{\mathbf{z}^\mathsf{T}\mathbf{C}_z^{-1}\boldsymbol{\mu}_z - \frac{1}{2}\mathbf{z}^\mathsf{T}\mathbf{C}_z^{-1}\mathbf{z}\right\}.
\end{aligned}
\tag{3.64}
$$

Comparing Eq. 3.63 and Eq. 3.64, we obtain the posterior mean and moment as follows:

$$
\begin{aligned}
\langle \mathbf{z}_i|\mathbf{x}_i\rangle &= \mathbf{L}^{-1}\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m}) \\
\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\rangle &= \mathbf{L}^{-1} + \langle \mathbf{z}_i|\mathbf{x}_i\rangle\langle \mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\rangle,
\end{aligned}
$$

where $\mathbf{L}^{-1} = (\mathbf{I} + \mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V})^{-1}$ is the posterior covariance matrix of $\mathbf{z}_i, \forall i$.

In summary, we have the following EM algorithm for factor analysis:

E-step:

$$\langle \mathbf{z}_i|\mathbf{x}_i\rangle = \mathbf{L}^{-1}\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \mathbf{m}) \tag{3.65a}$$

$$\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\rangle = \mathbf{L}^{-1} + \langle \mathbf{z}_i|\mathbf{x}_i\rangle\langle \mathbf{z}_i|\mathbf{x}_i\rangle^\mathsf{T} \tag{3.65b}$$

$$\mathbf{L} = \mathbf{I} + \mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V} \tag{3.65c}$$

M-step:

$$\mathbf{m}' = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \tag{3.65d}$$

$$\mathbf{V}' = \left[\sum_{i=1}^{N}(\mathbf{x}_i - \mathbf{m}')\langle \mathbf{z}_i|\mathbf{x}_i\rangle^\mathsf{T}\right]\left[\sum_{i=1}^{N}\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathbf{x}_i\rangle\right]^{-1} \tag{3.65e}$$

$$\boldsymbol{\Sigma}' = \frac{1}{N}\left\{\sum_{i=1}^{N}\left[(\mathbf{x}_i - \mathbf{m}')(\mathbf{x}_i - \mathbf{m}')^\mathsf{T} - \mathbf{V}'\langle \mathbf{z}_i|\mathbf{x}_i\rangle(\mathbf{x}_i - \mathbf{m}')^\mathsf{T}\right]\right\} \tag{3.65f}$$

Note that Eq. 3.65(a) and the Eq. 3.65(c) are the posterior mean and posterior precision of the Bayesian general linear model [46, Eq.10.32 and Eq. 10.34].

3.3.3 Relationship with Principal Component Analysis

Both principal component analysis (PCA) and factor analysis (FA) are dimension reduction techniques. But they are fundamentally different. In particular, components in PCA are orthogonal to each other, whereas factor analysis does not require the column vectors

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

in the loading matrix to be orthogonal, i.e., the correlation between the factors could be nonzero. Also, PCA finds a linear combination of the observed variables to preserve the variance of the data, whereas FA predicts observed variables from theoretical latent factors. Mathematically, if $\mathbf{W} = [\boldsymbol{w}_1\ \mathbf{w}_2\ \cdots\ \mathbf{w}_M]$ is a PCA projection matrix in which $\mathbf{w}_j$'s are the eigenvectors of the data covariance matrix and $\mathbf{V}$ is an FA loading matrix, we have

$$\begin{aligned} \text{PCA}: \ \mathbf{y}_i &= \mathbf{W}^{\mathsf{T}}(\mathbf{x}_i - \mathbf{m}) \text{ and } \hat{\mathbf{x}}_i = \mathbf{m} + \mathbf{W}\mathbf{y}_i \\ \text{FA}: \ \mathbf{x}_i &= \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_i, \end{aligned} \tag{3.66}$$

where $\mathbf{x}_i$'s are observed vectors, $\hat{\mathbf{x}}$'s are PCA reconstructed vectors, $\mathbf{y}_i$'s are PCA-projected vectors, $\mathbf{z}_i$'s are factors, and $\boldsymbol{\epsilon}_i$' are residues.

Under specific conditions, FA becomes PCA. Consider the special situation in which the covariance matrix of $\boldsymbol{\epsilon}$ is diagonal and all elements in the diagonal are equals, i.e., $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$. Then, Eq. 3.65(c) and Eq. 3.65(a) become

$$\mathbf{L} = \mathbf{I} + \frac{1}{\sigma^2}\mathbf{V}^{\mathsf{T}}\mathbf{V} \quad \text{and} \quad \langle \mathbf{z}_i | \mathbf{x}_i \rangle = \frac{1}{\sigma^2}\mathbf{L}^{-1}\mathbf{V}^{\mathsf{T}}(\mathbf{x}_i - \mathbf{m}),$$

respectively. When $\sigma^2 \to 0$ and $\mathbf{V}$ is an orthogonal matrix, then $\mathbf{L} \to \frac{1}{\sigma^2}\mathbf{V}^{\mathsf{T}}\mathbf{V}$ and

$$\begin{aligned} \langle \mathbf{z}_i | \mathbf{x}_i \rangle &\to \frac{1}{\sigma^2}\sigma^2(\mathbf{V}^{\mathsf{T}}\mathbf{V})^{-1}\mathbf{V}^{\mathsf{T}}(\mathbf{x}_i - \mathbf{m}) \\ &= \mathbf{V}^{-1}(\mathbf{V}^{\mathsf{T}})^{-1}\mathbf{V}^{\mathsf{T}}(\mathbf{x}_i - \mathbf{m}) \\ &= \mathbf{V}^{\mathsf{T}}(\mathbf{x}_i - \mathbf{m}). \end{aligned} \tag{3.67}$$

Note that Eq. 3.67 has the same form as the PCA projection in Eq. 3.66 and that the posterior covariance ($\mathbf{L}^{-1}$) of $\mathbf{z}$ becomes 0.

The analysis above suggests that PCA is a specific case of factor analysis in which the covariances of residue noise $\boldsymbol{\epsilon}$ collapse to zero. This means that in PCA, for a given $\mathbf{x}_i$ and $\mathbf{W}$ in Eq. 3.66, there is a *deterministic* $\mathbf{y}_i$. On the other hand, in factor analysis, an $\mathbf{x}_i$ can be generated by infinite combinations of $\mathbf{z}_i$ and $\boldsymbol{\epsilon}_i$, as evident in Eq. 3.56.

The covariance matrix ($\mathbf{V}\mathbf{V}^{\mathsf{T}} + \boldsymbol{\Sigma}$) of observed data in FA suggests that FA treats covariances and variances of observed data separately. More specifically, the D column vectors (factor loadings) of $\mathbf{V}$ in FA capture most of the covariances while the variances of the individual components of $\mathbf{x}$'s are captured by the diagonal elements of $\boldsymbol{\Sigma}$. Note that the diagonal elements in $\boldsymbol{\Sigma}$ can be different, providing extra flexibility in modeling variances. On the other hand, because $\boldsymbol{\Sigma} \to \mathbf{0}$ in PCA, the first M principal components attempt to capture most of the variabilities, including covariances and variances. As a result, FA is more flexible in terms of data modeling.

3.3.4 Relationship with Nuisance Attribute Projection

If we write the projected vectors in Eq. 3.43 as follows:

$$\begin{aligned} \overrightarrow{\boldsymbol{\mu}}^{(s)} &= \tilde{\boldsymbol{\mu}}^{(s)} + \mathbf{U}(\mathbf{U}^{\mathsf{T}}\overrightarrow{\boldsymbol{\mu}}^{(s)}) \\ &= \tilde{\boldsymbol{\mu}}^{(s)} + \mathbf{U}\mathbf{x}^{(s)}, \end{aligned} \tag{3.68}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where the superscript s stands for speaker s and $\mathbf{x}^{(s)} \equiv \mathbf{U}^{\mathsf{T}} \overrightarrow{\boldsymbol{\mu}}^{(s)}$, then we may consider NAP as a special case of factor analysis in which the loading matrix $\mathbf{U}$ is determined by minimizing within-speaker covariance and $\mathbf{x}^{(s)}$ is the factor that gives the appropriate offset to produce $\overrightarrow{\boldsymbol{\mu}}^{(s)}$. This factor analysis interpretation suggests that $\tilde{\boldsymbol{\mu}}^{(s)}$ is considered fixed for speaker s and that there is no intra-speaker variability for speaker s, i.e., all variability in $\overrightarrow{\boldsymbol{\mu}}^{(s)}$ is due to the channel. As a result, NAP does not consider the prior density of $\mathbf{x}^{(s)}$ and does not have a speaker subspace to model inter-speaker variability.

3.4 Probabilistic Linear Discriminant Analysis

3.4.1 Generative Model

The Gaussian probabilistic LDA (PLDA) is the supervised version of FA. Consider a data set comprising D-dimensional vectors $\mathcal{X} = \{\mathbf{x}_{ij}; i = 1, \ldots, N; j = 1, \ldots, H_i\}$ obtained from N classes such that the ith class contains H_i vectors (e.g., i-vectors). The i-vector $\mathbf{x}_{ij}$ is assumed to be generated by a factor analysis model:

$$\mathbf{x}_{ij} = \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_{ij}, \tag{3.69}$$

where $\mathbf{m}$ is the global mean of all i-vectors, $\mathbf{V}$ is the speaker loading matrix, $\mathbf{z}_i$ is the speaker factor (of the ith speaker), and $\boldsymbol{\epsilon}_{ij}$ is the residue that could not be captured by the factor analysis model.

The class-specific vectors should share the same latent factor:

$$\tilde{\mathbf{x}}_i = \tilde{\mathbf{m}} + \tilde{\mathbf{V}}\mathbf{z}_i + \tilde{\boldsymbol{\epsilon}}_i, \qquad \tilde{\mathbf{x}}_i, \tilde{\mathbf{m}} \in \mathbb{R}^{DH_i}, \ \tilde{\mathbf{V}} \in \mathbb{R}^{DH_i \times M}, \ \tilde{\boldsymbol{\epsilon}}_i \in \mathbb{R}^{DH_i}, \tag{3.70}$$

where $\tilde{\mathbf{x}}_i = [\mathbf{x}_{i1}^{\mathsf{T}} \ \cdots \ \mathbf{x}_{iH_i}^{\mathsf{T}}]^{\mathsf{T}}$, $\tilde{\mathbf{m}} = [\mathbf{m}^{\mathsf{T}} \ \cdots \ \mathbf{m}^{\mathsf{T}}]^{\mathsf{T}}$, $\tilde{\mathbf{V}} = [\mathbf{V}^{\mathsf{T}} \ \cdots \ \mathbf{V}^{\mathsf{T}}]^{\mathsf{T}}$, and $\tilde{\boldsymbol{\epsilon}}_i = [\boldsymbol{\epsilon}_{i1}^{\mathsf{T}} \ \cdots \ \boldsymbol{\epsilon}_{iH_i}^{\mathsf{T}}]^{\mathsf{T}}$. Figure 3.21 shows the graphical representation of a Gaussian PLDA model. Note that the key difference between the PLDA model in Figure 3.21 and the FA model in Figure 3.20 is that in the former, for each i we have H_i observed vectors; whereas for the latter, each i is associated with one observed vector only.

The E- and M-steps iteratively evaluate and maximize the expectation of the complete likelihood:

$$\begin{aligned} Q(\boldsymbol{\omega}'|\boldsymbol{\omega}) &= \mathbb{E}_{\mathcal{Z}}\{\ln p(\mathcal{X}, \mathcal{Z}|\boldsymbol{\omega}')|\mathcal{X}, \boldsymbol{\omega}\} \\ &= \mathbb{E}_{\mathcal{Z}}\left\{\sum\nolimits_{ij} \ln\left[p\left(\mathbf{x}_{ij}|\mathbf{z}_i, \boldsymbol{\omega}'\right) p(\mathbf{z}_i)\right] \middle| \mathcal{X}, \boldsymbol{\omega}\right\} \\ &= \mathbb{E}_{\mathcal{Z}}\left\{\sum\nolimits_{ij} \ln\left[\mathcal{N}\left(\mathbf{x}_{ij}|\mathbf{m}' + \mathbf{V}'\mathbf{z}_i, \boldsymbol{\Sigma}'\right) \mathcal{N}(\mathbf{z}_i|\mathbf{0}, \mathbf{I})\right] \middle| \mathcal{X}, \boldsymbol{\omega}\right\}, \end{aligned} \tag{3.71}$$

where the notation $\sum_{ij}$ is a shorthand form of $\sum_{i=1}^{N} \sum_{j=1}^{H_i}$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

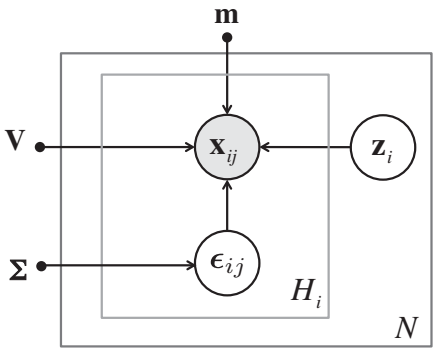


Figure 3.21 Graphical representation of a simplified Gaussian PLDA model in which the channel variability is modeled by the full covariance matrix $\mathbf{\Sigma}$. The prior of $\mathbf{z}_i$ follows a standard Gaussian distribution.

3.4.2 EM Formulations

Similar to Section 3.3.2, the auxiliary function can be simplified to

$$\begin{aligned} Q(\boldsymbol{\omega}) &= -\sum_{ij} \mathbb{E}_{\mathcal{Z}} \left\{ \frac{1}{2} \log |\mathbf{\Sigma}| + \frac{1}{2} (\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i)^{\mathsf{T}} \mathbf{\Sigma}^{-1} (\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i) \right\} \\ &= \sum_{ij} \left[-\frac{1}{2} \log |\mathbf{\Sigma}| - \frac{1}{2} (\mathbf{x}_{ij} - \mathbf{m})^{\mathsf{T}} \mathbf{\Sigma}^{-1} (\mathbf{x}_{ij} - \mathbf{m}) \right] \\ &\quad + \sum_{ij} (\mathbf{x}_{ij} - \mathbf{m})^{\mathsf{T}} \mathbf{\Sigma}^{-1} \mathbf{V} \langle \mathbf{z}_i | \mathcal{X}_i \rangle - \frac{1}{2} \left[\sum_{ij} \left\langle \mathbf{z}_i^{\mathsf{T}} \mathbf{V}^{\mathsf{T}} \mathbf{\Sigma}^{-1} \mathbf{V} \mathbf{z}_i | \mathcal{X}_i \right\rangle \right]. \end{aligned} \tag{3.72}$$

Taking the derivative of Q with respect to $\mathbf{V}$, we have

$$\frac{\partial Q}{\partial \mathbf{V}} = \sum_{ij} \mathbf{\Sigma}^{-1} (\mathbf{x}_{ij} - \mathbf{m}) \langle \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \rangle - \sum_{ij} \mathbf{\Sigma}^{-1} \mathbf{V} \left\langle \mathbf{z}_i \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \right\rangle. \tag{3.73}$$

Setting $\frac{\partial Q}{\partial \mathbf{V}} = 0$, we have

$$\sum_{ij} \mathbf{V} \left\langle \mathbf{z}_i \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \right\rangle = \sum_{ij} (\mathbf{x}_{ij} - \mathbf{m}) \langle \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \rangle \tag{3.74}$$

$$\mathbf{V} = \left[\sum_{ij} (\mathbf{x}_{ij} - \mathbf{m}) \langle \mathbf{z}_i | \mathcal{X}_i \rangle^{\mathsf{T}} \right] \left[\sum_{ij} \left\langle \mathbf{z}_i \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \right\rangle \right]^{-1}. \tag{3.75}$$

To find $\mathbf{\Sigma}$, we evaluate

$$\begin{aligned} \frac{\partial Q}{\partial \mathbf{\Sigma}^{-1}} &= \frac{1}{2} \sum_{ij} \left[\mathbf{\Sigma} - (\mathbf{x}_{ij} - \mathbf{m})(\mathbf{x}_{ij} - \mathbf{m})^{\mathsf{T}} \right] + \sum_{ij} (\mathbf{x}_{ij} - \mathbf{m}) \langle \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \rangle \mathbf{V}^{\mathsf{T}} \\ &\quad - \frac{1}{2} \sum_{ij} \mathbf{V} \langle \mathbf{z}_i \mathbf{z}_i^{\mathsf{T}} | \mathcal{X}_i \rangle \mathbf{V}^{\mathsf{T}}. \end{aligned}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Note that according to Eq. 3.61, we have

$$\frac{\partial Q}{\partial \boldsymbol{\Sigma}^{-1}} = \frac{1}{2}\sum_{ij}\left[\boldsymbol{\Sigma} - (\mathbf{x}_{ij}-\mathbf{m})(\mathbf{x}_{ij}-\mathbf{m})^{\mathsf{T}}\right] + \sum_{ij}(\mathbf{x}_{ij}-\mathbf{m})\langle \mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\rangle\mathbf{V}^{\mathsf{T}} - \frac{1}{2}\sum_{ij}(\mathbf{x}_{ij}-\mathbf{m})\langle \mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\rangle\mathbf{V}^{\mathsf{T}}.$$

Therefore, setting $\frac{\partial Q}{\partial \boldsymbol{\Sigma}^{-1}} = 0$ we have

$$\sum_{ij}\boldsymbol{\Sigma} = \sum_{ij}\left[(\mathbf{x}_{ij}-\mathbf{m})(\mathbf{x}_{ij}-\mathbf{m})^{\mathsf{T}} - (\mathbf{x}_{ij}-\mathbf{m})\langle \mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\rangle\mathbf{V}^{\mathsf{T}}\right].$$

Rearranging, we have

$$\boldsymbol{\Sigma} = \frac{1}{\sum_{ij}1}\sum_{ij}\left[(\mathbf{x}_{ij}-\mathbf{m})(\mathbf{x}_{ij}-\mathbf{m})^{\mathsf{T}} - \mathbf{V}\langle \mathbf{z}_i|\mathcal{X}_i\rangle(\mathbf{x}_{ij}-\mathbf{m})^{\mathsf{T}}\right].$$

To compute $\mathbf{m}$, we evaluate

$$\frac{\partial Q}{\partial \mathbf{m}} = -\sum_{ij}\left(\boldsymbol{\Sigma}^{-1}\mathbf{m} - \boldsymbol{\Sigma}^{-1}\mathbf{x}_{ij}\right) + \sum_{ij}\boldsymbol{\Sigma}^{-1}\mathbf{V}\langle \mathbf{z}_i|\mathcal{X}_i\rangle.$$

Setting $\frac{\partial Q}{\partial \mathbf{m}} = 0$, we have

$$\mathbf{m} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{H_i}\mathbf{x}_{ij}}{\sum_{i=1}^{N}\sum_{j=1}^{H_i}1},$$

where we have used the property $\sum_i \langle \mathbf{z}_i|\mathcal{X}_i\rangle \approx \mathbf{0}$ when the number of training speakers is sufficiently large. This property arises from the assumption that the factors $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$.

In the E-step, we compute the posterior means $\langle \mathbf{z}_i|\mathcal{X}_i\rangle$ and posterior moments $\langle \mathbf{z}_i\mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\rangle$, where $\mathcal{X}_i$ denotes the i-vectors of speaker i. Let's express the following posterior density in terms of its likelihood and prior:

$$\begin{aligned} & p(\mathbf{z}_i|\mathbf{x}_{i1},\ldots,\mathbf{x}_{iH_i},\boldsymbol{\omega}) \\ & \propto \prod_{j=1}^{H_i} p(\mathbf{x}_{ij}|\mathbf{z}_i,\boldsymbol{\omega})p(\mathbf{z}_i) \\ & \propto \exp\left\{-\frac{1}{2}\sum_{j}(\mathbf{x}_{ij}-\mathbf{m}-\mathbf{V}\mathbf{z}_i)^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij}-\mathbf{m}-\mathbf{V}\mathbf{z}_i) - \frac{1}{2}\mathbf{z}_i^{\mathsf{T}}\mathbf{z}_i\right\} \\ & = \exp\left\{\mathbf{z}_i^{\mathsf{T}}\mathbf{V}^{\mathsf{T}}\sum_{j}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij}-\mathbf{m}) - \frac{1}{2}\mathbf{z}_i^{\mathsf{T}}\left(\mathbf{I}+\sum_{j}\mathbf{V}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{V}\right)\mathbf{z}_i\right\}. \end{aligned} \tag{3.76}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Consider the following property of Gaussian distribution with mean μ_z and covariance $\mathbf{C}_z$

$$\begin{aligned}\mathcal{N}(\mathbf{z}|\mu_z, \mathbf{C}_z) &\propto \exp\left\{-\frac{1}{2}(\mathbf{z}-\mu_z)^\mathsf{T}\mathbf{C}_z^{-1}(\mathbf{z}-\mu_z)\right\} \\ &\propto \exp\left\{\mathbf{z}^\mathsf{T}\mathbf{C}_z^{-1}\mu_z - \frac{1}{2}\mathbf{z}^\mathsf{T}\mathbf{C}_z^{-1}\mathbf{z}\right\}.\end{aligned} \tag{3.77}$$

Comparing Eq. 3.76 and Eq. 3.77, we obtain the posterior mean and moment as follows:

$$\langle \mathbf{z}_i|\mathcal{X}_i\rangle = \mathbf{L}_i^{-1}\mathbf{V}^\mathsf{T}\sum_{j=1}^{H_i}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij}-\mathbf{m}) \tag{3.78}$$

$$\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathcal{X}_i\rangle = \mathbf{L}_i^{-1} + \langle \mathbf{z}_i|\mathcal{X}_i\rangle\langle \mathbf{z}_i^\mathsf{T}|\mathcal{X}_i\rangle \tag{3.79}$$

where $\mathbf{L}_i = \mathbf{I} + \sum_j \mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}$.

In summary, the EM algorithm for PLDA is as follows [47]:

E-step:

$$\begin{aligned}&\langle \mathbf{z}_i|\mathcal{X}_i\rangle = \mathbf{L}_i^{-1}\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\sum\nolimits_{j=1}^{H_i}(\mathbf{x}_{ij}-\mathbf{m}) \\ &\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathcal{X}_i\rangle = \mathbf{L}_i^{-1} + \langle \mathbf{z}_i|\mathcal{X}_i\rangle\langle \mathbf{z}_i|\mathcal{X}_i\rangle^\mathsf{T} \\ &\mathbf{L}_i = \mathbf{I} + H_i\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\end{aligned}$$

M-step:

$$\begin{aligned}&\mathbf{V}' = \left[\sum_{i=1}^{N}\sum_{j=1}^{H_i}(\mathbf{x}_{ij}-\mathbf{m}')\langle \mathbf{z}_i|\mathcal{X}_i\rangle^\mathsf{T}\right]\left[\sum_{i=1}^{N}\sum_{j=1}^{H_i}\langle \mathbf{z}_i\mathbf{z}_i^\mathsf{T}|\mathcal{X}_i\rangle\right]^{-1} \\ &\mathbf{m}' = \frac{\sum_{ij}\mathbf{x}_{ij}}{\sum_i H_i} \\ &\boldsymbol{\Sigma}' = \frac{1}{\sum_{i=1}^{N}H_i}\left\{\sum_{i=1}^{N}\sum_{j=1}^{H_i}\left[(\mathbf{x}_{ij}-\mathbf{m}')(\mathbf{x}_{ij}-\mathbf{m}')^\mathsf{T} - \mathbf{V}'\langle \mathbf{z}_i|\mathcal{X}_i\rangle(\mathbf{x}_{ij}-\mathbf{m}')^\mathsf{T}\right]\right\}\end{aligned} \tag{3.80}$$

3.4.3 PLDA Scoring

PLDA is by far the most popular back end for speaker verification. Typically, it accepts an i-vector (see Section 3.6) pair or more recently x-vector (see Section 5.2.1) pairs as input and produces a score that quantifies how likely the pair comes from the same speaker against the likelihood that the pair is from two different speakers.

Given a test i-vector $\mathbf{x}_t$ and target-speaker's i-vector $\mathbf{x}_s$, we need to test the hypothesis H_0 that both i-vectors come from the same speaker against the alternative hypothesis that they belong to different speakers. Using the notations in Eq. 3.55, we have

$$H_0 : \text{Same speaker} \begin{cases} \mathbf{x}_s = \mathbf{m} + \mathbf{V}\mathbf{z} + \boldsymbol{\epsilon}_s \\ \mathbf{x}_t = \mathbf{m} + \mathbf{V}\mathbf{z} + \boldsymbol{\epsilon}_t \end{cases} \tag{3.81}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$H_1 : \text{Different speakers} \begin{cases} \mathbf{x}_s = \mathbf{m} + \mathbf{V}\mathbf{z}_s + \boldsymbol{\epsilon}_s \\ \mathbf{x}_t = \mathbf{m} + \mathbf{V}\mathbf{z}_t + \boldsymbol{\epsilon}_t \end{cases} \tag{3.82}$$

Note that the speaker factor $\mathbf{z}$ is the same for both i-vectors in H_0, whereas they are different in H_1. This hypothesis testing problem can be solved by evaluating the likelihood ratio score:

$$\begin{aligned} S_{\text{LR}}(\mathbf{x}_s, \mathbf{x}_t) &= \frac{p(\mathbf{x}_s, \mathbf{x}_t | \text{same-speaker})}{p(\mathbf{x}_s, \mathbf{x}_t | \text{different-speakers})} \\ &= \frac{\int p(\mathbf{x}_s, \mathbf{x}_t, \mathbf{z} | \boldsymbol{\omega}) \mathrm{d}\mathbf{z}}{\int p(\mathbf{x}_s, \mathbf{z}_s | \boldsymbol{\omega}) \mathrm{d}\mathbf{z}_s \int p(\mathbf{x}_t, \mathbf{z}_t | \boldsymbol{\omega}) \mathrm{d}\mathbf{z}_t} \\ &= \frac{\int p(\mathbf{x}_s, \mathbf{x}_t | \mathbf{z}, \boldsymbol{\omega}) p(\mathbf{z}) \mathrm{d}\mathbf{z}}{\int p(\mathbf{x}_s | \mathbf{z}_s, \boldsymbol{\omega}) p(\mathbf{z}_s) \mathrm{d}\mathbf{z}_s \int p(\mathbf{x}_t | \mathbf{z}_t, \boldsymbol{\omega}) p(\mathbf{z}_t) \mathrm{d}\mathbf{z}_t} \\ &= \frac{\mathcal{N}\left(\begin{bmatrix} \mathbf{x}_s^\mathsf{T} & \mathbf{x}_t^\mathsf{T} \end{bmatrix}^\mathsf{T} \middle| \hat{\mathbf{m}}, \hat{\boldsymbol{\Lambda}} \right)}{\mathcal{N}\left(\begin{bmatrix} \mathbf{x}_s^\mathsf{T} & \mathbf{x}_t^\mathsf{T} \end{bmatrix}^\mathsf{T} \middle| \hat{\mathbf{m}}, \text{diag}\{\boldsymbol{\Lambda}, \boldsymbol{\Lambda}\} \right)} \end{aligned} \tag{3.83}$$

where $\hat{\mathbf{m}} = \begin{bmatrix} \mathbf{m}^\mathsf{T}, \mathbf{m}^\mathsf{T} \end{bmatrix}^\mathsf{T}$, $\hat{\boldsymbol{\Lambda}} = \hat{\mathbf{V}}\hat{\mathbf{V}}^\mathsf{T} + \hat{\boldsymbol{\Sigma}}$, $\hat{\mathbf{V}} = \begin{bmatrix} \mathbf{V}^\mathsf{T} & \mathbf{V}^\mathsf{T} \end{bmatrix}^\mathsf{T}$, $\hat{\boldsymbol{\Sigma}} = \text{diag}\{\boldsymbol{\Sigma}, \boldsymbol{\Sigma}\}$, and $\boldsymbol{\Lambda} = \mathbf{V}\mathbf{V}^\mathsf{T} + \boldsymbol{\Sigma}$. Because both the numerator and denominator of Eq. 3.83 are Gaussians, it can be simplified as follows:

$$\begin{aligned} \log S_{\text{LR}}(\mathbf{x}_s, \mathbf{x}_t) = &\log \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} \middle| \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right) \\ &- \log \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} \middle| \begin{bmatrix} \mathbf{m} \\ \mathbf{m} \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{tot} \end{bmatrix} \right), \end{aligned} \tag{3.84}$$

where $\boldsymbol{\Sigma}_{tot} = \mathbf{V}\mathbf{V}^\mathsf{T} + \boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}_{ac} = \mathbf{V}\mathbf{V}^\mathsf{T}$. Since $\mathbf{m}$ is a global offset that can be precomputed and removed from all i-vectors, we set it to zero and expand Eq. 3.84 to obtain:

$$\begin{aligned} \log S_{\text{LR}}(\mathbf{x}_s, \mathbf{x}_t) &= -\frac{1}{2} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} + \frac{1}{2} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & 0 \\ 0 & \boldsymbol{\Sigma}_{tot} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} + \text{const} \\ &= -\frac{1}{2} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & -\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} \\ -(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1} & (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} \\ &\quad + \frac{1}{2} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} \boldsymbol{\Sigma}_{tot}^{-1} & 0 \\ 0 & \boldsymbol{\Sigma}_{tot}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} + \text{const} \\ &= \frac{1}{2} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix}^\mathsf{T} \begin{bmatrix} \boldsymbol{\Sigma}_{tot}^{-1} - (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} \\ \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & \boldsymbol{\Sigma}_{tot}^{-1} - (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} + \text{const} \\ &= \frac{1}{2} \begin{bmatrix} \mathbf{x}_s^\mathsf{T} & \mathbf{x}_t^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbf{Q} & \mathbf{P} \\ \mathbf{P} & \mathbf{Q} \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_t \end{bmatrix} + \text{const} \\ &= \frac{1}{2} [\mathbf{x}_s^\mathsf{T}\mathbf{Q}\mathbf{x}_s + \mathbf{x}_s^\mathsf{T}\mathbf{P}\mathbf{x}_t + \mathbf{x}_t^\mathsf{T}\mathbf{P}\mathbf{x}_s + \mathbf{x}_t^\mathsf{T}\mathbf{Q}\mathbf{x}_t] + \text{const} \\ &= \frac{1}{2} [\mathbf{x}_s^\mathsf{T}\mathbf{Q}\mathbf{x}_s + 2\mathbf{x}_s^\mathsf{T}\mathbf{P}\mathbf{x}_t + \mathbf{x}_t^\mathsf{T}\mathbf{Q}\mathbf{x}_t] + \text{const}, \end{aligned} \tag{3.85}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where

$$\begin{aligned}\mathbf{Q} &= \boldsymbol{\Sigma}_{tot}^{-1} - (\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}, \\ \mathbf{P} &= \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot} - \boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}.\end{aligned} \tag{3.86}$$

Eq. 3.83 suggests that instead of using a point estimate of the latent variable $\mathbf{z}$, the verification process marginalizes the joint density of the latent and observed variables over the latent variables. Instead of the actual value of the latent variable, the verification process considers the likelihood that the two i-vectors share the same latent variable [17]. Therefore, the verification score can be interpreted as the likelihood ratio between the test and target speakers sharing the same latent variable and the likelihood that they have different latent variables. As a result, a large score suggests that the two i-vectors come from the same speaker (i.e., sharing the same latent variable) rather than coming from two different speakers (i.e., having two distinct latent variables). Eq. 3.85 gives the PLDA score when there is only one enrollment i-vector per target speaker. When there are multiple enrollment utterances per target speaker, we may either average the corresponding enrollment i-vectors (so-called i-vector averaging) or average the scores obtained by applying Eq. 3.85 to each of the enrollment i-vectors (so-called score averaging).

While research has shown that in practice, i-vector averaging is a simple and effective way to process multiple enrollment utterances [48], a more elegant approach is to derive the exact likelihood ratio:

$$\begin{aligned}S_{\text{LR}}(\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,R}, \mathbf{x}_t) &= \frac{p(\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,R}, \mathbf{x}_t | H_0)}{p(\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,R}, \mathbf{x}_t | H_1)} \\ &= \frac{p(\mathcal{X}_s, \mathbf{x}_t | H_0)}{p(\mathcal{X}_s | H_1) p(\mathbf{x}_t | H_1)} \\ &= \frac{p(\mathbf{x}_t | \mathcal{X}_s)}{p(\mathbf{x}_t)},\end{aligned} \tag{3.87}$$

where $\mathcal{X}_s = \{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,R}\}$ are R enrollment i-vectors from target-speaker s and $\mathbf{x}_t$ is an i-vector from a test speaker. Note that the numerator of Eq. 3.87 is a conditional likelihood [49]

$$p(\mathbf{x}_t | \mathcal{X}_s) = \mathcal{N}\left(\mathbf{x}_t \middle| \mathbf{m} + \mathbf{V}\langle \mathbf{z}_s | \mathcal{X}_s \rangle, \mathbf{V}\mathbf{L}_s^{-1}\mathbf{V}^{\mathsf{T}} + \boldsymbol{\Sigma}\right) \tag{3.88}$$

where $\langle \mathbf{z}_s | \mathcal{X}_s \rangle$ and $\mathbf{L}_s$ are the posterior mean and posterior precision given by

$$\langle z_s | \mathcal{X}_s \rangle = \mathbf{L}_s^{-1}\mathbf{V}\boldsymbol{\Sigma}^{-1}\sum_{r=1}^{R}(\mathbf{x}_{s,r} - \mathbf{m}) \tag{3.89a}$$

$$\mathbf{L}_s = \mathbf{I} + R\mathbf{V}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{V}. \tag{3.89b}$$

Eq. 3.88 and Eq. 3.89 can be derived by noting that

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$\begin{aligned} p(\mathbf{x}_t|\mathcal{X}_s) &= \int p(\mathbf{x}_t|\mathbf{z}_s)p(\mathbf{z}_s|\mathcal{X}_s)\mathrm{d}\mathbf{z}_s \\ &= \int \mathcal{N}(\mathbf{x}_t|\mathbf{m}+\mathbf{V}\mathbf{z}_s, \boldsymbol{\Sigma})\mathcal{N}(\mathbf{z}_s|\langle \mathbf{z}_s|\mathcal{X}_s\rangle, \mathbf{L}_s^{-1})\mathrm{d}\mathbf{z}_s \\ &= \mathcal{N}\left(\mathbf{x}_t\middle|\mathbf{m}+\mathbf{V}\langle \mathbf{z}_s|\mathcal{X}_s\rangle, \mathbf{V}\mathbf{L}_s^{-1}\mathbf{V}^{\mathsf{T}}+\boldsymbol{\Sigma}\right) \end{aligned} \tag{3.90}$$

where the second Gaussian can be obtained by using Eq. 3.76. The marginal likelihood in the denominator of Eq. 3.87 can be obtained by using Eq. 3.56, i.e., $p(\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_t|\mathbf{m}, \mathbf{V}\mathbf{V}^{\mathsf{T}}+\boldsymbol{\Sigma})$.

Substituting Eq. 3.88 and the marginal likelihood $p(\mathbf{x}_t)$ into Eq. 3.87, we have

$$S_{\mathrm{LR}}(\mathcal{X}_s, \mathbf{x}_t) = \frac{\mathcal{N}\left(\mathbf{x}_t\middle|\mathbf{m}+\mathbf{V}\langle \mathbf{z}_s|\mathcal{X}_s\rangle, \mathbf{V}\mathbf{L}_s^{-1}\mathbf{V}^{\mathsf{T}}+\boldsymbol{\Sigma}\right)}{\mathcal{N}(\mathbf{x}_t|\mathbf{m}, \mathbf{V}\mathbf{V}^{\mathsf{T}}+\boldsymbol{\Sigma})},$$

which can be viewed as the likelihood ratio between the speaker-adapted PLDA model and the universal PLDA model.

The above derivation assumes that the i-vectors from the target speaker are conditional independent, because the posterior mean and covariance matrix in Eq. 3.89 are derived from Eq. 3.76. If independency is not assumed, we may use the method described in [48]. But, the method will be computationally intensive when the number of enrollment utterances per speaker is large.

3.4.4 Enhancement of PLDA

A number of enhancements have been proposed to improve the performance of PLDA. For example, Cumani et al. [50] propose to map the acoustic vectors of an utterance to a posterior distribution of i-vectors rather than a single i-vector. The method is particularly suitable for short utterances. Burget et al. [51] and Vasilakakis et al. [52] consider pairs of i-vectors and formulate an SVM-style training procedure to train a PLDA model to discriminate between same-speaker pairs and different-speaker pairs. Other approaches include constraining the PLDA parameters [53], applying nonparametric discriminant analysis before PLDA [54–56], and the use of between-speaker scatter as well as within-speaker scatter in PLDA training [57].

3.4.5 Alternative to PLDA

As PLDA models are linear, attempts have been made to see if they can be replaced by nonlinear models. Research has shown that deep belief networks can be used for discriminating the i-vectors of target speakers from those of impostors [58]. In [59], a binary restricted Boltzmann machine (RBM) was used to model the joint density of target-speaker's i-vectors and test i-vectors. Because RBMs can model the distribution of arbitrary complexity, they are thought to be more powerful than the conventional PLDA models.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Instead of using PLDA for scoring i-vectors, [58, 60] use a deep belief network (DBN) with two output nodes to classify i-vectors into speaker-class and impostor-class. The key idea is to use unsupervised training to train a universal DBN for all speakers. Then, a layer comprising two output nodes is stacked on top of the universal DBN, followed by backpropagation fine-tuning to minimize the cross-entropy error.

3.5 Heavy-Tailed PLDA

In the simplified Gaussian PLDA discussed in Section 3.5, the prior of the latent factor $\mathbf{z}$ follows a standard Gaussian distribution. This means that the parameters of the prior are assumed deterministic. The idea of PLDA in [17] can be extended to a fully Bayesian model in which the parameters of the prior distribution are assumed random, with the distribution of the hyper-parameters follow Gamma distributions [61]. In that case, $\mathbf{z}$ follows a Student's t distribution rather than a Gaussian distribution and the resulting model is called heavy-tailed PLDA model.

3.5.1 Generative Model

Figure 3.22 shows the graphical model of the heavy-tailed PLDA. The generative model is given by

$$\mathbf{x}_{ij} = \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_{ij} \tag{3.91}$$

$$\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, u_i^{-1}\mathbf{I}) \tag{3.92}$$

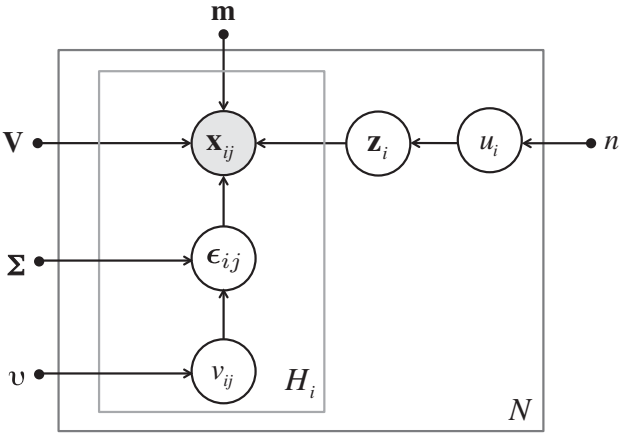


Figure 3.22 Graphical representation of a heavy-tailed PLDA model in which the channel variability is modeled by the full covariance matrix $\boldsymbol{\Sigma}$. The prior of $\mathbf{z}_i$ follows a Gaussian distribution $\mathcal{N}(\mathbf{0}, u_i^{-1}\mathbf{I})$, where u_i follows a Gamma distribution with n degree of freedom. The prior of ϵ_{ij} follows a Gaussian distribution $\mathcal{N}(\mathbf{0}, v_{ij}^{-1}\boldsymbol{\Sigma})$, where v_{ij} follows a Gamma distribution with ν degree of freedom.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$\boldsymbol{\epsilon}_{ij} \sim \mathcal{N}(\mathbf{0}, v_{ij}^{-1}\boldsymbol{\Sigma}) \tag{3.93}$$

$$u_i \sim \mathcal{G}(n/2, n/2) \tag{3.94}$$

$$v_{ij} \sim \mathcal{G}(\nu/2, \nu/2), \tag{3.95}$$

where $\mathcal{G}(\alpha/2, \alpha/2)$ is a Gamma distribution with α degree of freedom and identical scale and rate parameters. Note that because the latent variable u follows $\mathcal{G}(n/2, n/2)$, $\mathbf{z}_i$ follows a Student's t distribution with n degree of freedom [1, Eq. 2.161]. Similar argument applies to $\boldsymbol{\epsilon}$, i.e., $\boldsymbol{\epsilon} \sim \mathcal{T}(\mathbf{0}, \boldsymbol{\Sigma}, \nu)$ [62].

3.5.2 Posteriors of Latent Variables

Figure 3.22 suggests that given a set of training vectors $\mathcal{X}_i = \{\mathbf{x}_{ij}; j = 1, \ldots, H_i\}$ from the ith speakers, the posteriors of the latent variables $\mathbf{z}_i$, u_i, $\boldsymbol{\epsilon}_{ij}$, and v_{ij} are dependent. This means that $p(\mathbf{z}_i, u_i, \boldsymbol{\epsilon}_{ij}, v_{ij}|\mathcal{X}_i)$ cannot be factorized; as a result, we resort to using variational Bayes [1]:

$$\log p(\mathbf{z}_i, u_i, \boldsymbol{\epsilon}_{ij}, v_{ij}|\mathcal{X}_i) \approx \log q(\mathbf{z}_i) + \log q(u_i) + \log q(\boldsymbol{\epsilon}_{ij}) + \log q(v_{ij}). \tag{3.96}$$

We denote the combined latent variables as $\mathbf{h}_i = \{\mathbf{z}_i, u_i, \mathbf{v}_i\}$,[6] where $\mathbf{v}_i = [v_{i1} \ \ldots \ v_{iH_i}]^\mathsf{T}$. To find $q()$'s, we apply variational Bayes:

$$\log q(\mathbf{z}_i) = \mathbb{E}_{q(\mathbf{h}_i \setminus \mathbf{z}_i)} \log p(\mathcal{X}_i, \mathbf{h}_i) + \text{const} \tag{3.97a}$$

$$\log q(u_i) = \mathbb{E}_{q(\mathbf{h}_i \setminus u_i)} \log p(\mathcal{X}_i, \mathbf{h}_i) + \text{const} \tag{3.97b}$$

$$\log q(v_{ij}) = \mathbb{E}_{q(\mathbf{h}_i \setminus v_{ij})} \log p(\mathcal{X}_i, \mathbf{h}_i) + \text{const}, \tag{3.97c}$$

where the notation $\mathbb{E}_{q(\mathbf{h}_i \setminus \mathbf{z}_i)}$ means taking expectation with respect to all latent variables except $\mathbf{z}_i$, using $q(u_i)q(\mathbf{v}_i)$ as the density. Similar meaning applies to other latent variables.

Using Figure 3.22, we may express $\log p(\mathcal{X}_i, \mathbf{h}_i)$ as follows:

$$\begin{aligned}
\log p(\mathcal{X}_i, \mathbf{h}_i) &= \log p(\mathcal{X}_i|\mathbf{z}_i, \boldsymbol{\epsilon}_{ij}) p(\mathbf{z}_i|u_i) p(u_i) \prod_{j=1}^{H_i} p(\boldsymbol{\epsilon}_{ij}|v_{ij}) p(v_{ij}) \\
&= -\frac{1}{2}\left[\sum_{j=1}^{H_i} v_{ij}(\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i)^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i)\right] \\
&\quad + \frac{M}{2}\log u_i - \frac{u_i}{2}\mathbf{z}_i^\mathsf{T}\mathbf{z}_i + \log \mathcal{G}(u_i|n/2, n/2) \\
&\quad + \frac{D}{2}\sum_{j=1}^{H_i} \log v_{ij} - \frac{1}{2}\sum_{j=1}^{H_i} v_{ij}\boldsymbol{\epsilon}_{ij}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij} + \sum_{j=1}^{H_i} \log \mathcal{G}(v_{ij}|\nu/2, \nu/2) \\
&\quad + \text{const},
\end{aligned} \tag{3.98}$$

[6] Note that $\boldsymbol{\epsilon}_{ij}$ is not part of the latent variables because it can be determined from $\mathbf{z}_i$, i.e., $\boldsymbol{\epsilon}_{ij} = \mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where M is the dimension of $\mathbf{z}_i$ and

$$\mathcal{G}(u_i|n/2, n/2) = \frac{1}{\Gamma(\frac{n}{2})}\left(\frac{n}{2}\right)^{\frac{n}{2}} u_i^{\frac{n}{2}-1} e^{-\frac{n}{2}u_i} \tag{3.99}$$

is a Gamma distribution. The constant term in Eq. 3.98 absorbs the terms that do not contain the latent variables.

Substituting Eq. 3.98 to Eq. 3.97(a), we have

$$\begin{aligned}\log q(\mathbf{z}_i) = &-\frac{1}{2}\left[\sum_{j=1}^{H_i}(\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i)^\mathsf{T}\langle v_{ij}\rangle\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i)\right] \\ &- \frac{1}{2}\langle u_i\rangle\mathbf{z}^\mathsf{T}\mathbf{z} + \text{const},\end{aligned} \tag{3.100}$$

where "const" absorbs all terms independent of $\mathbf{z}$ and $\langle x\rangle$ is the shorthand form for the expectation of x using the variational posterior $q(x)$ as the density function, i.e., $\langle x\rangle \equiv \mathbb{E}_{q(x)}\{x\}$. Note that we have used the factorization property of variational Bayes in Eq. 3.96 to derive Eq. 3.100. Using the same technique as in Eq. 3.76 and Eq. 3.77, we obtain the posterior mean and posterior covariance of $\mathbf{z}_i$ as follows:

$$\begin{aligned}\langle\mathbf{z}_i\rangle &= \left(\langle u_i\rangle\mathbf{I} + \sum_{j=1}^{H_i}\langle v_{ij}\rangle\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\right)^{-1}\left(\sum_{j=1}^{H_i}\langle v_{ij}\rangle\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_{ij} - \mathbf{m})\right) \\ \text{Cov}(\mathbf{z}_i, \mathbf{z}_i) &= \left(\langle u_i\rangle\mathbf{I} + \sum_{j=1}^{H_i}\langle v_{ij}\rangle\mathbf{V}^\mathsf{T}\boldsymbol{\Sigma}^{-1}\mathbf{V}\right)^{-1},\end{aligned} \tag{3.101}$$

where $\langle u_i\rangle$ and $\langle v_{ij}\rangle$ are the posterior means of u_i and v_{ij}, respectively. Note that $q(\mathbf{z}_i)$ is a Gaussian distribution with mean $\langle\mathbf{z}_i\rangle$ and convariance matrix $\text{Cov}(\mathbf{z}_i, \mathbf{z}_i)$ even though u_i and v_{ij} follow Gamma distributions.

Substituting Eq. 3.98 to Eq. 3.97(b), we have

$$\begin{aligned}\log q(u_i) &= \frac{M}{2}\log u_i - \frac{u_i}{2}\langle\mathbf{z}_i^\mathsf{T}\mathbf{z}_i\rangle + \left(\frac{n}{2} - 1\right)\log u_i - \frac{n}{2}u_i + \text{const} \\ &= \left(\frac{n+M}{2} - 1\right)\log u_i - \left(\frac{n + \langle\mathbf{z}_i^\mathsf{T}\mathbf{z}_i\rangle}{2}\right)u_i + \text{const},\end{aligned} \tag{3.102}$$

where the const term absorbs all terms independent of u_i. Eq. 3.102 is the logarithm of a Gamma distribution with shape α and rate β given by

$$\alpha = \frac{n+M}{2} \quad \text{and} \quad \beta = \frac{n + \langle\mathbf{z}_i^\mathsf{T}\mathbf{z}_i\rangle}{2}. \tag{3.103}$$

As $q(u_i)$ is an approximation of the posterior density of u_i, the posterior mean $\langle u_i\rangle$ in Eq. 3.101 can be computed as follows:

$$\langle u_i\rangle = \mathbb{E}_{q(u_i)}\{u_i\} = \alpha\beta^{-1} = \frac{n+M}{n + \langle\mathbf{z}_i^\mathsf{T}\mathbf{z}_i\rangle}. \tag{3.104}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Substituting Eq. 3.98 to Eq. 3.97(c), we have

$$\begin{aligned}\log q(v_{ij}) &= \frac{D}{2}\log v_{ij} - \frac{1}{2}v_{ij}\langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i} + \left(\frac{\nu}{2}-1\right)\log v_{ij} - \frac{\nu}{2}v_{ij} + \text{const} \\ &= \left(\frac{\nu+D}{2}-1\right)\log v_{ij} - \left(\frac{\nu + \langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}}{2}\right)v_{ij} + \text{const},\end{aligned} \tag{3.105}$$

where D is the dimensionality of $\mathbf{x}_{ij}$ and $\langle\cdot\rangle_{\mathbf{z}_i}$ means taking expectation with respect to the approximated posterior $q(\mathbf{z}_i)$. Eq. 3.105 is the logarithm of a Gamma distribution with shape α and rate β given by

$$\alpha = \frac{\nu+D}{2} \quad \text{and} \quad \beta = \frac{\nu + \langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}}{2}.$$

The posterior mean of v_{ij} in Eq. 3.101 is given by

$$\langle v_{ij}\rangle = \mathbb{E}_{q(v_{ij})}\{v_{ij}\} = \frac{\nu+D}{\nu + \langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}}. \tag{3.106}$$

The remaining posterior expectation is $\langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}$. To compute this posterior expectation, we leverage the symmetry of $\boldsymbol{\Sigma}^{-1}$ [63, Eq. 318]:

$$\langle \boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i} = \text{Tr}\{\boldsymbol{\Sigma}^{-1}\text{Cov}(\boldsymbol{\epsilon}_{ij},\boldsymbol{\epsilon}_{ij})\} + \langle \boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\langle \boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i}.$$

Because $\boldsymbol{\epsilon}_{ij} = \mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\mathbf{z}_i$, we have

$$\begin{aligned}\langle \boldsymbol{\epsilon}_{ij}\rangle_{\mathbf{z}_i} &= \mathbf{x}_{ij} - \mathbf{m} - \mathbf{V}\langle \mathbf{z}_i\rangle \\ \text{Cov}(\boldsymbol{\epsilon}_{ij},\boldsymbol{\epsilon}_{ij}) &= \mathbf{V}\text{Cov}(\mathbf{z}_i,\mathbf{z}_i)\mathbf{V}^{\mathsf{T}},\end{aligned}$$

where $\text{Cov}(\mathbf{z}_i,\mathbf{z}_i)$ and $\langle \mathbf{z}_i\rangle$ can be obtained from Eq. 3.101.

3.5.3 Model Parameter Estimation

Denote the model parameter of a heavy-tailed PLDA model as $\boldsymbol{\omega} = \{\mathbf{m},\mathbf{V},\boldsymbol{\Sigma},n,\nu\}$. Also denote $\mathcal{H} = \{\mathbf{z}_i,u_i,v_{ij};i=1,\ldots,N;j=1,\ldots,H_i\}$ as the set of latent variables associated with the training vectors $\mathcal{X} = \{\mathcal{X}_1,\ldots,\mathcal{X}_N\}$, where $\mathcal{X}_i = \{\mathbf{x}_{i1},\ldots,\mathbf{x}_{iH_i}\}$.

Similar to PLDA, the model parameter can be estimated by maximizing the following auxiliary function:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$Q(\boldsymbol{\omega}'|\boldsymbol{\omega}) = \mathbb{E}_{\mathcal{H}}\{\ln p(\mathcal{X}, \mathcal{H}|\boldsymbol{\omega}')\} \tag{3.107a}$$

$$= \mathbb{E}_{\mathcal{H}}\left\{\sum_{ij} \log p(\mathbf{x}_{ij}|\mathbf{z}_i, u_i, v_{ij}, \boldsymbol{\omega}')p(\mathbf{z}_i|u_i, \boldsymbol{\omega}')p(u_i|\boldsymbol{\omega}')p(v_{ij}|\boldsymbol{\omega}')\right\} \tag{3.107b}$$

$$= \mathbb{E}_{\mathcal{H}}\left\{\sum_{ij} \log \mathcal{N}(\mathbf{x}_{ij}|\mathbf{m}' + \mathbf{V}'\mathbf{z}_i, v_{ij}^{-1}\boldsymbol{\Sigma}')\right\} + \mathbb{E}_{\mathcal{H}}\left\{\sum_{ij} \log \mathcal{N}(\mathbf{z}_i|\mathbf{0}, u_i^{-1}\mathbf{I})\right\}$$
$$+ \mathbb{E}_{\mathcal{H}}\left\{\sum_{ij} \log \mathcal{G}(u_i|n/2, n/2)\mathcal{G}(v_{ij}|\nu/2, \nu/2)\right\}. \tag{3.107c}$$

When estimating $\mathbf{m}$, $\mathbf{V}$, and $\boldsymbol{\Sigma}$, the second and third terms in Eq. 3.107 can be dropped because they do not contain these model parameters. Therefore, we may follow the derivation in Section 3.4.2 to obtain the parameter update formulae (M-step) as follows:

$$\mathbf{V}' = \left[\sum_{i=1}^{N}\sum_{j=1}^{H_i}\langle v_{ij}\rangle(\mathbf{x}_{ij} - \mathbf{m}')\langle \mathbf{z}_i\rangle^{\mathsf{T}}\right]\left[\sum_{i=1}^{N}\sum_{j=1}^{H_i}\langle v_{ij}\rangle\langle \mathbf{z}_i\mathbf{z}_i^{\mathsf{T}}\rangle\right]^{-1} \tag{3.108a}$$

$$\mathbf{m}' = \frac{\sum_{ij}\langle v_{ij}\rangle\mathbf{x}_{ij}}{\sum_{ij}\langle v_{ij}\rangle} \tag{3.108b}$$

$$\boldsymbol{\Sigma}' = \frac{1}{\sum_{i=1}^{N} H_i}\left\{\sum_{i=1}^{N}\sum_{j=1}^{H_i}\langle v_{ij}\rangle\left[(\mathbf{x}_{ij} - \mathbf{m}')(\mathbf{x}_{ij} - \mathbf{m}')^{\mathsf{T}} - \mathbf{V}'\langle \mathbf{z}_i\rangle(\mathbf{x}_{ij} - \mathbf{m}')^{\mathsf{T}}\right]\right\} \tag{3.108c}$$

The degrees of freedom n can be estimating by minimizing the sum of the KL-divergence:

$$d(n) = \sum_{i=1}^{N} \mathcal{D}_{\mathrm{KL}}(\mathcal{G}(\alpha_i, \beta_i)\|\mathcal{G}(n/2, n/2))$$
$$= \sum_{i=1}^{N}\left[\log\frac{\Gamma(n/2)}{\Gamma(\alpha_i)} + \alpha_i\log\beta_i + \left(\alpha_i - \frac{n}{2}\right)(\psi(\alpha_i) - \log\beta_i) + \frac{\alpha_i}{\beta_i}\left(\frac{n}{2} - \beta_i\right)\right]$$
$$- \frac{nN}{2}\log\frac{n}{2},$$

where α_i and β_i are Gamma distribution parameters obtained from Eq. 3.103, $\Gamma(\alpha_i)$ is the gamma function, and

$$\psi(x) = \frac{\partial}{\partial x}\log\Gamma(x) = \frac{\Gamma'(x)}{\Gamma(x)}$$

is the digamma function. We may use the following properties to simplify $d(n)$:

$$\langle u_i\rangle = \alpha_i\beta_i^{-1} \quad \text{and} \quad \langle \log u_i\rangle = \psi(\alpha_i) - \log\beta_i. \tag{3.109}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Also, consolidating terms in $d(n)$ that are independent of n as a constant term, we have

$$d(n) = N \log \Gamma(n/2) + \sum_{i=1}^{N} \Big[-\frac{n}{2} \langle \log u_i \rangle + \langle u_i \rangle \frac{n}{2} \Big] - \frac{nN}{2} \log \frac{n}{2} + \text{const.}$$

Setting the derivative of $d(n)$ to 0, we have

$$N \frac{\Gamma'(n/2)}{\Gamma(n/2)} + \sum_{i=1}^{N} \big[-\langle \log u_i \rangle + \langle u_i \rangle \big] - N \log \frac{n}{2} - N = 0,$$

which is equivalent to

$$N \psi \left(\frac{n}{2} \right) + \sum_{i=1}^{N} \big[-\langle \log u_i \rangle + \langle u_i \rangle \big] - N \log \frac{n}{2} - N = 0. \tag{3.110}$$

The optimal value of n can be found by applying line search on Eq. 3.110. The value of ν can be found similarly. Specifically, line search is applied to

$$N' \psi \left(\frac{\nu}{2} \right) + \sum_{i=1}^{N} \sum_{j=1}^{H_i} \big[-\langle \log v_{ij} \rangle + \langle v_{ij} \rangle \big] - N' \log \frac{\nu}{2} - N' = 0, \tag{3.111}$$

where $N' = \sum_{i=1}^{N} H_i$.

3.5.4 Scoring in Heavy-Tailed PLDA

Similar to the Gaussian PLDA in Section 3.4.3, scoring in heavy-tailed PLDA also involves the computation of a likelihood ratio score

$$S_{\text{LR}}(\mathbf{x}_s, \mathbf{x}_t) = \frac{p(\mathbf{x}_s, \mathbf{x}_t | H_1)}{p(\mathbf{x}_s | H_0) p(\mathbf{x}_t | H_0)}, \tag{3.112}$$

where H_1 and H_0 are the hypotheses that the input vectors $\mathbf{x}_s$ and $\mathbf{x}_t$ are from the same speaker and from different speakers, respectively. But unlike the Gaussian PLDA in Eq. 3.83, in heavy-tailed PLDA, there is no close form solution to the marginal likelihoods in Eq. 3.112. Therefore, we need to define a tractable solution and use it as a proxy to compute the marginal likelihoods. Let's define

$$\begin{aligned} \mathcal{L} &= \mathbb{E}_{\mathbf{h} \sim q(\mathbf{h})} \left\{ \log \frac{p(\mathbf{x}, \mathbf{h})}{q(\mathbf{h})} \right\} \\ &= \mathbb{E}_{\mathbf{h} \sim q(\mathbf{h})} \{ \log p(\mathbf{x} | \mathbf{h}) \} - \mathbb{E}_{\mathbf{h} \sim q(\mathbf{h})} \left\{ \log \frac{q(\mathbf{h})}{p(\mathbf{h})} \right\} \\ &= \mathcal{L}_1 + \mathcal{L}_2, \end{aligned} \tag{3.113}$$

where $\mathbf{x}$ can be $\mathbf{x}_s$ (target), $\mathbf{x}_t$ (test), or $[\mathbf{x}_s^{\mathsf{T}} \ \mathbf{x}_t^{\mathsf{T}}]^{\mathsf{T}}$; $\mathbf{h}$ comprises all latent variables, i.e., $\mathbf{h} = \{\mathbf{z}, u, v\}$; and $q(\mathbf{h})$ is the variational posterior of $\mathbf{h}$. Note that $\mathcal{L} \leq \log p(\mathbf{x})$ with equality holds when $q(\mathbf{h})$ is equal to the true posterior $p(\mathbf{h}|\mathbf{x})$. Therefore, we may use $\mathcal{L}$ as an approximation to the marginal likelihood $p(\mathbf{x})$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$\mathcal{L}_1$ can be obtained from Eq. 3.98 by dropping all the prior over the latent variables:

$$\begin{aligned}\mathcal{L}_1 &= \mathbb{E}_{\mathbf{h}\sim q(\mathbf{h})}\{\log p(\mathbf{x}|\mathbf{z},u,v,\boldsymbol{\omega})\}\\ &= \mathbb{E}_{\mathbf{h}\sim q(\mathbf{h})}\{\log \mathcal{N}(\mathbf{x}|\mathbf{m}+\mathbf{V}\mathbf{z},v^{-1}\boldsymbol{\Sigma})\}\\ &= \frac{D}{2}\langle\log v\rangle - \frac{D}{2}\log 2\pi - \frac{1}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}\langle v\rangle\langle\boldsymbol{\epsilon}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}\rangle,\end{aligned} \tag{3.114}$$

where $\boldsymbol{\epsilon} = \mathbf{x} - \mathbf{m} - \mathbf{V}\mathbf{z}$ and $\langle\cdot\rangle$ stands for expectation with respect to the latent variables using $q(\mathbf{h})$ as the density. The posterior expectation $\langle v\rangle$ can be obtained from Eq. 3.106 and $\langle\log v\rangle$ can be computed by replacing u by v in Eq. 3.109 as follows:

$$\langle\log v\rangle = \psi\left(\frac{v+D}{2}\right) - \log\left(\frac{v+\boldsymbol{\epsilon}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}}{2}\right).$$

As $\mathcal{L}_2$ is the negative of the Kullback–Leibler divergence from $q(\mathbf{h})$ to $p(\mathbf{h})$ and the variational posteriors are factorizable, we have

$$\begin{aligned}\mathcal{L}_2 &= -\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z},u,v)\|\,p(\mathbf{z},u,v))\\ &= -\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z},u)q(v)||\,p(\mathbf{z},u)p(v))\\ &= -\left[\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z},u)\|\,p(\mathbf{z},u)) + \mathcal{D}_{\mathrm{KL}}(q(v)\|\,p(v))\right]\\ &= -\left[\mathcal{D}_{\mathrm{KL}}(q(u)\|\,p(u)) + \mathbb{E}_{u\sim p(u)}\{\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z}|u)\|\,p(\mathbf{z}|u))\} + \mathcal{D}_{\mathrm{KL}}(q(v)\|\,p(v))\right]\\ &= -\left[\mathcal{D}_{\mathrm{KL}}(q(u)\|\,p(u)) + \mathbb{E}_{u\sim p(u)}\{\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z})\|\,p(\mathbf{z}|u))\} + \mathcal{D}_{\mathrm{KL}}(q(v)\|\,p(v))\right]\end{aligned} \tag{3.115}$$

where we have used the chain rule of KL-divergence[7] and the notation $q(\mathbf{z}) \equiv q(\mathbf{z}|u)$. Note that we have also used the fact $p(v|\mathbf{z},u) = p(v)$ and $\mathbb{E}_{p(\mathbf{z},u)}p(v) = p(v)$ in Line 3 of Eq. 3.115.

The distributions in Eq. 3.115 can be obtained from the posteriors and priors of the latent variables as follows:

$$q(u) = \mathcal{G}\left(\frac{n+M}{2}, \frac{n+\langle\mathbf{z}_i^{\mathsf{T}}\mathbf{z}_i\rangle}{2}\right) \tag{3.116a}$$

$$p(u) = \mathcal{G}\left(\frac{n}{2}, \frac{n}{2}\right) \tag{3.116b}$$

$$q(v) = \mathcal{G}\left(\frac{v+D}{2}, \frac{v+\langle\boldsymbol{\epsilon}_{ij}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\epsilon}_{ij}\rangle}{2}\right) \tag{3.116c}$$

$$p(v) = \mathcal{G}\left(\frac{v}{2}, \frac{v}{2}\right) \tag{3.116d}$$

$$q(\mathbf{z}) = \mathcal{N}\left(\langle\mathbf{z}\rangle, \mathrm{Cov}(\mathbf{z},\mathbf{z})\right) \tag{3.116e}$$

$$p(\mathbf{z}|u) = \mathcal{N}\left(\mathbf{0}, u^{-1}\mathbf{I}\right). \tag{3.116f}$$

Note that the KL-divergence between two Gamma distributions and between two Gaussian distributions are [64, 65]:

[7] $\mathcal{D}_{\mathrm{KL}}(q(x,y)\|\,p(x,y)) = \mathcal{D}_{\mathrm{KL}}(q(x)\|\,p(x)) + \mathbb{E}_{x\sim p(x)}\{\mathcal{D}_{\mathrm{KL}}(q(y|x)\|\,p(y|x))\}$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$\mathcal{D}_{\mathrm{KL}}\left(\mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q) \| \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\right)$$
$$= \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_p|}{|\boldsymbol{\Sigma}_q|} + \mathrm{Tr}\{\boldsymbol{\Sigma}_p^{-1}\boldsymbol{\Sigma}_q\} + (\boldsymbol{\mu}_q - \boldsymbol{\mu}_p)^{\mathsf{T}}\boldsymbol{\Sigma}_p^{-1}(\boldsymbol{\mu}_q - \boldsymbol{\mu}_p) - D\right] \quad (3.117a)$$

$$\mathcal{D}_{\mathrm{KL}}\left(\mathcal{G}(\alpha_q, \beta_q) \| \mathcal{G}(\alpha_p, \beta_p)\right)$$
$$= \alpha_p \log\frac{\beta_q}{\beta_p} - \log\frac{\Gamma(\alpha_q)}{\Gamma(\alpha_p)} + (\alpha_q - \alpha_p)\psi(\alpha_q) - (\beta_q - \beta_p)\frac{\alpha_q}{\beta_q}. \quad (3.117b)$$

Using these KL-divergence, the second term in Eq. 3.115 is given by

$$\mathbb{E}_{u\sim p(u)}\{\mathcal{D}_{\mathrm{KL}}(q(\mathbf{z}) \| p(\mathbf{z}|u))\}$$
$$= \frac{1}{2}\left[-D\langle \log u\rangle - \log|\mathrm{Cov}(\mathbf{z},\mathbf{z})| + \mathrm{Tr}\{\langle u\rangle \mathrm{Cov}(\mathbf{z},\mathbf{z})\} + \langle u\rangle\langle \mathbf{z}\rangle^{\mathsf{T}}\langle \mathbf{z}\rangle - D\right].$$

The first and the third terms in Eq. 3.115 can be computed similarly using Eq. 3.117(b) and Eq. 3.116.

3.5.5 Heavy-Tailed PLDA versus Gaussian PLDA

While the training and scoring of heavy-tailed PLDA are more complicated and computationally expensive than Gaussian PLDA, in speaker recognition, the former is proposed earlier than the latter. In fact, Kenny has shown in [61] that heavy-tailed PLDA performs better than Gaussian PLDA if no preprocessing is applied to the i-vectors. However, in 2011, Garcia-Romero and Espy-Wilson [66] discovered that if i-vectors are Gaussianized by length-normalization, Gaussian PLDA can perform as good as heavy-tailed PLDA. Since then, Gaussian PLDA became more popular than heavy-tailed PLDA.

In 2018, Silnova et al. [67] proposed a method that leverage the more exact modeling of heavy-tailed PLDA but with the computation advantage of Gaussian PLDA. With heavy-tailed PLDA, length-normalization, which may remove some speaker information, is no longer needed. The main idea is based on the observation that the conditional distribution of i-vectors given the speaker factor $\boldsymbol{z}$ is proportional to another t-distribution with degrees of freedom $n' = n + D - M$. If n' is large, the distribution is practically a Gaussian. In practice, D is about 500 and M is 150. As a result, n' is large enough to make the conditional distribution of $\mathbf{x}$ Gaussian. This approximation allows the computation of the joint variational posterior $q(\mathbf{v}_i)$ in Eq. 3.105 independent of the speaker factor $\boldsymbol{z}_i$, which results in a very fast E-step.

3.6 I-Vectors

I-vectors are based on factor analysis in which the acoustic features (typically MFCC and log-energy plus their first and second derivatives) are generated by a Gaussian mixture model (GMM). Since its first appeared [16], it has become the de facto method for many text-independent speaker verification systems. Although i-vectors were orig-

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

inally proposed for speaker verification, they have been found useful in many other applications, including speaker diarization [68], voice activity detection [69], language recognition [70], and ECG classification [71].

3.6.1 Generative Model

Given the ith utterance, we extract the F-dimensional acoustic vectors $\mathcal{O}_i = \{\mathbf{o}_{i1}, \ldots, \mathbf{o}_{iT_i}\}$ from the utterance, where T_i is the number of frames in the utterance. We assume that the acoustic vectors are generated by an utterance-dependent GMM with parameters $\Lambda_i = \{\pi_c, \boldsymbol{\mu}_{ic}, \boldsymbol{\Sigma}_c\}_{c=1}^{C}$, i.e.,

$$p(\mathbf{o}_{it}) = \sum_{c=1}^{C} \pi_c^{(b)} \mathcal{N}(\mathbf{o}_{it} | \boldsymbol{\mu}_{ic}, \boldsymbol{\Sigma}_c^{(b)}), \quad t = 1, \ldots, T_i, \tag{3.118}$$

where C is the number of mixtures in the GMM. For simplicity, we assume that $\lambda_c^{(b)}$ and $\boldsymbol{\Sigma}_c^{(b)}$ are tied across all utterances and are equal to the mixture weights and covariance matrices of the universal background model (UBM), respectively.

In the i-vector framework [16], the supervector representing the ith utterance is assumed to be generated by the following factor analysis model [1]:[8]

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}^{(b)} + \mathbf{T}\mathbf{w}_i \tag{3.119}$$

where $\boldsymbol{\mu}^{(b)}$ is obtained by stacking the mean vectors of the UBM, $\mathbf{T}$ is a $CF \times D$ low-rank total variability matrix modeling the speaker and channel variability, and $\mathbf{w}_i \in \mathbb{R}^D$ comprises the latent (total) factors. Eq. 3.119 suggests that the generated supervectors $\boldsymbol{\mu}_i$'s have mean $\boldsymbol{\mu}^{(b)}$ and convariance matrix $\mathbf{T}\mathbf{T}^{\mathsf{T}}$. Eq. 3.119 can also be written in a component-wise form:

$$\boldsymbol{\mu}_{ic} = \boldsymbol{\mu}_c^{(b)} + \mathbf{T}_c \mathbf{w}_i, \quad c = 1, \ldots, C \tag{3.120}$$

where $\boldsymbol{\mu}_{ic} \in \mathbb{R}^F$ is the cth sub-vector of $\boldsymbol{\mu}_i$ (similarly for $\boldsymbol{\mu}_c^{(b)}$) and $\mathbf{T}_c$ is an $F \times D$ sub-matrix of $\mathbf{T}$. Figure 3.23 shows the graphical representation of this generative model.

In [45], it was assumed that for every speaker s, the acoustic vectors (frames) aligning to the cth mixture component have mean $\boldsymbol{\mu}_{sc}$ and covariance $\boldsymbol{\Sigma}_c^{(b)}$. In the case of i-vectors, every utterance is assumed to be spoken by a different speaker. As a result, the frames of utterance i aligning to mixture c have mean $\boldsymbol{\mu}_{ic}$ and covariance matrix $\boldsymbol{\Sigma}_c^{(b)}$. This matrix measures the deviation of the acoustic vectors – which are aligned to the cth mixture – from the utterance-dependent mean $\boldsymbol{\mu}_{ic}$. A super-covariance matrix, denoted as $\boldsymbol{\Sigma}^{(b)}$, can be formed by packing the C covariance matrices, i.e., $\boldsymbol{\Sigma}^{(b)} = \text{diag}\{\boldsymbol{\Sigma}_1^{(b)}, \ldots, \boldsymbol{\Sigma}_C^{(b)}\}$. In the i-vector framework, model estimation amounts to finding the parameter set $\boldsymbol{\Lambda} = \{\boldsymbol{\mu}^{(b)}, \mathbf{T}, \boldsymbol{\Sigma}^{(b)}\}$. In practice, $\boldsymbol{\mu}^{(b)}$ and $\boldsymbol{\Sigma}^{(b)}$ are the mean vectors and covariance matrices of the UBM.

Note that unlike Eq. 3.55, Eq. 3.119 and Eq. 3.120 do not have the residue terms $\boldsymbol{\epsilon}_i$ and $\boldsymbol{\epsilon}_{ic}$, respectively. This is because the FA model in Eq. 3.119 aims to generate the supervector $\boldsymbol{\mu}_i$. If the supervector is observed, e.g., an GMM-supervector obtained from

[8] To simplify notations, from now on, we drop the over-arrow symbol ($\rightarrow$) in the supervectors.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

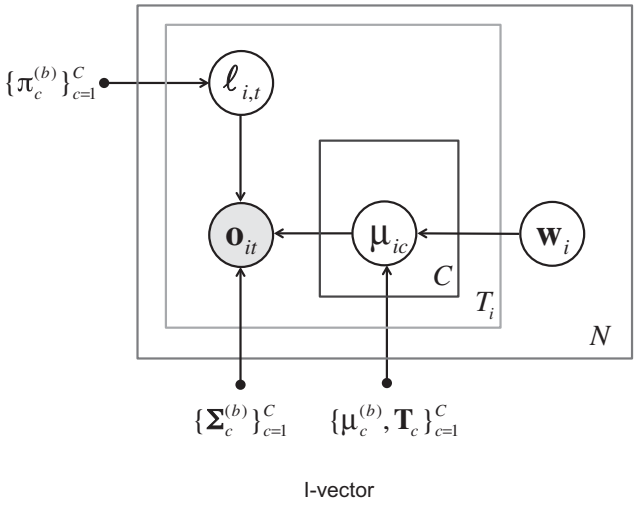


Figure 3.23 Graphical representation of the factor analysis model for i-vector extraction. $\boldsymbol{\ell}_{i,t} = [\ell_{i,t,1} \ \ldots \ \ell_{i,t,c} \ \ldots \ \ell_{i,t,C}]^\mathsf{T}$ are the mixture labels in one-hot encoding. The prior of $\mathbf{w}_i$ is assumed to be a standard Gaussian.

the MAP adaptation, an residue term will need to be added to Eq. 3.119 to represent the difference between the observed and the generated supervectors.

There is a close relationship between the residue covariance matrix $\mathbf{\Sigma}$ in Eq. 3.56 and the component-wise covariance matrices $\mathbf{\Sigma}_c^{(b)}$ described above. Given the acoustic vectors $\mathcal{O}_i$ of the ith utterance, we can compute the posterior mean (see Section 3.6.2), $\langle \mathbf{w}_i | \mathcal{O}_i \rangle$. Then, the density of $\boldsymbol{\mu}_{ic}$ in Eq. 3.120 is $\mathcal{N}(\boldsymbol{\mu}_{ic} | \boldsymbol{\mu}_c^{(b)} + \mathbf{T}_c \langle \mathbf{w}_i | \mathcal{O}_i \rangle, \mathbf{\Sigma}_c^{(b)})$. Here, the deviations of the acoustic vectors from the mean $(\boldsymbol{\mu}_c^{(b)} + \mathbf{T}_c \langle \mathbf{w}_i | \mathcal{O}_i \rangle)$ are specified by $\mathbf{\Sigma}_c^{(b)}$. In Eq. 3.55, these deviations are represented by $\boldsymbol{\epsilon}$ whose covariance matrix is $\mathbf{\Sigma}$.

3.6.2 Posterior Distributions of Total Factors

Given an utterance with acoustic vectors $\mathcal{O}_i$, the i-vector $\mathbf{x}_i$ representing the utterance is the posterior mean of $\mathbf{w}_i$, i.e., $\mathbf{x}_i = \langle \mathbf{w}_i | \mathcal{O}_i \rangle$. To determine $\mathbf{x}_i$, we may express the joint posterior density of $\mathbf{w}_i$ and indicator variables $\ell_{i,t,c}$'s, where $t = 1, \ldots, T_i$ and $c = 1, \ldots, C$. The indicator variable $\ell_{i,t,c}$ specifies which of the C Gaussians generates $\mathbf{o}_{it}$. More specifically, $\ell_{i,t,c} = 1$ if the cth mixture generates $\mathbf{o}_{it}$; otherwise, $\ell_{i,t,c} = 0$. The joint posterior density can be expressed in terms of $\mathbf{w}_i$ using the Bayes rule:

$$
\begin{aligned}
p(\mathbf{w}_i, \ell_{i,\cdot,\cdot} | \mathcal{O}_i) &\propto p(\mathcal{O}_i | \mathbf{w}_i, \ell_{i,\cdot,\cdot} = 1) p(\ell_{i,\cdot,\cdot}) p(\mathbf{w}_i) \\
&= \prod_{t=1}^{T} \prod_{c=1}^{C} \left[\pi_c p(\mathbf{o}_{it} | \ell_{i,t,c} = 1, \mathbf{w}_i) \right]^{\ell_{i,t,c}} p(\mathbf{w}_i) \quad \text{[1, Eq. 9.38]} \\
&= \underbrace{p(\mathbf{w}_i) \prod_{t=1}^{T} \prod_{c=1}^{C} \left[\mathcal{N}(\mathbf{o}_{it} | \boldsymbol{\mu}_c^{(b)} + \mathbf{T}_c \mathbf{w}_i, \mathbf{\Sigma}_c^{(b)}) \right]^{\ell_{i,t,c}}}_{\propto p(\mathbf{w}_i | \mathcal{O}_i)} \pi_c^{\ell_{i,t,c}}, \qquad (3.121)
\end{aligned}
$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\ell_{i,\cdot,\cdot}$ means all possible combinations of t and c for utterance i. In Eq. 3.121, we have used the fact that for each (i,t)-pair, only one of the $\ell_{i,t,c}$ for $c = 1, \ldots, C$ equals to 1, the rest are zeros. Extracting terms depending on $\mathbf{w}_i$ from Eq. 3.121, we obtain

$$\begin{aligned} & p(\mathbf{w}_i|\mathcal{O}_i) \\ & \propto \exp\left\{-\frac{1}{2}\sum_{c=1}^{C}\sum_{t\in\mathcal{H}_{ic}}(\mathbf{o}_{it}-\boldsymbol{\mu}_c^{(b)}-\mathbf{T}_c\mathbf{w}_i)^{\mathsf{T}}(\boldsymbol{\Sigma}_c^{(b)})^{-1}(\mathbf{o}_{it}-\boldsymbol{\mu}_c^{(b)}-\mathbf{T}_c\mathbf{w}_i)-\frac{1}{2}\mathbf{w}_i^{\mathsf{T}}\mathbf{w}_i\right\} \\ & = \exp\left\{\mathbf{w}_i^{\mathsf{T}}\sum_{c=1}^{C}\sum_{t\in\mathcal{H}_{ic}}\mathbf{T}_c^{\mathsf{T}}(\boldsymbol{\Sigma}_c^{(b)})^{-1}(\mathbf{o}_{it}-\boldsymbol{\mu}_c^{(b)})\right. \\ & \qquad \left.-\frac{1}{2}\mathbf{w}_i^{\mathsf{T}}\left(\mathbf{I}+\sum_{c=1}^{C}\sum_{t\in\mathcal{H}_{ic}}\mathbf{T}_c^{\mathsf{T}}(\boldsymbol{\Sigma}_c^{(b)})^{-1}\mathbf{T}_c\right)\mathbf{w}_i\right\}, \end{aligned} \tag{3.122}$$

where $\mathcal{H}_{ic}$ comprises the frame indexes for which $\mathbf{o}_{it}$ aligned to mixture c. Comparing Eq. 3.122 with Eq. 3.77, we obtain the following posterior expectations:

$$\begin{aligned} \langle\mathbf{w}_i|\mathcal{O}_i\rangle &= \mathbf{L}_i^{-1}\sum_{c=1}^{C}\sum_{t\in\mathcal{H}_{ic}}\mathbf{T}_c^{\mathsf{T}}\left(\boldsymbol{\Sigma}_c^{(b)}\right)^{-1}(\mathbf{o}_{it}-\boldsymbol{\mu}_c^{(b)}) \\ &= \mathbf{L}_i^{-1}\sum_{c=1}^{C}\mathbf{T}_c^{\mathsf{T}}\left(\boldsymbol{\Sigma}_c^{(b)}\right)^{-1}\sum_{t\in\mathcal{H}_{ic}}(\mathbf{o}_{it}-\boldsymbol{\mu}_c^{(b)}) \end{aligned} \tag{3.123}$$

$$\langle\mathbf{w}_i\mathbf{w}_i^{\mathsf{T}}|\mathcal{O}_i\rangle = \mathbf{L}_i^{-1} + \langle\mathbf{w}_i|\mathcal{O}_i\rangle\langle\mathbf{w}_i^{\mathsf{T}}|\mathcal{O}_i\rangle, \tag{3.124}$$

where

$$\mathbf{L}_i = \mathbf{I} + \sum_{c=1}^{C}\sum_{t\in\mathcal{H}_{ic}}\mathbf{T}_c^{\mathsf{T}}(\boldsymbol{\Sigma}_c^{(b)})^{-1}\mathbf{T}_c. \tag{3.125}$$

Note that when $C = 1$ (i.e., the UBM has one Gaussian only), Eq. 3.123 reduces to the posterior mean of the latent factor in the PLDA model in Eq. 3.78. This means that the factor analysis model in PLDA is a special case of the more general factor analysis model in i-vectors.

In Eq. 3.123 and Eq. 3.125, the sum over $\mathcal{H}_{ic}$ can be evaluated in two ways:

(1) *Hard Decisions*. For each t, the posterior probabilities of $\ell_{i,t,c}$, for $c = 1, \ldots, C$, are computed. Then, $\mathbf{o}_{it}$ is aligned to mixture c^* when

$$c^* = \underset{c}{\text{argmax}}\ \gamma(\ell_{i,t,c}),$$

where

$$\begin{aligned} \gamma(\ell_{i,t,c}) &\equiv \Pr(\text{Mixture} = c|\mathbf{o}_{it}) \\ &= \frac{\pi_c^{(b)}\mathcal{N}(\mathbf{o}_{it}|\boldsymbol{\mu}_c^{(b)},\boldsymbol{\Sigma}_c^{(b)})}{\sum_{j=1}^{C}\pi_j^{(b)}\mathcal{N}(\mathbf{o}_{it}|\boldsymbol{\mu}_j^{(b)},\boldsymbol{\Sigma}_j^{(b)})},\quad c = 1,\ldots,C \end{aligned} \tag{3.126}$$

are the posterior probabilities of mixture c given $\mathbf{o}_{it}$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

(2) *Soft Decisions*. Each frame is aligned to all of the mixtures with degree of alignment according to the posterior probabilities $\gamma(\ell_{i,t,c})$. Then, we have

$$\begin{aligned} \sum_{t\in\mathcal{H}_{ic}} 1 &\rightarrow \sum_{t=1}^{T_i} \gamma(\ell_{i,t,c}) \\ \sum_{t\in\mathcal{H}_{ic}} (\mathbf{o}_{it} - \boldsymbol{\mu}_c^{(b)}) &\rightarrow \sum_{t=1}^{T_i} \gamma(\ell_{i,t,c})(\mathbf{o}_{it} - \boldsymbol{\mu}_c^{(b)}). \end{aligned} \tag{3.127}$$

Eq. 3.127 results in the following Baum–Welch statistics:

$$N_{ic} \equiv \sum_{t=1}^{T_i} \gamma(\ell_{i,t,c}) \text{ and } \tilde{\mathbf{f}}_{ic} \equiv \sum_{t=1}^{T_i} \gamma(\ell_{i,t,c})(\mathbf{o}_{it} - \boldsymbol{\mu}_c^{(b)}). \tag{3.128}$$

3.6.3 I-Vector Extractor

Training of the total variability matrix involves estimating the total variability matrix $\mathbf{T}$ using the EM algorithm. The formulation can be derived based on the EM steps in Section 3.3. Specifically, using Eqs. 3.65, 3.123, and 3.124, the M-step for estimating $\mathbf{T}$ is

$$\mathbf{T}_c = \left[\sum\nolimits_i \tilde{\mathbf{f}}_{ic}\langle \mathbf{w}_i|\mathcal{O}_i\rangle^{\mathsf{T}}\right]\left[\sum\nolimits_i N_{ic}\langle \mathbf{w}_i\mathbf{w}_i^{\mathsf{T}}|\mathcal{O}_i\rangle\right]^{-1}, \quad c = 1,\ldots,C. \tag{3.129}$$

Figure 3.24 shows the pseudo-code of training an i-vector extractor.

When estimating the TV matrix $\mathbf{T}$, the likelihood of training data can also be maximized through minimizing the divergence [72]. The idea is to force the i-vectors to follow the standard Gaussian prior. This can be achieved by performing the following transformation after computing $\mathbf{T}_c$ in Eq. 3.129:

$$\mathbf{T}_c \leftarrow \mathbf{T}_c\mathbf{L} \quad \text{and} \quad \langle \mathbf{w}_i|\mathcal{O}_i\rangle \leftarrow \mathbf{L}^{-1}\langle \mathbf{w}_i|\mathcal{O}_i\rangle, \tag{3.130}$$

where $\mathbf{L}\mathbf{L}^{\mathsf{T}}$ is the Cholesky decomposition of $\frac{1}{M}\sum_{i=1}^{M}\langle \mathbf{w}_i\mathbf{w}_i^{\mathsf{T}}|\mathcal{O}_i\rangle$, where M is the number of training utterances. The advantage of performing these transformations is the speedup of the convergence [73].

To extract the i-vector $\mathbf{x}_i$ from an utterance with acoustic vectors $\mathcal{O}_i$, we substitute Eq. 3.128 into Eq. 3.123 and Eq. 3.125:

$$\begin{aligned} \mathbf{x}_i \equiv \langle \mathbf{w}_i|\mathcal{O}_i\rangle &= \mathbf{L}_i^{-1}\sum_{c=1}^{C}\mathbf{T}_c^{\mathsf{T}}\left(\boldsymbol{\Sigma}_c^{(b)}\right)^{-1}\tilde{\mathbf{f}}_{ic} \\ &= \mathbf{L}_i^{-1}\mathbf{T}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\tilde{\mathbf{f}}_i, \end{aligned} \tag{3.131}$$

where $\tilde{\mathbf{f}}_i = [\tilde{\mathbf{f}}_{i1}^{\mathsf{T}} \cdots \tilde{\mathbf{f}}_{iC}^{\mathsf{T}}]^{\mathsf{T}}$ and

$$\mathbf{L}_i = \mathbf{I} + \sum_{c=1}^{C} N_{ic}\mathbf{T}_c^{\mathsf{T}}(\boldsymbol{\Sigma}_c^{(b)})^{-1}\mathbf{T}_c = \mathbf{I} + \mathbf{T}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\mathbf{N}_i\mathbf{T}, \tag{3.132}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$M =$ no. of training utterances

Randomize $\mathbf{T} \in \Re^{CF \times R}$ with submatrices denoted by $\mathbf{T}_c \in \Re^{F \times R}$, $c = 1, ..., C$

***foreach* iteration**

for $i = 1$: no. of training utterances

$$\mathbf{N}_i = \begin{bmatrix} N_{i1}\mathbf{I}_{F\times F} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & N_{i2}\mathbf{I}_{F\times F} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & N_{iC}\mathbf{I}_{F\times F} \end{bmatrix}_{CF\times CF}$$

$$\mathbf{L}_i = \mathbf{I} + \mathbf{T}^T\boldsymbol{\Sigma}^{-1}\mathbf{N}_i\mathbf{T} \qquad \mathbf{L}_i \in \Re^{R\times R} \text{ and } \mathbf{T} \in \Re^{CF\times R}$$

$$\mathbf{x}_i = \mathbf{L}_i^{-1}\mathbf{T}^T\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_i \qquad \mathbf{x}_i \in \Re^R$$

end

$$\boldsymbol{\Omega}_c = \sum_{i=1}^{M} N_{ic}\left[\mathbf{L}_i^{-1} + \mathbf{x}_i\mathbf{x}_i^T\right] \qquad \boldsymbol{\Omega}_c \in \Re^{R\times R}$$

$$\boldsymbol{\Psi} = \sum_{i=1}^{M} \tilde{\mathbf{f}}_i\mathbf{x}_i^T \qquad \boldsymbol{\Psi} \in \Re^{CF\times R}$$

$$\mathbf{T}_c = \boldsymbol{\Psi}_c\boldsymbol{\Omega}_c^{-1}$$

end

Figure 3.24 The pseudo-code for computing the total variability matrix.

where $\mathbf{N}_i$ is a $CF \times CF$ block diagonal matrix containing $N_{ic}\mathbf{I}$, $c = 1, \ldots, C$, as its block diagonal elements.

Eq. 3.131 and Eq. 3.132 suggest that given a total variability matrix $\mathbf{T}$, an UBM, and an utterance with acoustic vectors $\mathcal{O}_i$, the i-vector of the utterance can be extracted by firstly aligning every frame in $\mathcal{O}_i$ with the UBM using Eq. 3.126 to compute the mixture posteriors $\gamma(\ell_{i,t,c})$'s. These posteriors, together with the mean vectors in the UBM, allow us to compute the zeroth and first-order sufficient statistics. Substituting these statistics into Eq. 3.132 and Eq. 3.131 leads to the i-vector. Figure 3.25 shows the procedure of i-vector extraction.

As will be discussed in Section 3.6.10, the posteriors $\gamma(\ell_{i,t,c})$'s do not have to be computed by aligning individual frames with the UBM. In the senone i-vectors, the mixture components c are replaced by senones and $\gamma(\ell_{i,t,c})$'s are replaced by senone posteriors. The latter can be estimated by a phonetic-aware deep neural network. Note that in senone i-vectors, a UBM is still required because Eq. 3.131 and Eq. 3.132 involve C covariance matrices. However, instead of computing these matrices through the EM algorithm, we may compute these matrices using the senone posteriors and the acoustic vectors in one iteration. Figure 3.26 shows the procedure of senone i-vector extraction.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

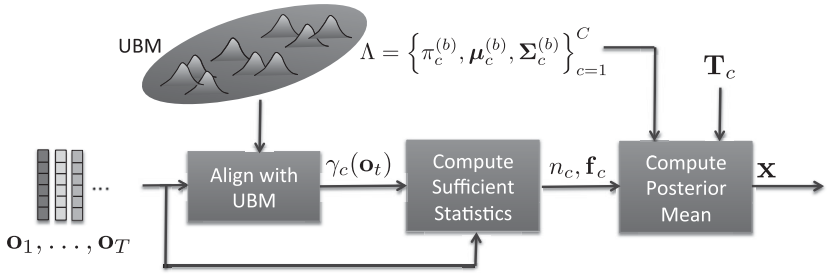


Figure 3.25 The procedure of GMM i-vector extraction. For clarity, the utterance index i is omitted.

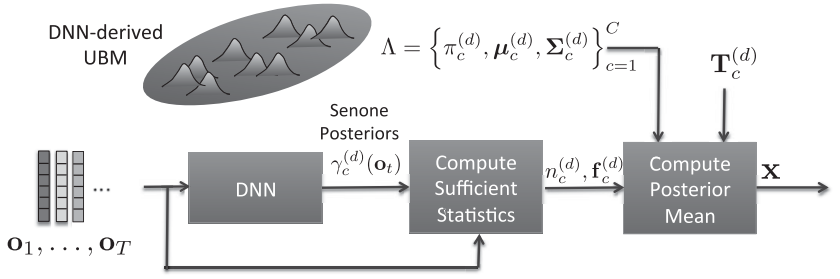


Figure 3.26 The procedure of DNN senone i-vector extraction. For clarity, the utterance index i is omitted.

3.6.4 Relation with MAP Adaptation in GMM–UBM

It can be shown that MAP adaptation in GMM–UBM [7] is a special case of the factor analysis model in Eq. 3.119. Specifically, when $\mathbf{T}$ is a $CF \times CF$ diagonal matrix denoted as $\mathbf{D}$, then Eq. 3.119 can be rewritten as

$$\boldsymbol{\mu}_i = \boldsymbol{\mu}^{(b)} + \mathbf{D}\mathbf{z}_i. \tag{3.133}$$

Using Eq. 3.131, we obtain the posterior mean of latent factor $\mathbf{z}_i$

$$\langle \mathbf{z}_i | \mathcal{O}_i \rangle = \mathbf{L}_i^{-1} \mathbf{D}^{\mathsf{T}} (\boldsymbol{\Sigma}^{(b)})^{-1} \tilde{\mathbf{f}}_i,$$

where $\mathcal{O}_i$ comprises the MFCC vectors of utterance i. Define r as the relevance factor in MAP adaptation such that $r\mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\mathbf{D} = \mathbf{I}$. Then, Eq. 3.132 becomes

$$\begin{aligned} \mathbf{L}_i &= r\mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\mathbf{D} + \mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\mathbf{N}_i\mathbf{D} \\ &= \mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}(r\mathbf{I} + \mathbf{N}_i)\mathbf{D}. \end{aligned}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

As a result, the offset to the UBM's supervector $\boldsymbol{\mu}^{(b)}$ for utterance i is

$$\begin{aligned}\mathbf{D}\langle \mathbf{z}_i|\mathcal{O}_i\rangle &= \mathbf{D}\left[\mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}(r\mathbf{I}+\mathbf{N}_i)\mathbf{D}\right]^{-1}\mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\tilde{\mathbf{f}}_i \\ &= \mathbf{D}\left[\mathbf{D}^{-1}(r\mathbf{I}+\mathbf{N}_i)^{-1}(\boldsymbol{\Sigma}^{(b)})\mathbf{D}^{-\mathsf{T}}\right]\mathbf{D}^{\mathsf{T}}(\boldsymbol{\Sigma}^{(b)})^{-1}\tilde{\mathbf{f}}_i \\ &= (r\mathbf{I}+\mathbf{N}_i)^{-1}\tilde{\mathbf{f}}_i.\end{aligned} \tag{3.134}$$

Recall from Eq. 3.16 that given utterance i, the MAP adaptation of mixture c in GMM–UBM is

$$\begin{aligned}\boldsymbol{\mu}_{i,c} &= \alpha_{i,c}E_c(\mathcal{O}_i) + (1-\alpha_{i,c})\boldsymbol{\mu}_c^{(b)} \\ &= \frac{N_{i,c}}{N_{i,c}+r}\frac{\mathbf{f}_{i,c}}{N_{i,c}} + \boldsymbol{\mu}_c^{(b)} - \frac{N_{i,c}}{N_{i,c}+r}\boldsymbol{\mu}_c^{(b)} \\ &= \boldsymbol{\mu}_c^{(b)} + \frac{1}{N_{i,c}+r}(\mathbf{f}_{i,c} - N_{i,c}\boldsymbol{\mu}_c^{(b)}) \\ &= \boldsymbol{\mu}_c^{(b)} + (r\mathbf{I}+\mathbf{N}_{i,c})^{-1}\tilde{\mathbf{f}}_{i,c},\end{aligned} \tag{3.135}$$

where r is the relevance factor, $N_{i,c}$ is the zeroth order sufficient statistics of mixture c, and $\mathbf{N}_{i,c} = \text{diag}\{N_{i,c}, \ldots, N_{i,c}\}$ is the cth block of matrix $\mathbf{N}_i$. Note that the offset to UBM's means in Eq. 3.135 is equivalent to Eq. 3.134.

3.6.5 I-Vector Preprocessing for Gaussian PLDA

It is necessary to preprocess the i-vectors before using the Gaussian PLDA because it requires the i-vectors to follow a Gaussian distribution. Gaussianization of i-vectors involves two steps. First, a whitening transform is applied to i-vectors:

$$\mathbf{x}^{\text{wht}} = \mathbf{W}^{\mathsf{T}}(\mathbf{x} - \bar{\mathbf{x}}), \tag{3.136}$$

where $\mathbf{W}$ is the Cholesky decomposition of the within-class covariance matrix of i-vectors [74], $\bar{\mathbf{x}}$ is the global mean of i-vectors, and $\mathbf{x}^{\text{wht}}$ is the whitened i-vector. In the second step, we apply a simple length-normalization to the whitened i-vectors:

$$\mathbf{x}^{\text{len-norm}} = \frac{\mathbf{x}^{\text{wht}}}{\|\mathbf{x}^{\text{wht}}\|}. \tag{3.137}$$

It is a common practice to include LDA (or NDA) and within-class covariance normalization (WCCN) [74] in the preprocessing steps. The whole preprocessing can be written in a more concise form:

$$\mathbf{x} \leftarrow \frac{\mathbf{P}(\mathbf{x} - \bar{\mathbf{x}})}{\|\mathbf{x}^{\text{wht}}\|}, \tag{3.138}$$

where $\mathbf{P}$ denotes the transformation matrix that combines whitening, LDA and WCCN, and $\mathbf{x}$ on the left of Eq. 3.138 is the preprocessed i-vector that is ready for PLDA modeling.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

3.6.6 Session Variability Suppression

Because the total variability matrix $\mathbf{T}$ models not only the speaker variability but also other variabilities in the acoustic vectors, it is important to remove the unwanted variabilities in the i-vectors. This can be done by minimizing within-speaker scatter and maximizing between-speaker scatter.

LDA+WCCN

A classical approach to session variability suppression is to apply linear discriminant analysis (LDA) followed by within-class covariance normalization (WCCN). As shown in Figure 3.27, given the utterances of a target speaker s and a test speaker t, the verification score is the cosine of the angle between the two transformed vectors:

$$S_{\text{CD}}(\mathbf{x}_s, \mathbf{x}_t) = \frac{\left(\mathbf{W}^{\mathsf{T}}\mathbf{x}_s\right) \cdot \left(\mathbf{W}^{\mathsf{T}}\mathbf{x}_t\right)}{\left\|\mathbf{W}^{\mathsf{T}}\mathbf{x}_s\right\| \left\|\mathbf{W}^{\mathsf{T}}\mathbf{x}_t\right\|}, \tag{3.139}$$

where $\mathbf{W}$ is the combined LDA+WCCN transformation matrix.

The LDA+WCCN transformation matrix can be estimated by supervised learning as follows. Denote $\mathbf{A}$ and $\mathbf{B}$ as the LDA and WCCN transformation matrices, respectively. Therefore, the combined matrix is $\mathbf{W} = \mathbf{BA}$. To find $\mathbf{A}$ and $\mathbf{B}$, we need a number of training speakers, each providing a number of utterances (i-vectors). Assume that we have K training speakers whose i-vectors are denoted as $\mathbf{x}_n$'s. To find $\mathbf{A}$, we maximize the between-speaker scatter and minimize the within-speaker scatter after the transformation. This can be achieved by maximizing the LDA criterion:

$$J(\mathbf{A}) = \frac{\text{Between-speaker scatter}}{\text{Within-speaker scatter}} = \text{Tr}\left\{\left(\mathbf{A}^{\mathsf{T}}\mathbf{S}_b\mathbf{A}\right)\left(\mathbf{A}^{\mathsf{T}}\mathbf{S}_w\mathbf{A}\right)^{-1}\right\}, \tag{3.140}$$

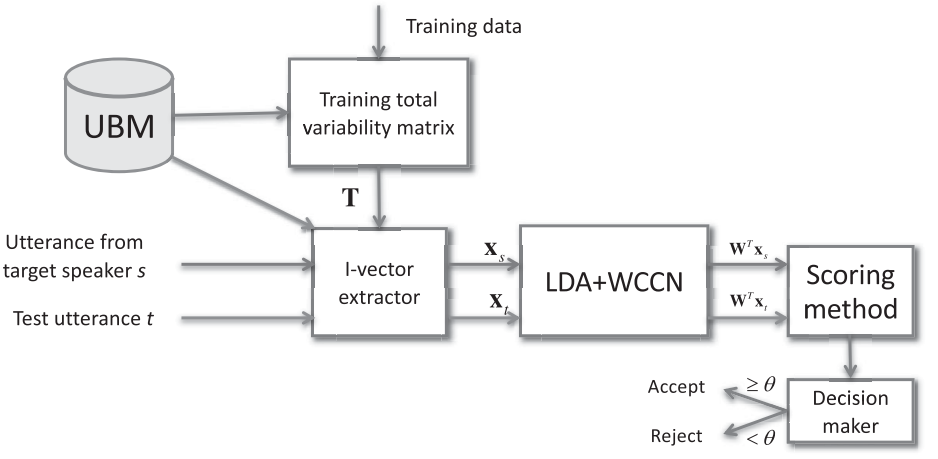


Figure 3.27 Preprocessing of i-vectors for cosine-distance or PLDA scoring.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\mathbf{S}_b$ and $\mathbf{S}_w$ are respectively the between-speaker and within-speaker scatter matrices

$$\mathbf{S}_b = \sum_{k=1}^{K} N_k(\bar{\mathbf{x}}_k - \bar{\mathbf{x}})(\bar{\mathbf{x}}_k - \bar{\mathbf{x}})^{\mathsf{T}} \tag{3.141a}$$

$$\mathbf{S}_w = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \bar{\mathbf{x}}_k)(\mathbf{x}_n - \bar{\mathbf{x}}_k)^{\mathsf{T}}. \tag{3.141b}$$

In Eq. 3.141, $\bar{\mathbf{x}}_k$ is the mean i-vector of speaker k, $\bar{\mathbf{x}}$ is the global mean of i-vectors, $\mathcal{C}_k$ comprises the utterance indexes of speaker k, and N_k is the number of samples in the class k, i.e., $N_k = |\mathcal{C}_k|$.

Eq. 3.140 can be framed as a constrained optimization problem:

$$\begin{array}{ll} \max_{\mathbf{A}} & \mathrm{Tr}\{\mathbf{A}^{\mathsf{T}}\mathbf{S}_b\mathbf{A}\} \\ \text{subject to} & \mathbf{A}^{\mathsf{T}}\mathbf{S}_w\mathbf{A} = \mathbf{I}, \end{array} \tag{3.142}$$

where $\mathbf{I}$ is an $M \times M$ identity matrix. Denote $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_{K'}]$. To find $\mathbf{a}_j$, $j = 1, \ldots, K'$, we write the Lagrangian function as:

$$L(\mathbf{a}_j, \lambda_j) = \mathbf{a}_j^{\mathsf{T}}\mathbf{S}_b\mathbf{a}_j - \lambda_j(\mathbf{a}_j^{\mathsf{T}}\mathbf{S}_w\mathbf{a}_j - 1),$$

where λ_j is a Lagrange multiplier. Setting $\frac{\partial L}{\partial \mathbf{w}_j} = 0$, we obtain the optimal solution of $\mathbf{a}_j$, which satisfies

$$(\mathbf{S}_w^{-1}\mathbf{S}_b)\mathbf{a}_j = \lambda_j \mathbf{a}_j.$$

Therefore, $\mathbf{A}$ comprises the first K' eigenvectors of $\boldsymbol{S}_w^{-1}\mathbf{S}_b$. A more formal proof can be find in [75]. As the maximum rank of $\mathbf{S}_b$ is $K - 1$, $\boldsymbol{S}_w^{-1}\mathbf{S}_b$ has at most $K - 1$ nonzero eigenvalues. As a result, K' can be at most $K - 1$. This suggests that for practical systems, a few hundred training speakers are required. Figure 3.28 shows the effect of LDA+WCCN on i-vectors. The diagram shows the projection of the original i-vectors and the LDA+WCCN-projected i-vectors on a three-dimensional t-SNE space [4]. Evidently, the LDA is very effective in separating the i-vectors of different speakers.

After finding $\mathbf{A}$, all training i-vectors are transformed, i.e., $\mathbf{A}^{\mathsf{T}}\mathbf{x}_n, \forall n$. Then, a within-class covariance matrix is computed based on the LDA-transformed i-vectors:

$$\mathbf{W}_{wccn} = \sum_{k=1}^{K} \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{A}^{\mathsf{T}}(\mathbf{x}_n - \bar{\mathbf{x}}_k)(\mathbf{x}_n - \bar{\mathbf{x}}_k)^{\mathsf{T}}\mathbf{A}. \tag{3.143}$$

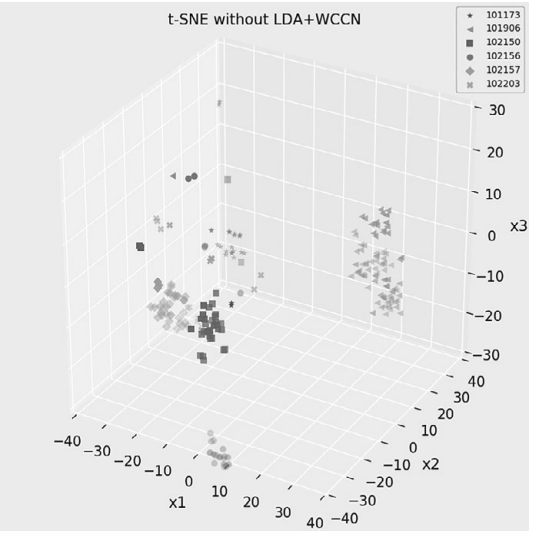
Then, the matrix $\mathbf{B}$ can be obtained from the Cholesky decomposition of $\mathbf{W}_{wccn}$, i.e., $\mathbf{B}\mathbf{B}^{\mathsf{T}} = \mathbf{W}_{wccn}$.

Once we have matrices $\mathbf{A}$ and $\mathbf{B}$, we may apply LDA+WCCN transformation on all i-vectors, including test i-vectors as follows:

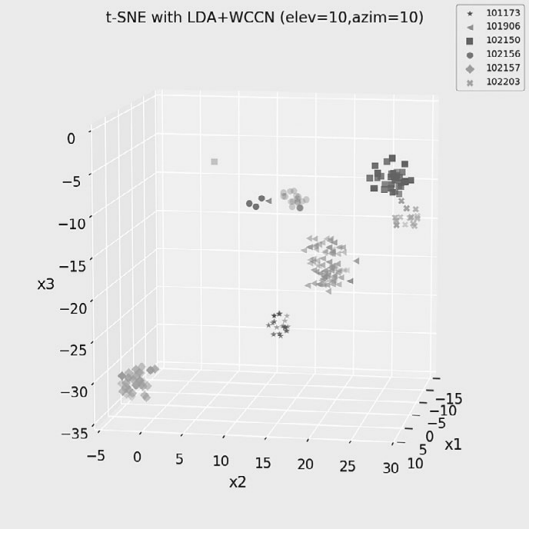
$$\mathbf{x} \leftarrow \mathbf{W}^{\mathsf{T}}\mathbf{x} = \mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{x}.$$

As Eq. 3.142 suggests, LDA aims to maximize speaker discrimination. WCCN, on the other hand, aims to inversely scale the LDA-projected space according to the

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press



(a)



(b)

Figure 3.28 T-SNE plot of I-vectors (a) before and (b) after LDA+WCCN transformation. Each marker type represents a speaker. The legend shows the speaker IDs.

within-speaker covariance matrix (Eq. 3.143). After the scaling, the directions of high intra-speaker variability will be de-emphasized. Thus, WCCN has the effect of variance normalization.

NDA

In LDA, it is assumed that samples in individual classes follow a Gaussian distribution. It is also assumed that these samples share the same covariance matrix. Because the

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

within-class and between-class scatter matrices are estimated empirically from training data, their accuracy depends on a number of factors. One important factor is the accuracy of the class-dependent means. Because in practical situation, the number of sessions per training speaker is not very large and some speakers may only have a few sessions, some of the class-dependent means could be very inaccurate. If the effect of these inaccurate means is not suppressed, the performance of LDA will suffer.

To address the limitation of LDA, a nonparametric discriminant analysis (NDA) [76] – which is originally proposed for face recognition – has successfully been applied to speaker verification [55, 56].

As mentioned earlier, in LDA, the class-dependent means are used for computing the within-class scatter matrix. On the other hand, in NDA, the class-dependent means are replaced by the nearest neighbors to each training vector. Moreover, instead of using the global mean and class-dependent means to compute the between-class scatter matrix, NDA uses all of the training vectors and their nearest neighbors from other classes to compute the between-class scatter matrix. It was found that this strategy is more effective than LDA in capturing the structural information of speaker boundaries.

Denote $\mathbf{x}_{ij}$ as the jth length-normalized i-vector from speaker i. In NDA, the nonparametric between-class and within-class scatter matrices are given by

$$\mathbf{S}_b^{\text{NDA}} = \sum_{i=1}^{S} \sum_{\substack{r=1 \\ r \neq i}}^{S} \sum_{l=1}^{K_b} \sum_{j=1}^{N_i} \omega(i,r,l,j)(\mathbf{x}_{ij} - \psi_l(\mathbf{x}_{ij},r))(\mathbf{x}_{ij} - \psi_l(\mathbf{x}_{ij},r))^{\mathsf{T}} \tag{3.144}$$

and

$$\mathbf{S}_w^{\text{NDA}} = \sum_{i=1}^{S} \sum_{l=1}^{K_w} \sum_{j=1}^{N_i} (\mathbf{x}_{ij} - \psi_l(\mathbf{x}_{ij},i))(\mathbf{x}_{ij} - \psi_l(\mathbf{x}_{ij},i))^{\mathsf{T}}, \tag{3.145}$$

respectively. In Eq. 3.144 and Eq. 3.145, N_i is the number of i-vectors from the ith speaker, S is the total number of speakers in the training set, $\psi_l(\mathbf{x}_{ij},r)$ is the lth nearest neighbor from speaker r to $\mathbf{x}_{ij}$, and K_b and K_w are the number of nearest neighbors selected for computing $\mathbf{S}_b^{\text{NDA}}$ and $\mathbf{S}_w^{\text{NDA}}$, respectively.

The weighting term $\omega(i,r,l,j)$ in Eq. 3.144 is defined as:

$$\begin{aligned} \omega(i,r,l,j) &= \frac{\min\{d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},i)), d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},r))\}}{d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},i)) + d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},r))} \\ &= \frac{1}{1+g(\mathbf{x}_{ij})^\alpha}, \qquad i \neq r \end{aligned} \tag{3.146}$$

where α controls the rate of change of the weight with respect to the distance ratio $g(\mathbf{x}_{ij})$ and $d(\mathbf{x}_p,\mathbf{x}_q)$ is the Euclidean distance between vector $\mathbf{x}_p$ and vector $\mathbf{x}_q$.

In Eq. 3.146, if $d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},i)) < d^\alpha(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},r))$, then

$$g(\mathbf{x}_{ij}) = \frac{d(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},r))}{d(\mathbf{x}_{ij},\psi_l(\mathbf{x}_{ij},i))},$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

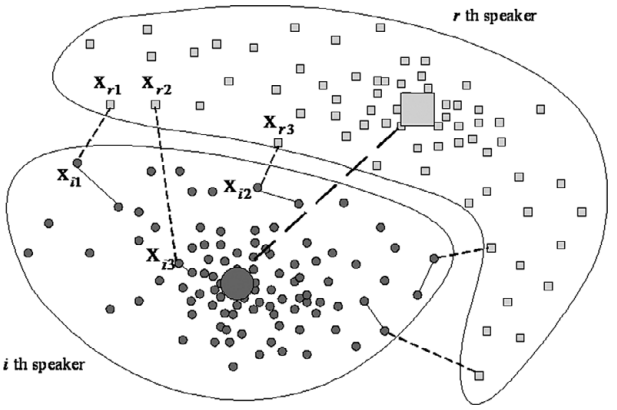


Figure 3.29 A two-dimensional example showing the within-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, i))$ [solid lines] and the between-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, r))$ [dashed lines] and for the training i-vectors $\mathbf{x}_{ij}$ from the ith speaker. For $\mathbf{x}_{i1}$ and $\mathbf{x}_{i2}$, the weighting function $\omega(\cdot)$ in Eq. 3.146 approaches 0.5. For $\mathbf{x}_{i3}$, $\omega(\cdot)$ approaches 0.0. [Reprinted from *SNR-Invariant PLDA Modeling in Nonparametric Subspace for Robust Speaker Verification (Figure 1), N. Li and M.W. Mak, IEEE/ACM Trans. on Audio Speech and Language Processing, vol. 23, no. 10, pp. 1648–1659, Oct. 2015, with permission of IEEE.*]

otherwise

$$g(\mathbf{x}_{ij}) = \frac{d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, i))}{d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, r))}. \tag{3.147}$$

If a selected i-vector $\mathbf{x}_{ij}$ is close to a speaker boundary (e.g., $\mathbf{x}_{i1}$ and $\mathbf{x}_{i2}$ in Figure 3.29), the between-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, r))$ [dashed line] will be comparable to the within-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, i))$ [solid line]. This will cause $g(\mathbf{x}_{ij})$ in Eq. 3.147 approaching 1.0. As a result, the weighting function $\omega(\cdot)$ in Eq. 3.146 will approach 0.5. On the other hand, if the selected i-vector $\mathbf{x}_{ij}$ is far away from a speaker boundary, e.g., $\mathbf{x}_{i3}$, the between-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, i))$ will be much smaller than the between-class distance $d(\mathbf{x}_{ij}, \psi_l(\mathbf{x}_{ij}, r))$. This will make the weighting function $\omega(\cdot)$ approaching 0.0. Consequently, the weighting function is able to emphasize the discriminative speaker boundary information in the training set. In [55], α was set to 2.

In LDA, the class-dependent means ($\bar{\mathbf{x}}_k$ in Eq. 3.141) are assumed to be the parameters of normal distributions. More precisely, each class has its own class-dependent mean but all classes share the same covariance matrix. In that sense, LDA is parametric. On the other hand, NDA is nonparametric because the terms in Eqs. 3.145 and 3.144 are vectors near the decision boundaries. They are not the parameters of normal distributions.

Similar to the standard LDA, the NDA projection matrix comprises the eigenvectors of $(\mathbf{S}_w^{\text{NDA}})^{-1}\mathbf{S}_b^{\text{NDA}}$.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

SVDA

SVDA (support vector discriminant analysis) [77] is a variant of LDA in that its objective function is the same as Eq. 3.140. The difference is that the between-speaker and within-speaker scatter matrices are computed by using the support vectors of linear SVMs that optimally separate any combinations of two speakers.

Given C speakers in a training set, there are $\frac{1}{2}C(C-1)$ combinations of speaker pairs. For each pair, we train a linear SVM to classify the two speakers in the pair. The linear SVM that separates the speakers pair (i, j), where $i < j$, has a decision function

$$\begin{aligned} f_{ij}(\mathbf{x}) &= \sum_{k \in \mathcal{S}_{ij}} \alpha_k y_k \mathbf{x}_k^{\mathsf{T}} \mathbf{x} + b_{ij} \\ &= \mathbf{w}_{ij}^{\mathsf{T}} \mathbf{x} + b_{ij}, \end{aligned}$$

where $\mathcal{S}_{ij}$ contains the support vector indexes for this speaker pair, α_k's and $y_k \in \{-1, +1\}$ are Lagrange multipliers and speaker labels, respectively, and b_{ij} is a bias term. As shown in Figure 3.8, $\boldsymbol{w}_{ij}$ is parallel to the optimal direction to classify speakers i and j. Therefore, we may use it to compute the within-speaker scatter matrix as follows:

$$\mathbf{S}_b = \sum_{1 \le i \le j \le C} \mathbf{w}_{ij} \mathbf{w}_{ij}^{\mathsf{T}}.$$

Similarly, the within-speaker scatter matrix can be computed from the support vectors only:

$$\mathbf{S}_w = \sum_{c=1}^{C} \sum_{k \in \mathcal{S}_c} (\mathbf{x}_k - \boldsymbol{\mu}_c)(\mathbf{x}_k - \boldsymbol{\mu}_c)^{\mathsf{T}},$$

where $\mathcal{S}_c$ contains the support vector indexes of speaker c and

$$\boldsymbol{\mu}_c = \frac{1}{|\mathcal{S}_c|} \sum_{k \in \mathcal{S}_c} \mathbf{x}_k$$

is the mean of support vectors from speaker c. Then, the SVDA projection matrix is obtained from the eigenvectors of $\mathbf{S}_w^{-1}\mathbf{S}_b$.

It was found in [77] that replacing LDA by SVDA can improve the performance by up to 32 percent.

3.6.7 PLDA versus Cosine-Distance Scoring

The cosine distance scoring in Eq. 3.139 simply computes the similarity between two i-vectors, regardless of their variability. In fact, all nonspeaker variabilities are assumed to be removed by the LDA+WCCN projection. One may think of this projection as creating a speaker subspace on which all projected i-vectors depend on speaker characteristics only. The PLDA discussed in Section 3.5, however, finds the speaker subspace by maximizing the likelihood of observed i-vectors using a factor analysis model and the speaker labels of the observed i-vectors. The advantage of PLDA is that it produces a probabilistic model through which true likelihood-ratio can be computed. As a result, in theory, score normalization [13] is not necessary.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

3.6.8 Effect of Utterance Length

An interesting question to ask is "will the modeling capability of i-vectors keeps on improving when the utterance duration increases." The answer is "no." This is because the information captured by i-vectors saturates when the utterances reach certain duration.

To demonstrate this saturation effect, we computed the intra- and inter-speaker cosine-distance scores of 272 speakers extracted from the telephone calls (phonecall_tel) and interview sessions (interview_mic) of SRE10. Each conversation was divided into a number of equal-length segments. By concatenating variable numbers of equal-length segments, sub-utterances of variable length were obtained. A voice activity detector (VAD) [78] was applied to remove the nonspeech regions in the sub-utterances. An i-vector was then estimated by using the acoustic vectors of each sub-utterance. Then, LDA and WCCN were applied to reduce the dimension of the vectors to 150 for computing cosine distance scores.

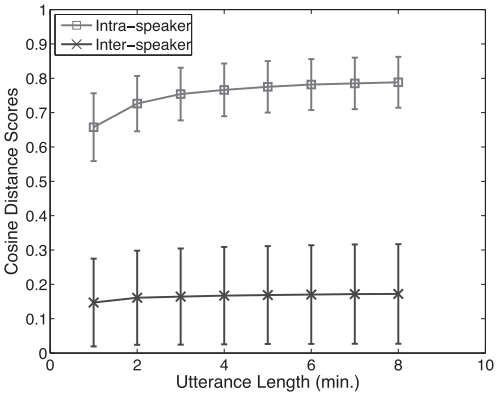
The mean intra- and inter-speaker scores (with error bars indicating two standard deviations) of the three types of speech are shown in Figure 3.30. The scores in "8-min interview_mic" were acquired from the 8-min interview sessions of 29 male speakers; each of these speakers give four interview-style conversations. Totally, for each utterance-length, there are 6496 inter-speaker scores and 174 intra-speaker scores. The scores in "3-min interview_mic" were acquired from the 3-minute interview sessions of 196 male speakers; each of them provide four interview sessions. This gives rise to 305,760 inter-speaker scores and 1176 intra-speaker scores for each utterance-length. For the "5-min phonecall_tel," the scores were acquired from the 5-minute telephone conversations of 47 male speakers, each giving four conversations. For each segment-length, there are 282 intra-speaker scores and 17,296 inter-speaker scores. Figure 3.30 clearly shows that both types of scores level off when the segment length is longer than a certain threshold.

Figure 3.31 shows the minimum decision cost (minDCF) versus the utterance length for estimating the i-vectors using the intra- and inter-speaker scores shown in Figure 3.30. The results further demonstrate the discriminative power of i-vectors with respect to the utterance duration. Evidently, the capability of discriminating speakers get saturated when the utterance duration is longer than two minutes. This result indicates that it may not be necessary to record very long utterances. From another perspective, when an utterance is sufficiently long, we may be able to leverage its long duration by dividing it into a number of sub-utterances so that more i-vectors can be derived, as illustrated in Figure 3.32 [79].

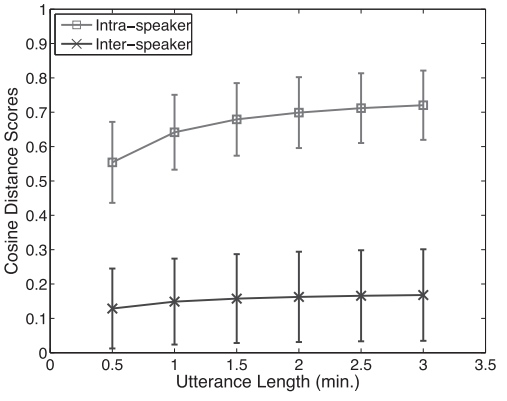
3.6.9 Gaussian PLDA with Uncertainty Propagation

While the i-vector/PLDA framework has been a great success, its performance drops rapidly when both the enrollment and test utterances have a wide range of durations. There are several reasons for this performance degradation. First, the duration of utterances is totally ignored in i-vector extraction, which means that the utterances are represented by vectors of fixed dimension without taking the utterance duration into account.

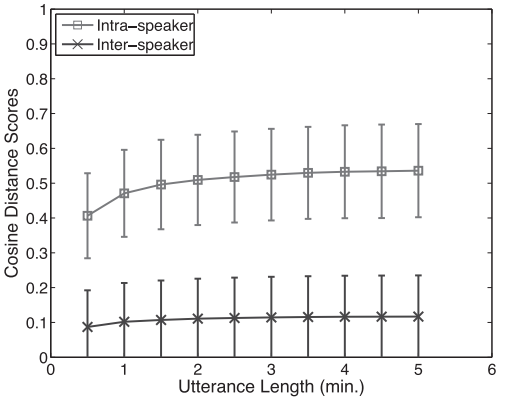
https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press



(a) 8-min interview_mic



(b) 3-min interview_mic



(c) 5-min phonecall_tel

Figure 3.30 Inter- and intra-speaker cosine-distance scores against utterance duration. (a) 8-min interview conversation; (b) 3-min interview conversation; (c) 5-min telephone conversation. [Reprinted from *Boosting the Performance of I-Vector Based Speaker Verification via Utterance Partitioning (Figure 1), W. Rao and M.W. Mak, IEEE Trans. on Audio Speech and Language Processing, vol. 21, no. 5, pp. 1012–1022, May 2013, with permission of IEEE.*]

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

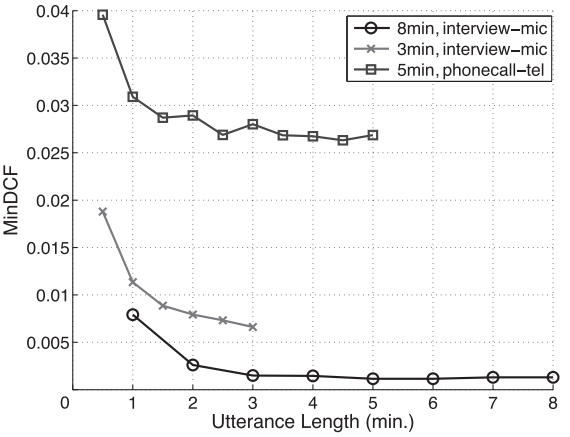


Figure 3.31 Minimum decision costs (MinDCF) versus utterance duration. The costs were based on the inter- and intra-speaker cosine-distances shown in Figure 3.30. [Reprinted from *Boosting the Performance of I-Vector Based Speaker Verification via Utterance Partitioning (Figure 2), W. Rao and M.W. Mak, IEEE Trans. on Audio Speech and Language Processing, vol. 21, no. 5, pp. 1012–1022, May 2013, with permission of IEEE.*]

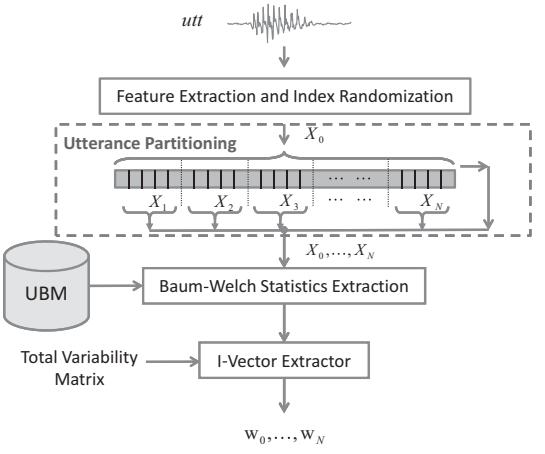


Figure 3.32 The utterance partitioning with acoustic vector resampling (UP-AVR) process. [Reprinted from *Boosting the Performance of I-Vector Based Speaker Verification via Utterance Partitioning (Figure 3), W. Rao and M.W. Mak, IEEE Trans. on Audio Speech and Language Processing, vol. 21, no. 5, pp. 1012–1022, May 2013, with permission of IEEE.*]

The accuracy of i-vectors depends on the number of acoustic vectors used for computing the posterior mean of the latent variable $\mathbf{w}$. If we do not take the utterance duration into consideration, we essentially treat all i-vectors to be equally reliable. Second, PLDA assumes that the intra-speaker variability (represented by $\boldsymbol{\Sigma}$ in Eq. 3.80) is the same across all i-vectors. This does not really reflect the real situation because intra-speaker variability in short utterances is larger than that of long utterances.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Kenny et al. [80] proposed to modify the standard PLDA to better accommodate utterance-length variability. The method aims to tie the i-vector extraction and PLDA modeling through propagating the uncertainty of i-vectors into the PLDA model. As shown in Eq. 3.132, the uncertainty (posterior covariance) of i-vectors depends on the number of acoustic vectors involves in computing the sufficient statistics. The longer the utterances, the smaller the uncertainty. By propagating the uncertainty to the PLDA model through the loading matrix, we will be able to model the variability in i-vectors due to difference in utterance duration. The resulting model will be better in handling the duration variability in utterances than the conventional PLDA model.

Preprocessing for Gaussian PLDA with UP

To apply UP, we need to apply the preprocessing steps in Section. 3.6.5 to the posterior covariance matrices as well. Assume that only a linear transformation $\mathbf{P}$ is applied to an i-vector, the corresponding preprocessed covariance matrix is

$$\text{cov}(\mathbf{Pw}, \mathbf{Pw}) = \mathbf{P}\mathbf{L}^{-1}\mathbf{P}^{\mathsf{T}}, \tag{3.148}$$

where $\mathbf{L}$ is the precision matrix in Eq. 3.132. If length-normalization is also applied, the preprocessed covariance matrix can be estimated as follows [80]:

$$\mathbf{\Lambda} \leftarrow \frac{\mathbf{P}\mathbf{L}^{-1}\mathbf{P}^{\mathsf{T}}}{\|\mathbf{x}^{\text{wht}}\|}. \tag{3.149}$$

Other methods to deal with this nonlinear transformation on the posterior matrix can be found in [80, 81].

Generative Model for Gaussian PLDA with UP

In uncertainty propagation, the i-vectors and the PLDA model are considered as a whole. Given an i-vector $\mathbf{x}_r$, its uncertainty is propagated to the PLDA model by adding an utterance-dependent loading matrix $\mathbf{U}_r$ to the generative model as follows:

$$\mathbf{x}_r = \mathbf{m} + \mathbf{V}\mathbf{z} + \mathbf{U}_r\mathbf{q}_r + \boldsymbol{\epsilon}_r. \tag{3.150}$$

In this model, $\mathbf{U}_r$ is the Cholesky decomposition of the posterior covariance matrix $\mathbf{\Lambda}_r$ of the i-vector, $\mathbf{q}_r$ is a latent variable with prior $\mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\boldsymbol{\epsilon}_r$ is the residue that follows $\mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$. The intra-speaker variability of $\mathbf{x}_r$ in Eq. 3.150 is:

$$\text{cov}(\mathbf{x}_r, \mathbf{x}_r|\mathbf{z}) = \mathbf{\Lambda}_r + \mathbf{\Sigma}, \tag{3.151}$$

where $\mathbf{\Lambda}_r$ varies from utterances to utterances, thus reflecting the reliability of i-vector $\mathbf{x}_r$.

Given a target speaker's i-vector $\mathbf{x}_s$ together with its posterior covariance matrix $\mathbf{\Lambda}_s$ and a test i-vector $\mathbf{x}_t$ together with its posterior covariance matrix $\mathbf{\Lambda}_t$, the log-likelihood ratio can be written as:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

$$
\begin{aligned}
S_{LR}(\mathbf{x}_s, \mathbf{x}_t; \mathbf{\Lambda}_s, \mathbf{\Lambda}_t) &= \log \frac{p(\mathbf{x}_s, \mathbf{x}_t; \mathbf{\Lambda}_s, \mathbf{\Lambda}_t | \text{same-speaker})}{p(\mathbf{x}_s, \mathbf{x}_t; \mathbf{\Lambda}_s, \mathbf{\Lambda}_t | \text{different-speaker})} \\
&= \log p\left(\begin{bmatrix}\mathbf{x}_s \\ \mathbf{x}_t\end{bmatrix} \middle| \begin{bmatrix}\mathbf{0} \\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}\mathbf{\Sigma}_s & \mathbf{\Sigma}_{ac} \\ \mathbf{\Sigma}_{ac} & \mathbf{\Sigma}_t\end{bmatrix}\right) \\
&\quad - \log p\left(\begin{bmatrix}\mathbf{x}_s \\ \mathbf{x}_t\end{bmatrix} \middle| \begin{bmatrix}\mathbf{0} \\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}\mathbf{\Sigma}_s & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_t\end{bmatrix}\right) \\
&= \frac{1}{2}\mathbf{x}_s^{\mathsf{T}}\mathbf{A}_{x,t}\mathbf{x}_s + \mathbf{x}_s^{\mathsf{T}}\mathbf{B}_{s,t}\mathbf{x}_t + \frac{1}{2}\mathbf{x}_t^{\mathsf{T}}\mathbf{C}_{s,t}\mathbf{x}_t + D_{s,t}
\end{aligned} \tag{3.152}
$$

where

$$\mathbf{A}_{s,t} = \mathbf{\Sigma}_s^{-1} - (\mathbf{\Sigma}_s - \mathbf{\Sigma}_t^{-1}\mathbf{\Sigma}_{ac})^{-1} \tag{3.153}$$

$$\mathbf{B}_{s,t} = \mathbf{\Sigma}_s^{-1}\mathbf{\Sigma}_{ac}(\mathbf{\Sigma}_t - \mathbf{\Sigma}_{ac}\mathbf{\Sigma}_s^{-1}\mathbf{\Sigma}_{ac})^{-1} \tag{3.154}$$

$$\mathbf{C}_{s,t} = \mathbf{\Sigma}_t^{-1} - (\mathbf{\Sigma}_t - \mathbf{\Sigma}_s^{-1}\mathbf{\Sigma}_{ac})^{-1} \tag{3.155}$$

$$D_{s,t} = -\frac{1}{2}\log\begin{vmatrix}\mathbf{\Sigma}_s & \mathbf{\Sigma}_{ac} \\ \mathbf{\Sigma}_{ac} & \mathbf{\Sigma}_t\end{vmatrix} + \frac{1}{2}\log\begin{vmatrix}\mathbf{\Sigma}_s & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_t\end{vmatrix} \tag{3.156}$$

$$\mathbf{\Sigma}_t = \mathbf{V}\mathbf{V}^{\mathsf{T}} + \mathbf{\Lambda}_t + \mathbf{\Sigma} \tag{3.157}$$

$$\mathbf{\Sigma}_s = \mathbf{V}\mathbf{V}^{\mathsf{T}} + \mathbf{\Lambda}_s + \mathbf{\Sigma} \tag{3.158}$$

$$\mathbf{\Sigma}_{ac} = \mathbf{V}\mathbf{V}^{\mathsf{T}}. \tag{3.159}$$

Note that Eqs. 3.153–3.156 involve terms dependent on both the target speaker's utterance and the test utterance, which means that these terms need to be evaluated during scoring. This means that uncertainty propagation is computationally expansive.

Fortunately, fast scoring methods [82] have been proposed to reduce the complexity of UP. The basic idea in [82] is to group i-vectors with similar reliability so that for each group the utterance-dependent loading matrices are replaced by a representative one. The procedure is as follows:

1. Compute the posterior covariance matrices from development data
2. For the kth group, select the representative $\mathbf{U}_k\mathbf{U}_k^{\mathsf{T}}$ as shown in Figure 3.33.

In [83], three approaches to finding the representative posterior covariances were investigated: (1) utterance duration, (2) mean of diagonal elements of posterior covariance matrices, and (3) based on the largest eigenvalue of posterior covariance matrices. With the above procedure, a set of representative matrices that cover all possible i-vectors can be precomputed. It has been demonstrated that by replacing the group

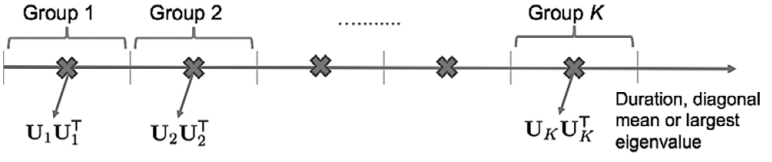


Figure 3.33 Representing the posterior covariance groups by one posterior covariance matrix.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

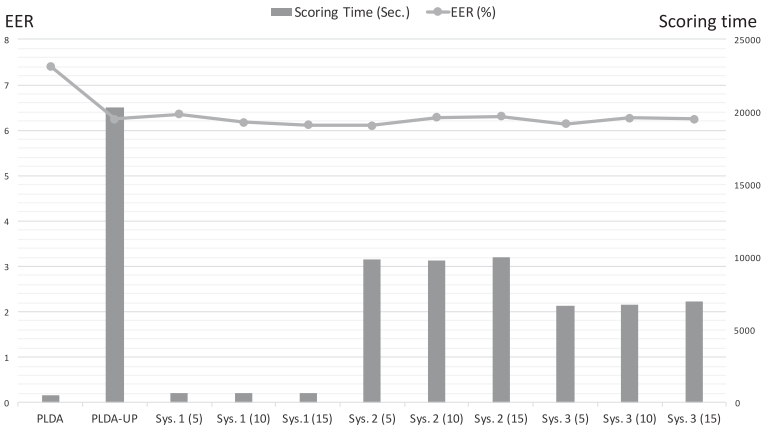


Figure 3.34 Equal error rates (EERs) and scoring time of PLDA, PLDA-UP, and fast PLDA-UP. For fast PLDA-UP, the EERs and scoring time of three systems, each with a different number of i-vector groups, were shown. The numbers within the parenthesis in the x-axis labels denote the numbers of i-vector groups. [Reprinted from *Fast Scoring for PLDA with Uncertainty Propagation via I-vector Grouping (Figure 2), W.W. Lin, M.W. Mak and J.T. Chien, Computer Speech and Language, vol. 45, pp. 503–515, 2017, with permission of Elsevier.*]

members by their representative, scoring time can be reduced by 33 times with negligible performance degradation. Figure 3.34 shows the dramatic reduction in the scoring time when this strategy is used.

3.6.10 Senone I-Vectors

Recently, Lei et al. [84] and Ferrer et al. [85] proposed to replace the UBM in the i-vector extractor by a deep neural network (DNN). The idea is to replace the zeroth order sufficient statistics (Eq. 3.126) by the outputs of a DNN trained for speech recognition. An advantage of this method is that it enables each test frame to be compared with the training frames for the same phonetic content.

One important characteristic of this method is that the acoustic features for speech recognition in the DNN are not necessarily the same as the features for the i-vector extractor. Specifically, denote the acoustic features of utterance i for the DNN and i-vector extractor as $\mathbf{a}_{it}$ and $\mathbf{o}_{it}$, respectively, where $t = 1, \ldots, T_i$. Then, the Baum–Welch statistics in Eq. 3.128 can be rewritten as:

$$N_{ic} \equiv \sum_{t=1}^{T_i} \gamma_c^{\text{DNN}}(\mathbf{a}_{it}) \text{ and } \tilde{\mathbf{f}}_{ic} \equiv \sum_{t=1}^{T_i} \gamma_c^{\text{DNN}}(\mathbf{a}_{it})(\mathbf{o}_{it} - \boldsymbol{\mu}_c), \tag{3.160}$$

where $\gamma_c^{\text{DNN}}(\mathbf{a}_{it})$'s are the senone posterior probabilities obtained from the DNN's outputs and $\boldsymbol{\mu}_c$'s are the probabilistic weighted sum of speaker features:

$$\boldsymbol{\mu}_c = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i} \gamma_c^{\text{DNN}}(\mathbf{a}_{it})\mathbf{o}_{it}}{\sum_{i=1}^{N}\sum_{t=1}^{T_i} \gamma_c^{\text{DNN}}(\mathbf{a}_{it})}. \tag{3.161}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Similar strategy can also be applied to Eq. 3.132:

$$\mathbf{L}_i = \mathbf{I} + \sum_{c=1}^{C} N_{ic}\mathbf{T}_c^{\mathsf{T}}(\mathbf{\Sigma}_c)^{-1}\mathbf{T}_c = \mathbf{I} + \mathbf{T}^{\mathsf{T}}\mathbf{\Sigma}^{-1}\mathbf{N}_i\mathbf{T}, \tag{3.162}$$

where N_{ic} is obtained in Eq. 3.160 and

$$\mathbf{\Sigma}_c = \frac{\sum_{i=1}^{N}\sum_{t=1}^{T_i}\gamma_c^{\text{DNN}}(\mathbf{a}_{it})\mathbf{o}_{it}\mathbf{o}_{it}^{\mathsf{T}}}{\sum_{i=1}^{N}\sum_{t=1}^{T_i}\gamma_c^{\text{DNN}}(\mathbf{a}_{it})} - \boldsymbol{\mu}_c\boldsymbol{\mu}_c^{\mathsf{T}}. \tag{3.163}$$

Note that $\mathbf{T}_c$ in Eq. 3.162 should be estimated by replacing $\tilde{\mathbf{f}}_{ic}$ in Eq. 3.129 with the first-order statistics in Eq. 3.160. Similarly, the posterior expectation in Eq. 3.123 should also be computed as follows:

$$\langle \mathbf{w}_i | \mathcal{O}_i \rangle = \mathbf{L}_i^{-1} \sum_{c=1}^{C} \mathbf{T}_c^{\mathsf{T}}\mathbf{\Sigma}_c^{-1}\tilde{\mathbf{f}}_{ic}, \tag{3.164}$$

where $\mathbf{L}_i$ and $\mathbf{\Sigma}_c$ are obtained from Eqs. 3.162 and 3.163, respectively. Because the mixture posteriors are now replaced by senone posteriors, the resulting i-vectors are termed senone i-vectors [3]. In some literature, they are also called DNN i-vectors.

Figure 3.35 shows an example of senone i-vector extraction [3]. The input $\mathbf{a}_{it}$ to the DNN comprises a context frames of acoustic features (could be MFCCs or filterbank outputs) centered at frame t and the vector $\mathbf{o}_{it}$ can be obtained from the bottleneck layer of the DNN.

3.7 Joint Factor Analysis

Before the introduction of i-vectors in 2010, joint factor analysis (JFA) [15] was the de facto standard for text-independent speaker verification. JFA is based on the assumption that a speaker- and channel-dependent supervector can be decomposed into the sum of two statistically independent and normally distributed supervectors: a speaker-dependent supervector and a channel-dependent supervector. Because a JFA model comprises three sets of latent factors, it is the most complex among all FA models in speaker recognition.

3.7.1 Generative Model of JFA

Denote $\mathbf{V}$ and $\mathbf{U}$ as the loading matrices representing the speaker- and channel subspaces, respectively. In JFA, the speaker- and channel-dependent supervectors $\mathbf{m}_h(s)$ can be generated by the following generative model:[9]

$$\mathbf{m}_h(s) = \mathbf{m} + \mathbf{V}\mathbf{y}(s) + \mathbf{U}\mathbf{x}_h(s) + \mathbf{D}\mathbf{z}(s), \tag{3.165}$$

[9] We follow the notations in [15, 86] as much as possible.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

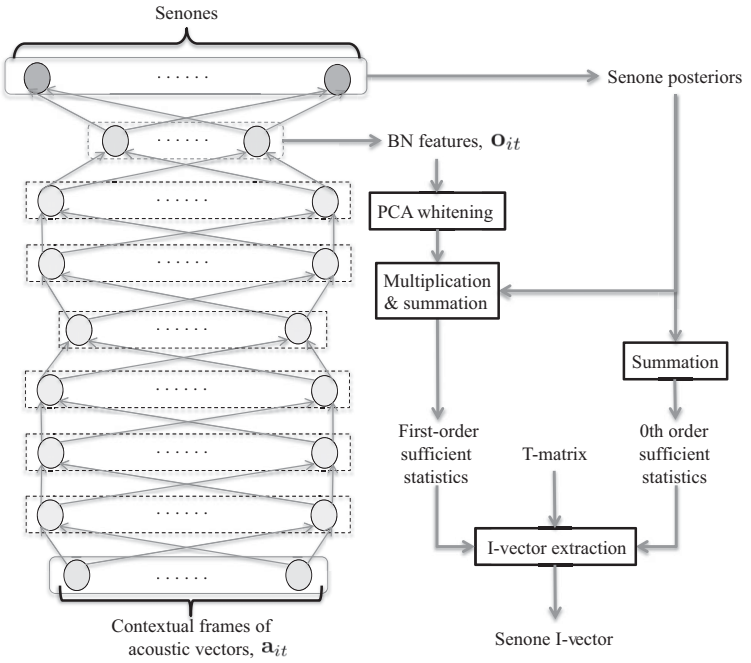


Figure 3.35 The procedure of extracting senone i-vectors. [Reprinted from *Denoised Senone I-Vectors for Robust Speaker Verification (Figure 2), Z.L. Tan, M.W. Mak, et al., IEEE/ACM Trans. on Audio Speech and Language Processing, vol. 26, no. 4, pp. 820–830, April 2018, with permission of IEEE.*]

where h and s stand for the recording (session) and speaker, respectively. In this model, $\mathbf{y}(s)$ is a speaker factor that depends on the speaker s and $\mathbf{x}_h(s)$ is a channel factor that depends on both the speaker and the session. Also, $\mathbf{D}$ is a diagonal matrix and the priors of $\mathbf{y}(s)$, $\mathbf{x}_h(s)$ and $\mathbf{z}(s)$ follow the standard Gaussian distribution. Figure 3.36 shows the graphical model of JFA.

Without the second and the third terms, Eq. 3.165 reduces to the classical MAP adaptation [7] where the adapted model is the mode of the posterior distribution of $\mathbf{m}_h(s)$. Also, without the third and the fourth terms, Eq. 3.165 reduces to the eigenvoice MAP [45]. Unlike Eq. 3.55, Eq. 3.165 does not have the residue term $\boldsymbol{\epsilon}$. This is because Eq. 3.165 is to generate the supervector $\mathbf{m}_h(s)$. If the supervector is observed, an residue term will need to be added to Eq. 3.165 to represent the difference between the observed and the generated supervectors.

3.7.2 Posterior Distributions of Latent Factors

To find the posterior distribution of all latent factors jointly, we may stack all the latent factors of speaker s with recording $h = 1, \ldots, H_s$ and pack the loading matrices as follows:

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

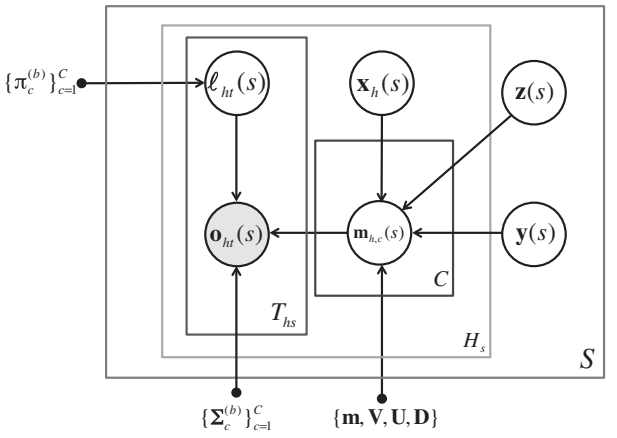


Figure 3.36 Graphical representation of the joint factor analysis model. $\boldsymbol{\ell}_{ht} = [\ell_{h,t,1} \ \ldots \ \ell_{h,t,c} \ \ldots \ \ell_{h,t,C}]^\mathsf{T}$ are the mixture labels in one-hot encoding. The priors of $\mathbf{x}_h(s)$, $\mathbf{z}(s)$, and $\mathbf{y}(s)$ are assumed to be standard Gaussians.

$$\begin{bmatrix} \mathbf{m}_1(s) \\ \vdots \\ \mathbf{m}_{H_s}(s) \end{bmatrix} = \begin{bmatrix} \mathbf{m} \\ \vdots \\ \mathbf{m} \end{bmatrix} + \begin{bmatrix} \mathbf{V} & \mathbf{U} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{D} \\ \mathbf{V} & \mathbf{0} & \mathbf{U} & \mathbf{0} & \cdots & \mathbf{D} \\ \vdots & \vdots & \ddots & \ddots & \cdots & \vdots \\ \mathbf{V} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{U} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{y}(s) \\ \mathbf{x}_1(s) \\ \vdots \\ \mathbf{x}_{H_s}(s) \\ \mathbf{z}(s) \end{bmatrix}. \tag{3.166}$$

Eq. 3.166 can be written in a compact form:[10]

$$\underline{\mathbf{m}}(s) = \underline{\mathbf{m}} + \underline{\mathbf{V}}(s)\underline{\mathbf{z}}(s), \tag{3.167}$$

which has the same form as Eq. 3.119. This means that we can use Eq. 3.132 to compute the posterior precision matrix of $\underline{\mathbf{z}}(s)$ for speaker s as follows:

$$\underline{\mathbf{L}}(s) = \underline{\mathbf{I}} + \underline{\mathbf{V}}(s)^\mathsf{T}\underline{\boldsymbol{\Sigma}}^{-1}(s)\underline{\mathbf{N}}(s)\underline{\mathbf{V}}(s), \tag{3.168}$$

In Eq. 3.168, $\underline{\mathbf{N}}(s)$ is a $CFH_s \times CFH_s$ block-diagonal matrix

$$\underline{\mathbf{N}}(s) = \begin{bmatrix} \mathbf{N}_1(s) & & & & \\ & \ddots & & & \\ & & \mathbf{N}_h(s) & & \\ & & & \ddots & \\ & & & & \mathbf{N}_{H(s)} \end{bmatrix},$$

where $\mathbf{N}_h(s)$ is a $CF \times CF$ diagonal matrix with elements

$$N_{hc}(s) = \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o}), \tag{3.169}$$

[10] While $\mathbf{V}$, $\mathbf{U}$, $\mathbf{D}$, and $\boldsymbol{\Sigma}$ are independent of s, $\underline{\mathbf{V}}(s)$ and $\underline{\boldsymbol{\Sigma}}(s)$ do, because their size depends on H_s.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\mathcal{O}_h(s)$ comprises the acoustic vectors of the hth recording of speaker s. Similarly, $\underline{\boldsymbol{\Sigma}}$ is a $CFH_s \times CFH_s$ block-diagonal matrix whose hth diagonal block is a $CF \times CF$ matrix $\boldsymbol{\Sigma}$. In practice, $\boldsymbol{\Sigma}$ can be substituted by the super-covariance matrix of the UBM $\boldsymbol{\Sigma}^{(b)}$.

Using Eq. 3.131, the posterior mean and the posterior moment of $\underline{\mathbf{z}}(s)$ are given by

$$\left\langle \underline{\mathbf{z}}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \underline{\mathbf{L}}^{-1}(s)\underline{\mathbf{V}}^{\mathsf{T}}(s)\underline{\boldsymbol{\Sigma}}^{-1}(s)\tilde{\underline{\mathbf{f}}}(s), \tag{3.170}$$

and

$$\left\langle \underline{\mathbf{z}}(s)\underline{\mathbf{z}}(s)^{\mathsf{T}} \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \underline{\mathbf{L}}^{-1}(s) + \left\langle \underline{\mathbf{z}}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle \left\langle \underline{\mathbf{z}}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle^{\mathsf{T}}, \tag{3.171}$$

respectively, where $\tilde{\underline{\mathbf{f}}}(s)$ is a CFH_s-dimensional vector whose cth segment of the hth block is an F-dimension vector

$$\tilde{\mathbf{f}}_{hc}(s) = \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{m}_c), \qquad h = 1, \ldots, H_s \text{ and } c = 1, \ldots, C. \tag{3.172}$$

3.7.3 Model Parameter Estimation

The training in JFA amounts to finding the model parameters $\boldsymbol{\Lambda} = \{\mathbf{m}, \mathbf{V}, \mathbf{U}, \mathbf{D}, \boldsymbol{\Sigma}\}$, where $\boldsymbol{\Sigma}$ is a diagonal covariance matrix. For each speaker s, we assume that there are H_s recording sessions. Define the joint latent factor

$$\hat{\mathbf{z}}_h(s) \equiv [\mathbf{y}(s) \;\; \mathbf{x}_h(s) \;\; \mathbf{z}(s)]^{\mathsf{T}}, \qquad h = 1, \ldots, H_s. \tag{3.173}$$

Also define the joint loading matrix $\hat{\mathbf{V}} \equiv [\mathbf{V} \;\; \mathbf{U} \;\; \mathbf{D}]$. Then, for recording h of speaker s, we have

$$\mathbf{m}_h(s) = \mathbf{m} + \hat{\mathbf{V}}\hat{\mathbf{z}}_h(s), \qquad h = 1, \ldots, H_s.$$

Using Eq. 3.170 and Eq. 3.171, we obtain the posterior mean and posterior covariance of $\hat{\mathbf{z}}_h(s)$ as follows:

$$\left\langle \hat{\mathbf{z}}_h(s) | \mathcal{O}_h(s) \right\rangle = \hat{\mathbf{L}}_h^{-1}(s)\hat{\mathbf{V}}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}_h(s) \tag{3.174}$$

and

$$\left\langle \hat{\mathbf{z}}_h(s)\hat{\mathbf{z}}_h(s)^{\mathsf{T}} | \mathcal{O}_h(s) \right\rangle = \hat{\mathbf{L}}_h^{-1}(s) + \left\langle \hat{\mathbf{z}}_h(s) | \mathcal{O}_h(s) \right\rangle \left\langle \hat{\mathbf{z}}_h(s) | \mathcal{O}_h(s) \right\rangle^{\mathsf{T}}. \tag{3.175}$$

In Eq. 3.174, $\tilde{\mathbf{f}}_h(s)$ is a $CF \times 1$ vector formed by stacking the C components of Eq. 3.172 and the precision matrix $\hat{\mathbf{L}}_h(s)$ is an $(R_v + R_u + CF) \times (R_v + R_u + CF)$ matrix

$$\hat{\mathbf{L}}_h(s) = \hat{\mathbf{I}} + \hat{\mathbf{V}}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}_h(s)\hat{\mathbf{V}}, \tag{3.176}$$

where R_v and R_u are the rank of $\mathbf{U}$ and $\mathbf{V}$, respectively, and $\hat{\mathbf{I}}$ is an $(R_v + R_u + CF) \times (R_v + R_u + CF)$ identity matrix.

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Using Eq. 3.80 and Eq. 3.129, the maximum-likelihood solution of $\hat{\mathbf{V}}$ is

$$\hat{\mathbf{V}}_c = \left[\sum_s \sum_{h=1}^{H_s} \tilde{\mathbf{f}}_{hc}(s) \left\langle \hat{\mathbf{z}}_h(s) | \mathcal{O}_h(s) \right\rangle^{\mathsf{T}}\right] \left[\sum_s \sum_{h=1}^{H_s} N_{hc}(s) \left\langle \hat{\mathbf{z}}_h(s) \hat{\mathbf{z}}_h(s)^{\mathsf{T}} | \mathcal{O}_h(s) \right\rangle\right]^{-1}. \tag{3.177}$$

Concatenating the latent factors as in Eq. 3.166 and Eq. 3.173 means that these factors are correlated in the posterior, i.e., the precision matrices $\underline{\mathbf{L}}(s)$ in Eq. 3.168 and $\hat{\mathbf{L}}_h(s)$ in Eq. 3.176 are non-diagonal. Because the dimension of $\mathbf{z}(s)$ is very high ($= 61,440$ for 1024 Gaussians and 60-dimensional acoustic vectors), using Eqs. 3.177, 3.174, and 3.175 to compute the model parameters is impractical.

A more practical approach is to estimate the loading matrices separately, as suggested in [87]. Specifically, the loading matrices are estimated using the following steps:

1. Estimate the eigenvoice matrix $\mathbf{V}$ by assuming that $\mathbf{U} = \mathbf{D} = \mathbf{0}$.
2. Given $\mathbf{V}$, estimate the eigenchannel matrix $\mathbf{U}$ by assuming that $\mathbf{D} = \mathbf{0}$.
3. Given $\mathbf{V}$ and $\mathbf{U}$, estimate $\mathbf{D}$.

With this simplification, the formulations in Step 1 are

$$\mathbf{m}_h(s) = \mathbf{m} + \mathbf{V}\mathbf{y}(s), \qquad h = 1, \ldots, H_s \tag{3.178a}$$

$$\mathbf{L}(s) = \mathbf{I} + \mathbf{V}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \mathbf{N}(s) \mathbf{V} \tag{3.178b}$$

$$\mathbf{N}(s) = \text{diag}\{N_1(s)\mathbf{I}, \ldots, N_C(s)\mathbf{I}\} \tag{3.178c}$$

$$N_c(s) = \sum_{h=1}^{H_s} \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o}) \tag{3.178d}$$

$$\tilde{\mathbf{f}}_c(s) = \sum_{h=1}^{H_s} \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{m}_c), \qquad c = 1, \ldots, C \tag{3.178e}$$

$$\left\langle \mathbf{y}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \mathbf{L}^{-1}(s) \mathbf{V}^{\mathsf{T}} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) \tag{3.178f}$$

$$\left\langle \mathbf{y}(s)\mathbf{y}(s)^{\mathsf{T}} \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \mathbf{L}^{-1}(s) + \left\langle \mathbf{y}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle \left\langle \mathbf{y}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle^{\mathsf{T}} \tag{3.178g}$$

$$\mathbf{V}_c = \left[\sum_s \tilde{\mathbf{f}}_c(s) \left\langle \mathbf{y}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle^{\mathsf{T}}\right] \left[\sum_s N_c(s) \left\langle \mathbf{y}(s)\mathbf{y}(s)^{\mathsf{T}} \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle\right]^{-1}. \tag{3.178h}$$

In Step 2, for all of the recordings from speaker s, we account for his/her speaker shift using the estimated $\mathbf{V}$ and speaker factor $\mathbf{y}(s)$, which results in the first-order statistics:

$$\tilde{\mathbf{f}}_{hc}(s) = \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{V}_c \mathbf{y}(s) - \mathbf{m}_c), \quad h = 1, \ldots, H_s \text{ and } c = 1, \ldots, C,$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where $\mathbf{y}(s)$ can be obtained from the posterior mean in Eq. 3.178f. Because we are modeling channel variability, there is no need to sum over the recordings of speaker s to compute the zeroth order statistic, i.e., $N_c(s)$ in Eq. 3.178d becomes

$$N_{hc}(s) = \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o}).$$

Also, the posterior precision matrix of $\mathbf{x}_h(s)$ is recording- and speaker-dependent:

$$\mathbf{L}_h(s) = \mathbf{I} + \mathbf{U}^\mathsf{T} \boldsymbol{\Sigma}^{-1} \mathbf{N}_h(s) \mathbf{U}.$$

Finally, we compute the posterior mean and posterior moment of $\mathbf{x}_h(s)$ and estimate the eigenchannel matrix $\mathbf{U}$ as follows:

$$\langle \mathbf{x}_h(s) | \mathcal{O}_h(s) \rangle = \mathbf{L}_h^{-1}(s) \mathbf{U}^\mathsf{T} \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}_h(s) \tag{3.179a}$$

$$\left\langle \mathbf{x}_h(s) \hat{\mathbf{x}}_h(s)^\mathsf{T} | \mathcal{O}_h(s) \right\rangle = \mathbf{L}_h^{-1}(s) + \langle \mathbf{x}_h(s) | \mathcal{O}_h(s) \rangle \langle \mathbf{x}_h(s) | \mathcal{O}_h(s) \rangle^\mathsf{T} \tag{3.179b}$$

$$\mathbf{U}_c = \left[\sum_s \sum_{h=1}^{H_s} \tilde{\mathbf{f}}_{hc}(s) \langle \mathbf{x}_h(s) | \mathcal{O}_h(s) \rangle^\mathsf{T} \right] \left[\sum_s \sum_{h=1}^{H_s} N_{hc}(s) \left\langle \mathbf{x}_h(s) \mathbf{x}_h(s)^\mathsf{T} | \mathcal{O}_h(s) \right\rangle \right]^{-1} \tag{3.179c}$$

In Step 3, for each recording from speaker s, we need to account for the speaker shift and the channel shift using the estimated $\mathbf{V}$ and $\mathbf{U}$, respectively. Using the same principle as in Step 2, we have the following zeroth- and first-order statistics:

$$N_c(s) = \sum_{h=1}^{H_s} \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o})$$

$$\tilde{\mathbf{f}}_c(s) = \sum_{h=1}^{H_s} \sum_{\mathbf{o} \in \mathcal{O}_h(s)} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{V}_c \mathbf{y}(s) - \mathbf{U}_c \mathbf{x}_h(s) - \mathbf{m}_c), \quad c = 1, \ldots, C.$$

The posterior precision matrix becomes

$$\mathbf{L}(s) = \mathbf{I} + \mathbf{D}^2 \boldsymbol{\Sigma}^{-1} \mathbf{N}(s),$$

where $\mathbf{N}(s) = \text{diag}\{N_1(s)\mathbf{I}, \ldots, N_C(s)\mathbf{I}\}$. Then, the diagonal elements of $\mathbf{D}$ can be estimated as follows:

$$\text{diag}\{\mathbf{D}_c\} = \frac{\text{diag}\left\{ \sum_s \tilde{\mathbf{f}}_c(s) \left\langle \mathbf{z}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle^\mathsf{T} \right\}}{\text{diag}\left\{ \sum_s N_c(s) \left\langle \mathbf{z}(s) \mathbf{z}(s)^\mathsf{T} \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle \right\}}, \tag{3.180}$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

where diag{$\mathbf{A}$} means converting the diagonal elements of $\mathbf{A}$ into a vector and the division is applied element-wise, and

$$\left\langle \mathbf{z}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \mathbf{L}^{-1}(s)\mathbf{D}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(s) \tag{3.181a}$$

$$\left\langle \mathbf{z}(s)\mathbf{z}(s)^{\mathsf{T}} \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle = \mathbf{L}^{-1}(s) + \left\langle \mathbf{z}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle \left\langle \mathbf{z}(s) \middle| \{\mathcal{O}_h(s)\}_{h=1}^{H_s} \right\rangle^{\mathsf{T}}. \tag{3.181b}$$

3.7.4 JFA Scoring

Scoring in JFA involves computing the likelihoods of acoustic vectors with respect to the target speaker and the UBM. There are several approaches to computing these likelihoods [88]. Here, we focus on the one based on the point estimates of the channel factors.

Point Estimate of Channel Factors

To express the likelihood of acoustic vectors, we need some notations. Let's define the channel-independent supervector of target-speaker s as

$$\mathbf{m}(s) = \mathbf{m} + \mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s), \tag{3.182}$$

where $\mathbf{m}$, $\mathbf{V}$, $\mathbf{D}$, $\mathbf{y}(s)$, and $\mathbf{z}(s)$ have the same meaning as those in Eq. 3.165. Note that while we use the same symbol s in both Eq. 3.182 and Eq. 3.165, the target speaker does not necessarily to be one of the training speakers in Eq. 3.165. Enrollment in JFA amounts to estimating $\mathbf{m}(s)$ for each target speaker using his/her enrollment utterances. The point estimates of $\mathbf{y}(s)$ and $\mathbf{z}(s)$ can be obtained from their posterior means.

Denote $\mathcal{O}_{\text{tst}}$ as the set of acoustic vectors from a test utterance. We compute the posterior mean of the channel factor given $\mathcal{O}$ and $\mathbf{m}(s)$:

$$\mathbf{x}_s \equiv \langle \mathbf{x} | \mathcal{O}_{\text{tst}}, \mathbf{m}(s) \rangle = (\mathbf{I} + \mathbf{U}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}\mathbf{U})^{-1}\mathbf{U}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(s),$$

where $\mathbf{N}$ comprises the zeroth order statistics along its diagonal and $\tilde{\mathbf{f}}(s)$ contains the first-order statistics, i.e., the cth component of $\tilde{\mathbf{f}}(s)$ is

$$\tilde{\mathbf{f}}_c(s) = \sum_{\mathbf{o} \in \mathcal{O}_{\text{tst}}} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{m}_c(s)) = \mathbf{f}_c - \mathbf{N}_c\mathbf{m}_c(s).$$

Similarly, we also compute the posterior mean of the channel factor based on the supervector of the UBM:

$$\mathbf{x}_{\text{ubm}} \equiv \langle \mathbf{x} | \mathcal{O}_{\text{tst}}, \mathbf{m} \rangle = (\mathbf{I} + \mathbf{U}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{N}\mathbf{U})^{-1}\mathbf{U}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(\text{ubm}),$$

where

$$\tilde{\mathbf{f}}_c(\text{ubm}) = \sum_{\mathbf{o} \in \mathcal{O}_{\text{tst}}} \gamma_c(\mathbf{o})(\mathbf{o} - \mathbf{m}_c) = \mathbf{f}_c - \mathbf{N}_c\mathbf{m}_c.$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

Using Theorem 1 of [86], the log-likelihood of $\mathcal{O}_{\text{tst}}$ with respect to the target speaker s is

$$\begin{aligned}\log p(\mathcal{O}_{\text{tst}}|\mathbf{m}(s),\mathbf{x}_s) = \sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{\frac{F}{2}}|\boldsymbol{\Sigma}_c|^{\frac{1}{2}}} - \frac{1}{2}\text{Tr}\{\boldsymbol{\Sigma}^{-1}\mathbf{S}\} \\ + \mathbf{r}(s)^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}} - \frac{1}{2}\mathbf{r}(s)^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{G}\mathbf{r}(s)\end{aligned} \tag{3.183}$$

where

$$\mathbf{r}(s) \equiv \begin{bmatrix}\mathbf{y}(s)\\ \mathbf{z}(s)\\ \mathbf{x}_s\end{bmatrix}, \qquad \mathbf{G} \equiv [\mathbf{V}\ \ \mathbf{D}\ \ \mathbf{U}]$$

and the cth component of $\mathbf{S}$ is

$$\mathbf{S}_c = \text{diag}\left\{\sum_{\mathbf{o}\in\mathcal{O}_{\text{tst}}} (\mathbf{o}-\mathbf{m}_c)(\mathbf{o}-\mathbf{m}_c)^{\mathsf{T}}\right\}.$$

Note that $\mathbf{y}(s)$ and $\mathbf{z}(s)$ – which are approximated by their posterior means – have been computed during enrollment, whereas $\mathbf{x}_s$ is computed during verification. Similarly, the the log-likelihood of $\mathcal{O}_{\text{tst}}$ with respect to the UBM is

$$\begin{aligned}\log p(\mathcal{O}|\mathbf{m},\mathbf{x}_{\text{ubm}}) = \sum_{c=1}^{C} N_c \log \frac{1}{(2\pi)^{\frac{F}{2}}|\boldsymbol{\Sigma}_c|^{\frac{1}{2}}} - \frac{1}{2}\text{Tr}\{\boldsymbol{\Sigma}^{-1}\mathbf{S}\} \\ + \mathbf{r}(\text{ubm})^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(\text{ubm}) - \frac{1}{2}\mathbf{r}(\text{ubm})^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{G}\mathbf{r}(\text{ubm})\end{aligned} \tag{3.184}$$

where

$$\mathbf{r}(\text{ubm}) = \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbf{x}_{\text{ubm}}\end{bmatrix}.$$

Using Eq. 3.183 and Eq. 3.184, the log-likelihood ratio (LLR) score for speaker verification can be computed as follows:

$$\begin{aligned}S_{\text{LLR}}(\mathcal{O}_{\text{tst}}|\mathbf{m}(s)) &= \log p(\mathcal{O}_{\text{tst}}|\mathbf{m}(s),\mathbf{x}_s) - \log p(\mathcal{O}|\mathbf{m},\mathbf{x}_{\text{ubm}}) \\ &= \mathbf{r}(s)^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(s) - \frac{1}{2}\mathbf{r}(s)^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{G}\mathbf{r}(s) \\ &\quad - \mathbf{r}(\text{ubm})^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(\text{ubm}) + \frac{1}{2}\mathbf{r}(\text{ubm})^{\mathsf{T}}\mathbf{G}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{G}\mathbf{r}(\text{ubm}).\end{aligned} \tag{3.185}$$

Linear Approximation

Define $\mathbf{q} \equiv \mathbf{G}\mathbf{r}(s)$ and consider the first-order Taylor expansion of

$$f(\mathbf{q}) = \mathbf{q}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\tilde{\mathbf{f}}(s) - \frac{1}{2}\mathbf{q}^{\mathsf{T}}\mathbf{N}\boldsymbol{\Sigma}^{-1}\mathbf{q}.$$

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press

We have $f(\mathbf{q}) \approx \mathbf{q}^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s)$ and therefore only the first and the third term in Eq. 3.185 remain:

$$\begin{aligned} S_{\text{LLR}}(\mathcal{O}_{\text{tst}}|\mathbf{m}(s)) &\approx \mathbf{r}(s)^\top \mathbf{G}^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) - \mathbf{r}(\text{ubm})^\top \mathbf{G}^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(\text{ubm}) \\ &= (\mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s) + \mathbf{U}\mathbf{x}_s)^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) - \mathbf{x}_{\text{ubm}}^\top \mathbf{U}^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(\text{ubm}). \end{aligned} \tag{3.186}$$

If we further assume that the first-order statistics are obtained from the target speaker $\mathbf{m}(s)$ for both the target speaker and the UBM and that $\mathbf{x}_s = \mathbf{x}_{\text{ubm}}$, we have

$$\begin{aligned} S_{\text{LLR}}(\mathcal{O}_{\text{tst}}|\mathbf{m}(s)) &\approx (\mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s) + \mathbf{U}\mathbf{x}_{\text{ubm}})^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) - (\mathbf{U}\mathbf{x}_{\text{ubm}})^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) \\ &= (\mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s))^\top \boldsymbol{\Sigma}^{-1} \tilde{\mathbf{f}}(s) \\ &= (\mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s))^\top \boldsymbol{\Sigma}^{-1} (\mathbf{f} - \mathbf{N}\mathbf{m}(s)) \\ &= (\mathbf{V}\mathbf{y}(s) + \mathbf{D}\mathbf{z}(s))^\top \boldsymbol{\Sigma}^{-1} (\mathbf{f} - \mathbf{N}\mathbf{m} - \mathbf{N}\mathbf{U}\mathbf{x}_{\text{ubm}}). \end{aligned} \tag{3.187}$$

3.7.5 From JFA to I-Vectors

One problem of JFA is its insufficiency in distinguishing between speaker and channel information. It was found in [16] that the channel factors also contain speaker information, causing imprecise modeling if the speaker subspace and channel subspace are modeled separately by two loading matrices. This finding motivates the development of i-vectors, which combines the two subspaces into one subspace called total variability space. Modeling in i-vectors involves two steps: (1) use low-dimensional vectors (called i-vectors) that comprise both speaker and channel information to represent utterances (see Section 3.6.3); and (2) model the channel variabilities of the i-vectors during scoring (see Sections 3.4.3 and 3.6.6).

https://doi.org/10.1017/9781108552332.004 Published online by Cambridge University Press