Chapter 1
Deep Neural Network Sequence-Discriminative Training

Abstract The cross-entropy criterion discussed in the previous chapters treats each frame independently. However, speech recognition is a sequence classification problem. In this chapter, we introduce the sequence-discriminative training techniques that match better to the problem. We describe the popular maximum mutual information (MMI), boosted MMI (BMMI), minimum phone error (MPE), and minimum Bayes risk (MBR) training criteria, and discuss the practical techniques, including lattice generation, lattice compensation, frame dropping, frame smoothing, and learning rate adjustment, to make DNN sequence-discriminative training effective.

1.1 Sequence-Discriminative Training Criteria

In the previous chapters, deep neural networks (DNNs) for speech recognition are trained to classify individual frames based on a cross-entropy (CE) criterion, which minimizes the expected frame error. However, speech recognition is a sequence classification problem. Sequence-discriminative training [8, 11, 9, 15, 16] seeks to better match the maximum a posteriori (MAP) decision rule of large vocabulary continous speech recognition (LVCSR) by considering sequence (inter-frame) constraints from hidden Markov models (HMMs), dictionary, and the language model (LM). Intuitively better recognition accuracy can be achieved if the CD-DNN-HMM speech recognizer is trained using sequence-discriminative criteria such as maximum mutual information (MMI) [1, 7] , boosted MMI (BMMI) [13], minimum phone error (MPE) [14] or minimum Bayes risk (MBR) [2] that have been proven to obtain state-of-the-art results in the GMM-HMM framework. Experimental results have shown that sequence-discriminative training can obtain from 3% to 17% relative error rate reduction against the CE trained models depends on the implementation and the dataset.

1.1.1 MMI

The MMI criterion [1, 7] used in automatic speech recognition (ASR) systems aims at maximizing the mutual information between the distributions of the observation sequence and the word sequence, which is highly correlated to minimizing the expected sentence error. Let us denote $\mathbf{o}^m = \mathbf{o}_1^m, \cdots, \mathbf{o}_t^m, \cdots, \mathbf{o}_{T_m}^m$ and $\mathbf{w}^m = \mathbf{w}_1^m, \cdots, \mathbf{w}_t^m, \cdots, \mathbf{w}_{N_m}^m$ the observation sequence and the correct word transcription of the m-th utterance, respectively, where T_m is the total number of frames in utterance m, and N_m is the total number of words in the transcription of the same utterance. The MMI criterion over the training set $\mathbb{S} = \{(\mathbf{o}^m, \mathbf{w}^m) | 0 \leq m < M\}$ is:

$$\begin{aligned} J_{MMI}(\theta; \mathbb{S}) &= \sum_{m=1}^{M} J_{MMI}(\theta; \mathbf{o}^m, \mathbf{w}^m) \\ &= \sum_{m=1}^{M} \log P(\mathbf{w}^m | \mathbf{o}^m; \theta) \\ &= \sum_{m=1}^{M} \log \frac{p(\mathbf{o}^m | \mathbf{s}^m; \theta)^\kappa P(\mathbf{w}^m)}{\sum_{\mathbf{w}} p(\mathbf{o}^m | \mathbf{s}^w; \theta)^\kappa P(\mathbf{w})}, \end{aligned} \tag{1.1}$$

where θ is the model parameter including DNN weight matrices and biases, $\mathbf{s}^m = s_1^m, \cdots, s_t^m, \cdots, s_{T_m}^m$ is the sequence of states corresponding to $\mathbf{w}^m$, and κ is the acoustic scaling factor. Theoretically the sum in the denominator should be taken over all possible word sequences. In practice, however, the sum is constrained by the decoded speech lattice for utterance m to reduce the computational cost. Note that the gradient of the criterion 1.1 with regard to the model parameters θ can be computed as

$$\begin{aligned} \nabla_\theta J_{MMI}(\theta; \mathbf{o}^m, \mathbf{w}^m) &= \sum_m \sum_t \nabla_{\mathbf{z}_{mt}^L} J_{MMI}(\theta; \mathbf{o}^m, \mathbf{w}^m) \frac{\partial \mathbf{z}_{mt}^L}{\partial \theta} \\ &= \sum_m \sum_t \ddot{\mathbf{e}}_{mt}^L \frac{\partial \mathbf{z}_{mt}^L}{\partial \theta}, \end{aligned} \tag{1.2}$$

where the error signal $\ddot{\mathbf{e}}_{mt}^L$ is defined as $\nabla_{\mathbf{z}_{mt}^L} J_{MMI}(\theta; \mathbf{o}^m, \mathbf{w}^m)$ and $\mathbf{z}_{mt}^L$ is the softmax layer's excitation (the value before softmax is applied) for utterance m at frame t. Since $\frac{\partial \mathbf{z}_{mt}^L}{\partial \theta}$ is irrelevant to the training criterion, the only difference the new training criterion introduces compared to the frame-level cross entropy training criterion **??** is the way the error signal is calculated. In the MMI training, the error signal becomes

$$
\begin{aligned}
\ddot{\mathbf{e}}_{mt}^{L}(i) &= \nabla_{\mathbf{z}_{mt}^{L}(i)} J_{MMI}(\theta; \mathbf{o}^m, \mathbf{w}^m) \\
&= \sum_r \frac{\partial J_{MMI}(\theta; \mathbf{o}^m, \mathbf{y}^m)}{\partial \log p(\mathbf{o}_t^m | r)} \frac{\partial \log p(\mathbf{o}_t^m | r)}{\partial \mathbf{z}_{mt}^{L}(i)} \\
&= \sum_r \kappa \left(\delta(r = s_t^m) - \frac{\sum_{\mathbf{w}: s_{t=r}} p(\mathbf{o}^m | \mathbf{s})^{\kappa} P(\mathbf{w})}{\sum_{\mathbf{w}} p(\mathbf{o}^m | \mathbf{s}^w)^{\kappa} P(\mathbf{w})} \right) \frac{\partial \log P(r | \mathbf{o}_t^m) - \log P(r) + \log p(\mathbf{o}_t^m)}{\partial \mathbf{z}_{mt}^{L}(i)} \\
&= \sum_r \kappa \left(\delta(r = s_t^m) - \ddot{\gamma}_{mt}^{DEN}(r) \right) \frac{\partial \log \mathbf{v}_{mt}^{L}(r)}{\partial \mathbf{z}_{mt}^{L}(i)} \\
&= \kappa \left(\delta(i = s_t^m) - \ddot{\gamma}_{mt}^{DEN}(i) \right),
\end{aligned} \tag{1.3}
$$

where $\ddot{\mathbf{e}}_{mt}^{L}(i)$ is the i-th element of the error signal, $\mathbf{v}_{mt}^{L}(r) = P(r|\mathbf{o}_t^m) = \text{softmax}_r\left(\mathbf{z}_{mt}^{L}\right)$ is the r-th output of the DNN,

$$
\ddot{\gamma}_{mt}^{DEN}(r) = \frac{\sum_{\mathbf{w}: s_{t=r}} p(\mathbf{o}^m | \mathbf{s})^{\kappa} P(\mathbf{w})}{\sum_{\mathbf{w}} p(\mathbf{o}^m | \mathbf{s}^w)^{\kappa} P(\mathbf{w})} \tag{1.4}
$$

is the posterior probability of being in state r at time t, computed over the denominator lattices for utterance m, $P(r)$ is the prior probability of state r, $p(\mathbf{o}_t^m)$ is the prior probability of observing $\mathbf{o}_t^m$, and $\delta(\bullet)$ is the Kronecker delta. Both $P(r)$ and $p(\mathbf{o}_t^m)$ are independent of $\mathbf{z}_{mt}^{L}$. Here we assumed that the nominator reference state labels are obtained through a forced alignment of the acoustics with the word transcript. If we consider all possible state sequences that lead to the reference transcription $\mathbf{w}^m$, we can use the forward-backward algorithm over the word reference to obtain the numerator occupancies $\ddot{\gamma}_{mt}^{NUM}(i)$ to replace $\delta(i = s_t^m)$.

If your DNN training algorithm is defined to minimize an objective function, you can, instead of maximizing the mutual information, minimize $J_{NMMI}(\theta; \mathbb{S}) = -J_{MMI}(\theta; \mathbb{S})$, in which case the error signals are negated. Note that criterion similar to MMI has been explored in the early ANN/HMM hybrid work [6].

1.1.2 Boosted MMI

The boosted MMI (BMMI) [13] criterion

$$
\begin{aligned}
J_{BMMI}\left(\theta;\mathbb{S}\right) &= \sum_{m=1}^{M} J_{BMMI}\left(\theta;\mathbf{o}^m,\mathbf{w}^m\right) \\
&= \sum_{m=1}^{M} \log \frac{P\left(\mathbf{w}^m|\mathbf{o}^m\right)}{\sum_{\mathbf{w}} P\left(\mathbf{w}|\mathbf{o}^m\right) e^{-bA(\mathbf{w},\mathbf{w}^m)}} \\
&= \sum_{m=1}^{M} \log \frac{p\left(\mathbf{o}^m|\mathbf{s}^m\right)^{\kappa} P\left(\mathbf{w}^m\right)}{\sum_{\mathbf{w}} p\left(\mathbf{o}^m|\mathbf{s}^w\right)^{\kappa} P\left(\mathbf{w}\right) e^{-bA(\mathbf{w},\mathbf{w}^m)}}
\end{aligned} \tag{1.5}
$$

is a variant of the MMI objective 1.1 to boost the likelihood of paths that contain more errors, where b, whose typical value is 0.5, is the boosting factor, and $A\left(\mathbf{w},\mathbf{w}^m\right)$ is a raw accuracy measure between word sequences $\mathbf{w}$ and $\mathbf{w}^m$ and can be computed at word level, phoneme level, or state level. For example, if it is measured at the phone level, it equals to the number of correct phones minus the number of insertions. This raw accuracy must be approximated efficiently. It has been shown that the BMMI criterion may be interpreted as incorporating a margin term in the MMI objective [13]. Since the only difference between the MMI and BMMI objective functions is the boosting term $e^{-bA(\mathbf{w},\mathbf{w}^m)}$ at the denominator, the error signal $\ddot{\ddot{\mathbf{e}}}^{L}_{mt}\left(i\right)$ can be similarly derived as

$$
\begin{aligned}
\ddot{\ddot{\mathbf{e}}}^{L}_{mt}\left(i\right) &= \nabla_{\mathbf{z}^{L}_{mt}(i)} J_{BMMI}\left(\theta;\mathbf{o}^m,\mathbf{w}^m\right) \\
&= \kappa\left(\delta\left(i=s^m_t\right) - \ddot{\ddot{\gamma}}^{DEN}_{mt}\left(i\right)\right),
\end{aligned} \tag{1.6}
$$

where, different from the MMI criterion, the denominator posterior probability is computed as

$$
\ddot{\ddot{\gamma}}^{DEN}_{mt}\left(i\right) = \frac{\sum_{\mathbf{w}:s_{t=i}} p\left(\mathbf{o}^m|\mathbf{s}\right)^{\kappa} P\left(\mathbf{w}\right) e^{-bA(\mathbf{w},\mathbf{w}^m)}}{\sum_{\mathbf{w}} p\left(\mathbf{o}^m|\mathbf{s}^w\right)^{\kappa} P\left(\mathbf{w}\right) e^{-bA(\mathbf{w},\mathbf{w}^m)}}. \tag{1.7}
$$

The extra computation involved in BMMI as opposed to MMI is very small if $A\left(\mathbf{w},\mathbf{w}^m\right)$ can be efficiently estimated. The only change occurs in the forward-backward algorithm on the denominator lattice. For each arc in the denominator lattice we subtract from the acoustic log-likelihood $bA\left(\mathbf{s},\mathbf{s}^m\right)$ that is corresponding to the arc. This behave is similar to modifying the language model contribution on each arc.

1.1.3 MPE/sMBR

The MBR [2, 8] family of objective functions aims at minimizing the expected error corresponding to different granularity of labels. For example, the MPE criterion aims to minimize the expected phone error, while state

MBR (sMBR) aims to minimize the expected state error when HMM topology and language models are taken into consideration. In general the MBR objective can be written as

$$\begin{aligned} J_{MBR}\left(\theta;\mathbb{S}\right) &= \sum_{m=1}^{M} J_{MBR}\left(\theta;\mathbf{o}^m,\mathbf{w}^m\right) \\ &= \sum_{m=1}^{M}\sum_{\mathbf{w}} P\left(\mathbf{w}|\mathbf{o}^m\right) A\left(\mathbf{w},\mathbf{w}^m\right) \\ &= \sum_{m=1}^{M} \frac{\sum_{\mathbf{w}} p\left(\mathbf{o}^m|\mathbf{s}^w\right)^{\kappa} P\left(\mathbf{w}\right) A\left(\mathbf{w},\mathbf{w}^m\right)}{\sum_{\mathbf{w}'} p\left(\mathbf{o}^m|\mathbf{s}^{w'}\right)^{\kappa} P\left(\mathbf{w}'\right)}, \end{aligned} \tag{1.8}$$

where $A\left(\mathbf{w},\mathbf{w}^m\right)$ is the raw accuracy between word sequences $\mathbf{w}$ and $\mathbf{w}^m$. For example, for MPE, it is the number of correct phone labels, while for sMBR it is the number of correct state labels. Similar to that in the MMI/BMMI case, the error signal is

$$\begin{aligned} \dddot{\mathbf{e}}_{mt}^{L}\left(i\right) &= \nabla_{\mathbf{z}_{mt}^{L}(i)} J_{MBR}\left(\theta;\mathbf{o}^m,\mathbf{w}^m\right) \\ &= \sum_{r} \frac{\partial J_{MBR}\left(\theta;\mathbf{o}^m,\mathbf{w}^m\right)}{\partial \log p\left(\mathbf{o}_t^m|r\right)} \frac{\partial \log p\left(\mathbf{o}_t^m|r\right)}{\partial \mathbf{z}_{mt}^{L}\left(i\right)} \\ &= \sum_{r} \kappa \dddot{\gamma}_{mt}^{DEN}\left(r\right)\left(\bar{A}^m\left(r=s_t^m\right)-\bar{A}^m\right)\frac{\partial \log \mathbf{v}_{mt}^{L}\left(r\right)}{\partial \mathbf{z}_{mt}^{L}\left(i\right)} \\ &= \kappa \dddot{\gamma}_{mt}^{DEN}\left(i\right)\left(\bar{A}^m\left(i=s_t^m\right)-\bar{A}^m\right), \end{aligned} \tag{1.9}$$

where$\bar{A}^m$ is the average accuracy of all paths in the lattice, $\bar{A}^m\left(r=s_t^m\right)$ is the average accuracy of all paths in the lattice for utterance m that passes through state r at time t, and $\dddot{\gamma}_{mt}^{DEN}\left(r\right)$ is the MBR *occupancy* statistics. For sMBR,

$$\dddot{\gamma}_{mt}^{DEN}\left(r\right) = \sum_{\mathbf{s}} \delta\left(r=s_t\right) P\left(\mathbf{s}|\mathbf{o}^m\right) \tag{1.10}$$

$$A\left(\mathbf{w},\mathbf{w}^m\right) = A\left(\mathbf{s}^w,\mathbf{s}^m\right) = \sum_{t} \delta\left(s_t^w = s_t^m\right) \tag{1.11}$$

$$\bar{A}^m\left(r=s_t^m\right) = \mathbb{E}\left\{A\left(\mathbf{s},\mathbf{s}^m\right)|s_t=r\right\} = \frac{\sum_{\mathbf{s}} \delta\left(r=s_t\right) P\left(\mathbf{s}|\mathbf{o}^m\right) A\left(\mathbf{s},\mathbf{s}^m\right)}{\sum_{\mathbf{s}} \delta\left(r=s_t\right) P\left(\mathbf{s}|\mathbf{o}^m\right)} \tag{1.12}$$

and

$$\bar{A}^m = \mathbb{E}\left\{A\left(\mathbf{s}, \mathbf{s}^m\right)\right\} = \frac{\sum_{\mathbf{s}} P\left(\mathbf{s}|\mathbf{o}^m\right) A\left(\mathbf{s}, \mathbf{s}^m\right)}{\sum_{\mathbf{s}} P\left(\mathbf{s}|\mathbf{o}^m\right)}. \tag{1.13}$$

1.1.4 A Uniformed Formulation

There can be other sequence-discriminative training criteria $J_{SEQ}\left(\theta; \mathbf{o}, \mathbf{w}\right)$. If the criteria are formulated as objective functions to be maximized (e.g., MMI/BMMI) instead of loss functions to be minimized we can always derive a loss function to be minimized by multiplying the original objective function by -1 . Such loss functions can always be formulated as a ratio of values computed from two lattices: the numerator lattice that represents the reference transcription and the denominator lattice that represents competing hypotheses. The expected occupancies $\gamma_{mt}^{NUM}\left(i\right)$ and $\gamma_{mt}^{DEN}\left(i\right)$ for each state i required by the extended Baum-Welch (EBW) algorithm are computed with forward-backward passes over the numerator and denominator lattices, respectively.

Note that the gradient of the loss with respect to state log-likelihood is

$$\frac{\partial J_{SEQ}\left(\theta; \mathbf{o}^m, \mathbf{w}^m\right)}{\partial \log p\left(\mathbf{o}_t^m|r\right)} = \kappa\left(\gamma_{mt}^{DEN}\left(r\right) - \gamma_{mt}^{NUM}\left(r\right)\right). \tag{1.14}$$

Since $\log p\left(\mathbf{o}_t^m|r\right) = \log P\left(r|\mathbf{o}_t^m\right) - \log P\left(r\right) + \log p\left(\mathbf{o}_t^m\right)$, by the chain rule,

$$\frac{\partial J_{SEQ}\left(\theta; \mathbf{o}^m, \mathbf{w}^m\right)}{\partial P\left(r|\mathbf{o}_t^m\right)} = \kappa\frac{\left(\gamma_{mt}^{DEN}\left(r\right) - \gamma_{mt}^{NUM}\left(r\right)\right)}{P\left(r|\mathbf{o}_t^m\right)}. \tag{1.15}$$

Given that $P\left(r|\mathbf{o}_t^m\right) = \text{ softmax}_r\left(\mathbf{z}_{mt}^L\right)$ we get

$$\mathbf{e}_{mt}^L\left(i\right) = \frac{\partial J_{SEQ}\left(\theta; \mathbf{o}^m, \mathbf{w}^m\right)}{\partial \mathbf{z}_{mt}^L\left(i\right)} = \kappa\left(\gamma_{mt}^{DEN}\left(i\right) - \gamma_{mt}^{NUM}\left(i\right)\right). \tag{1.16}$$

This formula can be applied to all the above sequence-discriminative training criteria as well as new ones [8, 16, 15]. The only difference is the way the occupancy statistics $\gamma_{mt}^{NUM}\left(i\right)$ and $\gamma_{mt}^{DEN}\left(i\right)$ are computed.

1.2 Practical Considerations

The above discussion seems to indicate that the sequence-discriminative training can be trivial. The only difference between the sequence-discriminative training and the frame-level cross entropy training is the more complicated error signal computation, which now involves numerator and denominator lat-

tices. In practice, however, many practical techniques would help and sometimes are critical to obtain good recognition accuracy.

1.2.1 Lattice Generation

Similar to training GMM-HMM systems, the first step in the sequence-discriminative training of DNNs is to generate the numerator and denominator lattices. As we have pointed out above, the numerator lattices are often reduced to forced-alignment of the transcription. It was shown that it is important to generate the lattices (especially the denominator lattices) by decoding the training data with a unigram LM in the GMM-HMM [12]. This still holds in the CD-DNN-HMM. In addition, people have found that it is desirable to use the best model available to generate the lattice and to serve as the seed model for the sequence-discriminative training [15]. Since CD-DNN-HMMs often outperform the CD-GMM-HMMs, we should at least use the CD-DNN-HMM trained with the CE criterion as both the seed model and the model for generating the alignments and lattices of each training utterance. Since lattice generation is an expensive process, the lattice is thus typically generated once and reused across training epochs. Further improvement can be obtained if new alignment and lattices are generated after each epoch. It is important that all the lattices are regenerated using the same model when doing so.

Table 1.1, based on results extracted from [15], clearly indicates the effect of the lattice quality and the seed model to the final recognition accuracy. From the table we can make several observations. First, compared to the CE1 model trained with the CE criterion and the alignment generated from the GMM model, the model trained with the sequence-discriminative training only obtains 2% relative error reduction if the lattices used are generated from the GMM model. However, we can obtain 13% relative error reduction if the lattice is generated from the CE1 model even though the same CE1 model is used as the seed model in both conditions. Second, if the same lattice generated from the CE1 model is used to generate statistics in the sequence-discriminative training, additional 2% relative error reduction can be obtained if we use CE2 instead of CE1 model as the seed model. The best result of 17% relative word error rate (WER) reduction can be obtained using the CE2 model as both the seed model and the model for generating the lattice.

Table 1.1 The effect (WER on Hub5'00 dataset) of the quality of the seed model and the lattice to the performance of the sequence-discriminative training on SWB 300hr task. Relative WER reduction over the CE1 model are in parentheses. CE1: trained with the alignment generated from GMM. CE2: refinement of CE1 model with alignment generated from CE1. (Summarized from Su et al. [15])

Model to Generate Lattice	Seed Model	
	CE1	CE2
GMM	15.8% (-2%)	-
DNN CE1 (WER 16.2%)	14.1% (-13%)	13.7% (-15%)
DNN CE2 (WER 15.6%)	-	13.5% (-17%)

1.2.2 Lattice Compensation

Since the error signal is the weighted difference between the statistics calculated from the denominator and numerator lattices, the quality of the lattice is thus very important. However, even if the beam width used in the lattice generation process is large it is still impossible to cover all possible competing hypotheses. Actually, if indeed all competing hypotheses are included, using lattice to constraint the computation of the denominator lattice can no longer speedup the training process.

The problem often happens when the reference hypothesis is missing from or misaligned with the denominator lattice, under which condition the gradient can be unfairly higher since $\gamma_{mt}^{DEN}(i)$ is 0. This behavior can be frequently seen for silence frames since they are likely to be missing from the denominator lattices but occur very often in the numerator lattices. One of the behaviors introduced by poor lattice quality is the run-away silence frames. The number of silence frames in the decoding results increases as the training epoch increases. This results in increased deletion errors in the decoding result.

There are several ways to fix this problem. The first approach is to remove the frames where the reference hypothesis is not in the denominator lattice when computing the gradients. For silence frames, for example, this can be achieved by counting silence frames as *incorrect* in $A(\mathbf{s}, \mathbf{s}^m)$ in sMBR which effectively sets the error signal of these frames to be zero. A more general approach, referred as frame-rejection [16], is to just remove these frames directly. Another approach, which is believed to perform better, is to augment the lattices with the reference hypothesis [15]. For example, we can add artificial silence arcs to the lattice, one for each start/end node pair connected by a word arc, with an appropriate entering probability and without introducing redundant silence paths.

Figure 1.1, shown in [16], is the result on the SWB 110hr dataset with and without using the frame-rejection technique. From the figure we can observe that, without frame-rejection, the MMI training starts overfitting

after epoch 3. However, when frame rejection is used the training is stable even after epoch 8 and better test accuracy can be achieved.

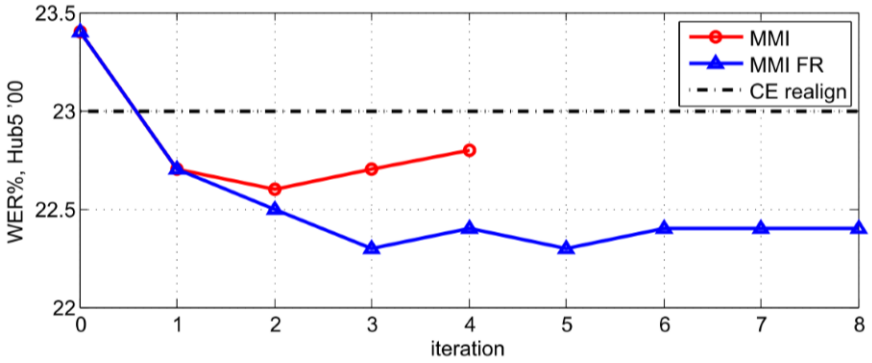

Fig. 1.1 Effect of frame-rejection (FR) measured as WER on Hub5'00 when the SWB 110hr training set is used. (Figure from Vesely et al. [16], permitted to use by ISCA.)

Figure 1.2, based on the results from [15], compares the WER on Hub5'00 using 300hr training set with and without using lattice compensation for silences. In this figure, a relatively large learning rate was used. Without using the silence treatment severe overfitting can be observed even at the first epoch. However, when the silence frames are specially handled we can see great WER reduction at the first epoch and relatively stable results after that.

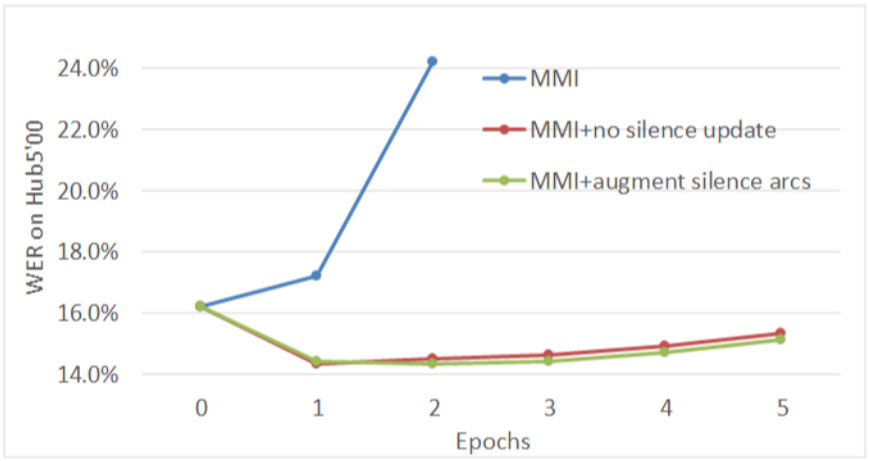

Fig. 1.2 WER on Hub5'00 with and without using silence treatment. (Based on the results from [15])

1.2.3 Frame-Smoothing

Even if the lattices are correctly compensated, we can still observe the quick overfitting phenomena during the training, which can be easily identified by the diverge of the sequence training criterion and the frame accuracy computed from the DNN score alone: The training criterion continues to improve while the frame-accuracy computed from the DNN score alone tanks significantly. While people have hypothesized that the overfitting problem is caused by the sparse lattice (e.g., even the fattest lattices that can be practically generated reference only about 3% of senones [15]), we believe this is not the only reason. The overfitting problem may also attribute to the fact that sequence is in a higher dimensional space than the frames. As such, the posterior distribution estimated from the training set is more likely to be different from that in the testing. This problem can be alleviated by making the sequence-discriminative training criterion closer to the frame-discriminative training criterion, for example, by using weak LM. The problem can be further alleviated using a technique referred as frame-smoothing (F-smoothing) [15], which instead of minimizing the sequence-discriminative training criterion alone, minimizing a weighted sum of the sequence and frame criteria

$$J_{FS-SEQ}\left(\theta;\mathbb{S}\right) = \left(1-H\right) J_{CE}\left(\theta;\mathbb{S}\right) + H J_{SEQ}\left(\theta;\mathbb{S}\right), \tag{1.17}$$

where H is a smoothing factor often set empirically. It has been shown that a frame/sequence ratio of 1:4 (or $H = 4/5$) to 1:10 (or $H = 10/11$) is often effective. F-smoothing not only reduces the possibility of overfitting but also makes the training process less sensitive to the learning rate. F-smoothing is inspired by I-smoothing [12] and similar regularization approaches for adaptation [17]. Note that normal regularization techniques such as L1 and L2 regularization do not help.

Figure 1.3, based on [15], demonstrates the results on SWB Hub5'00 with and without using F-smoothing. With F-smoothing it is much less likely to overfit to the training set. Overall F-smoothing achieves 0.6% absolute or 4% relative WER reduction.

1.2.4 Learning Rate Adjustment

The learning rate used in the sequence-discriminative training should be smaller than that used in the frame cross-entropy training for two reasons. First, the sequence-discriminative training is often started from the CE-trained model which has already been well trained and thus requires smaller updates. Second, the sequence-discriminative training is more prone to overfitting. Using smaller learning rate can control the convergence more effectively. In practice people have found that using a learning rate that is similar

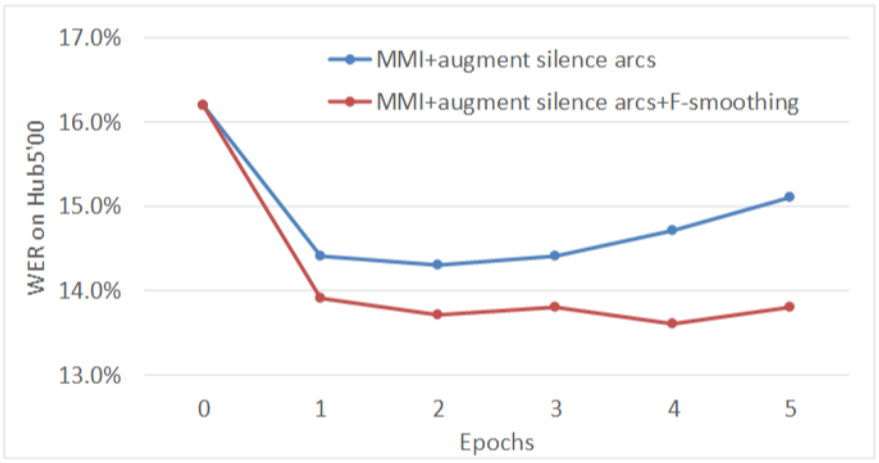

Fig. 1.3 WER on SWB Hub5'00 with and without using F-smoothing in the sequence-discriminative training of DNNs. (Based on results from [15])

to that used in the final stage of CE-training to be effective. For example, Vesely et al. [16] reported that an effective learning rate of $1e^{-4}$ per utterance worked well for both (B)MMI and sMBR, while Su et al. [15] showed that a learning rate of 1/128,000 per frame (or 0.002 per 256 frames) worked well when F-smoothing is used. The requirement to choose a good learning rate may be eliminated if algorithms such as Hessian-free [9] are used.

1.2.5 Training Criterion Selection

There are different observations with regard to the training criterion. Most results seem to suggest that the training criterion is not critical. For example Table 1.2, which is extracted from [16], indicate that across MMI, BMMI, MPE, and sMBR, the WER on the SWB Hub5'00 and Hub5'01 datasets are very close although sMBR slightly outperforms other criteria. Since MMI is best understood and easiest to implement, it is thus suggested to use MMI if you need to implement one from scratch.

1.2.6 Other Considerations

Sequence-discriminative training is more computationally demanding. As such it is much slower. For example, a simple CPU-side implementation may increase runtime 12-fold compared to the frame CE training. Fortunately,

Table 1.2 The effect of different sequence-discriminative training criteria measured as WER on the Hub5'00 and Hub5'01 datasets when the 300hr training set is used. (Summarized from [16])

	Hub5'00 SWB	Hub5'01 SWB
GMM BMMI	18.6%	18.9%
DNN CE	14.2%	14.5%
DNN MMI	12.9%	13.3%
DNN BMMI	12.9%	13.2%
DNN MPE	12.9%	13.2%
DNN sMBR	12.6%	13.0%

with careful engineering, it is possible to achieve significant speedups through parallelized execution on a GPGPU. To speedup the acoustic-score computation each arc can be processed at a separate CUDA thread. The lattice-level forward-backward processing, though, requires special treatment since the computation must be decomposed into sequential, dependency-free CUDA launches. In an example provided by Su et al. [15], there are 106 dependency-free node regions (=launches) for a 7.5-second lattice with 211,846 arcs and 6974 nodes at an average of 1999 arcs (=threads per launch). Moreover, lattice forward/backward and error-signal accumulation require atomic summation of log-probabilities. This can be emulated through CUDA's atomic *compare-and-swap* instruction. To reduce target-operand collisions it has been found to be critical to shuffle operations into a random-like order.

Further speed improvement can be obtained by using a parallel read-ahead thread to preload data and by generating lattices using a cluster of CPU computers. In the runtime experiments conducted by Su et al. [15], it was shown that the overall runtime increases only by about 70% compared to the CE training (when lattice generation is not considered) even on fat lattices of nearly 500 arcs per frame.

1.3 Noise Contrastive Estimation

In the above discussion, we assumed that the conventional minibatch-based SGD algorithm is used for sequence-discriminative training. Although careful engineering can speed up the training (e.g., as in [15]), the possible speed improvement is limited by the nature of the algorithm. In this section, we introduce noise contrastive estimation (NCE), an advanced training algorithm that can potentially further improve the training speed.

NCE was first proposed by Gutmann and Hyvarinen [3, 4] as a more reliable algorithm to estimate unnormalized statistical models. It was later successfully applied to training neural network language models (LMs) [10].

1.3.1 Casting Probability Density Estimation Problem as a Classifier Design Problem

Assume p_d is an unknown probability density function (pdf) and $\mathbf{X} = (\mathbf{x}_1, ..., \mathbf{x}_{T_d})$ of a random vector $\mathbf{x} \in \mathbb{R}^N$ is sampled from p_b, where T_d is the sample size. To estimate p_b we assume it belongs to a parameterized family of functions $p_m(.;\theta)_\theta$, where θ is a vector of parameters. In other words $p_d(.) = p_m(.;\theta^*)$ for some parameter ϑ^*. The parametric density estimation problem can thus be converted into the problem of finding ϑ^* from the observed sample $\mathbf{X}$.

Typically, we require, for any θ,

$$\int p_m(\mathbf{u};\vartheta)d\mathbf{u} = 1, \tag{1.18}$$

$$p_m(\mathbf{u};\vartheta) \geq 0 \qquad \forall \mathbf{u} \tag{1.19}$$

so that $p_m(.;\vartheta)$ is a valid pdf. In this case we say the model is normalized and the maximum likelihood principle can then be used to estimate θ. However, in many cases, we only require that for certain θ (e.g., the true parameter θ^*) the normalization constraint is satisfied. In this case we say that the model is unnormalized. Since we assume p_b belongs to $p_m(.;\theta)_\theta$, we know that the unnormalized model integrates to one at least for parameter ϑ^*.

Following [4], we denote by $p^0_m(.;\alpha)$ the unnormalized model parameterized by α. The unnormalized model $p^0_m(.;\alpha)$ can be converted into a normalized one as $p^0_m(.;\alpha)/Z(\alpha)$, where

$$Z(\alpha) = \int p^0_m(\mathbf{u};\alpha)d\mathbf{u} \tag{1.20}$$

is the partition function and often is expensive to compute when $\mathbf{u}$ is of high dimension. Since for each α there is a corresponding $Z(\alpha)$, we may define a normalizing parameter $c = -\ln Z(\alpha)$ and represent the likelihood of of the normalized model as

$$\ln p_m(.;\vartheta) = \ln p^0_m(.;\alpha) + c \tag{1.21}$$

with parameter $\vartheta = (\alpha, c)$. Note that at ϑ^* we have $\vartheta^* = (\alpha^*, 0)$.

The basic idea of NCE is to convert the density estimation problem to a two-class classification problem by describing properties of the observed sample $\mathbf{X}$ relative to the properties of some reference i.i.d. sample $\mathbf{Y} = (\mathbf{y}_1, ...\mathbf{y}_{\mathrm{T_n}})$ of a random variable $\mathbf{y} \in \mathbb{R}^N$ sampled from pdf p_n, which we can control, where T_n is the sample size. In [4] Gutmann and Hyvarinen proposed to use logistic regression, a popular two-class classifier, to provide the relative description on the form of the ratio p_d/p_n.

Let's construct a unified dataset $\mathbf{U} = \mathbf{X} \cup \mathbf{Y} = (\mathbf{u}_1, ..., \mathbf{u}_{T_d+T_n})$ and assign to each data point $\mathbf{u}_t$ a binary class label

$$C_t = \begin{cases} 1 & \text{if } \mathbf{u}_t \in \mathbf{X} \\ 0 & \text{if } \mathbf{u}_t \in \mathbf{Y} \end{cases} \tag{1.22}$$

Note that the prior probabilities are

$$P(C=1) = \frac{T_d}{T_d + T_n}, \tag{1.23}$$

$$P(C=0) = \frac{T_n}{T_d + T_n} \tag{1.24}$$

and the class-conditional probability densities are

$$p(\mathbf{u}|C=1) = p_m(\mathbf{u};\vartheta), \tag{1.25}$$

$$p(\mathbf{u}|C=0) = p_n(\mathbf{u}) \tag{1.26}$$

The posterior probabilities for the classes are therefore

$$h(\mathbf{u};\vartheta) \triangleq P(C=1|\mathbf{u};\vartheta) = \frac{p_m(\mathbf{u};\vartheta)}{p_m(\mathbf{u};\vartheta) + \nu p_n(\mathbf{u})}, \tag{1.27}$$

$$P(C=0|\mathbf{u};\vartheta) = 1 - h(\mathbf{u};\vartheta) = \frac{\nu p_n(\mathbf{u})}{p_m(\mathbf{u};\vartheta) + \nu p_n(\mathbf{u})}, \tag{1.28}$$

where

$$\nu \triangleq \frac{P(C=0)}{P(C=1)} = T_n/T_d. \tag{1.29}$$

If we further define $G(.;\theta)$ as the log-ratio between $p_m(.;\vartheta)$ and p_n,

$$G(\mathbf{u};\vartheta) \triangleq \ln p_m(\mathbf{u};\vartheta) - \ln p_n(\mathbf{u}), \tag{1.30}$$

$h(\mathbf{u};\vartheta)$ can be written as

$$h(\mathbf{u};\vartheta) = \frac{1}{1 + \nu \frac{p_n(\mathbf{u})}{p_m(\mathbf{u};\vartheta)}} = \sigma_\nu(G(\mathbf{u};\vartheta)), \tag{1.31}$$

where

$$\sigma_\nu(u) = \frac{1}{1 + \nu \exp(-u)} \tag{1.32}$$

is the logistic function parameterized by ν. If we assume the class labels C_t are independent and follow a Bernoulli distribution the conditional log-likelihood (or negative cross-entropy) is

$$\ell(\vartheta) = \sum_{t=1}^{T_d+T_n} C_t \ln P(C_t = 1|\mathbf{u_t};\vartheta) + (1 - C_t) P(C_t = 0|\mathbf{u}_t;\vartheta)$$
$$= \sum_{t=1}^{T_d} \ln [h(\mathbf{x}_t;\vartheta)] + \sum_{t=1}^{T_n} \ln [1 - h(\mathbf{y}_t;\vartheta)] \quad (1.33)$$

By optimizing $\ell(\vartheta)$ with respect to θ we get an estimate of p_d. In other words, the density estimation, which is an unsupervised learning problem, is now converted into a two-class supervised classifier design problem as pointed out firstly by Hastie et al. in [5].

1.3.2 Extension to Unnormalized Models

The above argument was further extended to the unnormalized model by Gutmann and Hyvarinen in [4]. They defined the criterion

$$J_T(\vartheta) = \frac{1}{T_d} \left\{ \sum_{t=1}^{T_d} \ln [h(\mathbf{x}_t;\vartheta)] + \sum_{t=1}^{T_n} \ln [1 - h(\mathbf{y}_t;\vartheta)] \right\}$$
$$= \frac{1}{T_d} \sum_{t=1}^{T_d} \ln [h(\mathbf{x}_t;\vartheta)] + \nu \frac{1}{T_n} \sum_{t=1}^{T_n} \ln [1 - h(\mathbf{y}_t;\vartheta)] \quad (1.34)$$

to find the best ϑ to estimate p_d. It is obvious that improving $J_T(\vartheta)$ means the two-class classifier can more accurately distinguish between the observed data and the reference data.

As T_d increases and by fixting ν, $T_n = \nu T_d$ also increases, $J_T(\vartheta)$ converges in probability to

$$J(\vartheta) = \mathbb{E}\{\ln [h(\mathbf{x}_t;\vartheta)]\} + \nu \mathbb{E}\{\ln [1 - h(\mathbf{y}_t;\vartheta)]\}. \quad (1.35)$$

It is proven [4], by defining $f_m(.) = \ln p_m(.;\vartheta)$ and rewritting the criterion as a function of f_m, that

- $J(\vartheta)$ attains a maximum when $p_m(.;\vartheta) = p_d$. This is the only extrema if the noise density p_n is chosen such that it is nonzero whenever p_d is nonzero. More importantly, maximization is performed without any normalization constraint for $p_m(.;\vartheta)$.
- $\hat{\theta}_T$, the value of θ which (globally) maximizes $J_T(\vartheta)$, converges to θ if the following three conditions are all satisfied: a) p_n is nonzero whenever p_d is nonzero; b) J_T uniformly convergent in probability to J; and c) for large sample sizes the objective function J_T becomes peaked enough around the true value θ^* .

- $\sqrt{T_d}\left(\hat{\vartheta}_T - \theta^*\right)$ is asymptotically normal with mean zero and a bounded covariance matrix Σ.
- For $\nu \to \infty$, Σ is independent of the choice of p_n.

Based on these properties, it is suggested that we should choose noise for which an analytical expression for $\ln p_n$ is available, can be sampled easily, and in some aspect similar to the data. It is also suggested that the noise sample size should be as large as computationally possible. Example noise distributions are Gaussian and uniform distributions.

1.3.3 Apply NCE in DNN Training

In the acoustic model training using the cross-entropy criterion, we estimate the distribution $P(s|\mathbf{o};\theta)$ of the senone s given the observation $\mathbf{o}$. For each observation $\mathbf{o}$ with label s we generate ν noise labels $y_1, \cdots, y_\nu$ and optimize

$$J_T(\mathbf{o},\vartheta) = \ln\left[h(s|\mathbf{o};\vartheta)\right] + \sum_{t=1}^{\nu} \ln\left[1 - h(y_t|\mathbf{o};\vartheta)\right]. \tag{1.36}$$

Since

$$\begin{aligned}
\frac{\partial}{\partial\theta}\ln\left[h(s|\mathbf{o};\vartheta)\right] &= \frac{h(s|\mathbf{o};\vartheta)\left[1 - h(s|\mathbf{o};\vartheta)\right]}{h(s|\mathbf{o};\vartheta)}\frac{\partial}{\partial\vartheta}\ln P_m(s|\mathbf{o};\vartheta) \\
&= \left[1 - h(s|\mathbf{o};\vartheta)\right]\frac{\partial}{\partial\vartheta}\ln P_m(s|\mathbf{o};\vartheta) \\
&= \frac{\nu P_n(s|\mathbf{o})}{P_m(s|\mathbf{o};\vartheta) + \nu P_n(s|\mathbf{o})}\frac{\partial}{\partial\vartheta}\ln P_m(s|\mathbf{o};\vartheta)
\end{aligned} \tag{1.37}$$

and

$$\begin{aligned}
\frac{\partial}{\partial\theta}\ln\left[1 - h(y_t|\mathbf{o};\vartheta)\right] &= -\frac{h(y_t|\mathbf{o};\vartheta)\left[1 - h(y_t|\mathbf{o};\vartheta)\right]}{1 - h(y_t|\mathbf{o};\vartheta)}\frac{\partial}{\partial\vartheta}\ln P_m(y_t|\mathbf{o};\vartheta) \\
&= -h(y_t|\mathbf{o};\vartheta)\frac{\partial}{\partial\vartheta}\ln P_m(y_t|\mathbf{o};\vartheta) \\
&= -\frac{P_m(y_t|\mathbf{o};\vartheta)}{P_m(y_t|\mathbf{o};\vartheta) + \nu P_n(y_t|\mathbf{o})}\frac{\partial}{\partial\vartheta}\ln P_m(y_t|\mathbf{o};\vartheta)
\end{aligned} \tag{1.38}$$

we get

$$\frac{\partial}{\partial\theta}J_T\left(\mathbf{o},\vartheta\right)=\frac{\nu P_n\left(s|\mathbf{o}\right)}{P_m\left(s|\mathbf{o};\vartheta\right)+\nu P_n\left(s|\mathbf{o}\right)}\frac{\partial}{\partial\vartheta}\ln P_m\left(s|\mathbf{o};\vartheta\right)$$
$$-\sum_{t=1}^{\nu}\left[\frac{P_m\left(y_t|\mathbf{o};\vartheta\right)}{P_m\left(y_t|\mathbf{o};\vartheta\right)+\nu P_n\left(y_t|\mathbf{o}\right)}\frac{\partial}{\partial\vartheta}\ln P_m\left(y_t|\mathbf{o};\vartheta\right)\right]. \quad (1.39)$$

Note that the weights $\frac{P_m(y_t|\mathbf{o};\vartheta)}{P_m(y_t|\mathbf{o};\vartheta)+\nu P_n(y_t|\mathbf{o})}$ are always between 0 and 1 and so the NCE learning is very stable. In addition, since $P_m\left(s|\mathbf{o};\vartheta\right)$ is an unnormalized model, the gradient $\frac{\partial}{\partial\vartheta}\ln P_m\left(.|\mathbf{o};\vartheta\right)$ can be computed efficiently. However, in the unnormalized model there is a normalization factor c for each observation $\mathbf{o}$ which can become a problem when the AM training set is very large. Fortunately, experiments have shown that there is no or little performance degradation even if the same c is used for all observations or even when c is always set to 0 [4, 10]. Since it typically does not increase the computation a lot and often helps to boost the estimation accuracy using a shared c is recommended.

Since the conditional probability distributions for different observations are estimated with the same DNN, we cannot learn these distributions independently. Instead, we define a global NCE objective

$$J_T^G\left(\vartheta\right)=\sum_{t=1}^{T_d}J_T\left(\mathbf{o}_\mathrm{t},\vartheta\right). \quad (1.40)$$

The above derivation can be easily extended to sequence-discriminative training. The only difference is that in the sequence-discriminative training there are significantly more classes than that in the frame-level training since each label sequence is considered as a different class. More specifically, for the m-th utterance the distribution we need to estimate is

$$\log P\left(\mathbf{w}^m|\mathbf{o}^m;\theta\right)=\log p\left(\mathbf{o}^m|\mathbf{s}^m;\theta\right)^\kappa P\left(\mathbf{w}^m\right)+c^m. \quad (1.41)$$

Recall that here $\mathbf{o}^m=\mathbf{o}_1^m,\cdots,\mathbf{o}_t^m,\cdots,\mathbf{o}_{T_m}^m$ and $\mathbf{w}^m=\mathbf{w}_1^m,\cdots,\mathbf{w}_t^m,\cdots,\mathbf{w}_{N_m}^m$ are the observation sequence and the correct word transcription of the m-th utterance, respectively, $\mathbf{s}^m=s_1^m,\cdots,s_t^m,\cdots,s_{T_m}^m$ is the sequence of states corresponding to $\mathbf{w}^m$, κ is the acoustic scaling factor, T_m is the total number of frames in utterance m, and N_m is the total number of words in the transcription of the same utterance. In the sequence-discriminative training, we can use uniform distributions over all possible state sequences or over sequences in the lattice as the noise distribution.

References

1. Bahl, L., Brown, P., De Souza, P., Mercer, R.: Maximum mutual information estimation of hidden markov model parameters for speech recognition. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 11, pp. 49–52 (1986)
2. Goel, V., Byrne, W.J.: Minimum Bayes-risk automatic speech recognition. Computer Speech and Language **14**(2), 115–135 (2000)
3. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: International Conference on Artificial Intelligence and Statistics, pp. 297–304 (2010)
4. Gutmann, M.U., Hyvärinen, A.: Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. The Journal of Machine Learning Research **13**, 307–361 (2012)
5. Hastie, T., Tibshirani, R., Friedman, J., Hastie, T., Friedman, J., Tibshirani, R.: The elements of statistical learning, vol. 2. Springer (2009)
6. Hennebert, J., Ris, C., Bourlard, H., Renals, S., Morgan, N.: Estimation of global posteriors and forward-backward training of hybrid hmm/ann systems. (1997)
7. Kapadia, S., Valtchev, V., Young, S.: MMI training for continuous phoneme recognition on the TIMIT database. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 2, pp. 491–494 (1993)
8. Kingsbury, B.: Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 3761–3764 (2009)
9. Kingsbury, B., Sainath, T.N., Soltau, H.: Scalable minimum bayes risk training of deep neural network acoustic models using distributed hessian-free optimization. In: Proc. Annual Conference of International Speech Communication Association (INTERSPEECH) (2012)
10. Mnih, A., Teh, Y.W.: A fast and simple algorithm for training neural probabilistic language models. arXiv preprint arXiv:1206.6426 (2012)
11. rahman Mohamed, A., Yu, D., Deng, L.: Investigation of full-sequence training of deep belief networks for speech recognition. In: Proc. Annual Conference of International Speech Communication Association (INTERSPEECH), pp. 2846–2849 (2010)
12. Povey, D.: Discriminative training for large vocabulary speech recognition. Ph.D. thesis, Cambridge University Engineering Dept (2003)
13. Povey, D., Kanevsky, D., Kingsbury, B., Ramabhadran, B., Saon, G., Visweswariah, K.: Boosted MMI for model and feature-space discriminative training. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4057–4060 (2008)

14. Povey, D., Woodland, P.C.: Minimum phone error and I-smoothing for improved discriminative training. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), vol. 1, pp. I–105 (2002)
15. Su, H., Li, G., Yu, D., Seide, F.: Error back propagation for sequence training of context-dependent deep networks for conversational speech transcription. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP) (2013)
16. Veselỳ, K., Ghoshal, A., Burget, L., Povey, D.: Sequence-discriminative training of deep neural networks. In: Proc. Annual Conference of International Speech Communication Association (INTERSPEECH) (2013)
17. Yu, D., Yao, K., Su, H., Li, G., Seide, F.: Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition. In: Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 7893–7897 (2013)