Maximum a Posteriori Adaptation of Network Parameters in Deep Models

Zhen Huang[1], *Sabato Marco Siniscalchi*[1,2], *I-Fan Chen*[1], *Jinyu Li*[3], *Jiadong Wu*[1], *and Chin-Hui Lee*[1]

[1]School of ECE, Georgia Institute of Technology, Atlanta, GA. USA
[2]Department of Computer Engineering, Kore University of Enna, Enna, Italy
[3] Microsoft Corporation, One Microsoft Way, Redmond, WA. USA

huangzhenee@gatech.edu, marco.siniscalchi@unikore.it, ichen8@gatech.edu, jinyli@exchange.microsoft.com, jwu65@gatech.edu, chl@ece.gatech.edu

Abstract

We present a Bayesian approach to adapting parameters of a well-trained context-dependent, deep-neural-network, hidden Markov model (CD-DNN-HMM) to improve automatic speech recognition performance. Given an abundance of DNN parameters but with only a limited amount of data, the effectiveness of the adapted DNN model can often be compromised. We formulate *maximum a posteriori* (MAP) adaptation of parameters of a specially designed CD-DNN-HMM with an augmented linear hidden networks connected to the output tied states, or senones, and compare it to feature space MAP linear regression previously proposed. Experimental evidences on the 20,000-word open vocabulary Wall Street Journal task demonstrate the feasibility of the proposed framework. In supervised adaptation, the proposed MAP adaptation approach provides more than 10% relative error reduction and consistently outperforms the conventional transformation based methods. Furthermore, we present an initial attempt to generate hierarchical priors to improve adaptation efficiency and effectiveness with limited adaptation data by exploiting similarities among senones.

Index Terms: deep neural networks, hidden Markov model, Bayesian adaptation, automatic speech recognition.

1. Introduction

Despite the recent outstanding results demonstrated by context-dependent, deep-neural-network based hidden Markov models (CD-DNN-HMMs) in various automatic speech recognition (ASR) tasks and data sets [1, 2, 3], these acoustic models, similarly to conventional context-dependent, Gaussian-mixture-model based HMMs (CD-GMM-HMMs) [4], still suffer from a performance degradation under potential mismatched conditions between training and testing conditions. For standard hybrid system using artificial neural networks (ANNs) and HMMs [5] in which CD-DNN-HMM is a special case, there exist many adaptation techniques. The simplest approach modifies all weights of the connectionist architecture using some adaptation materials. Unfortunately, it leads to over-fitting on the adaptation material when the amount of adaptation patterns is limited [6]. Recent approaches, such as regularization based [7, 8], subspace based [9, 10], transformation based [6, 11, 12, 13], i-Vector based [14], native neural network based [15, 16, 17], factorization based [18] and fast adaptation schemes based on discriminant speaker codes [19, 16, 20, 21], have been proposed to circumvent the problem.

Transformation based methods are the most popular connectionist adaptation techniques. The key idea is to augment the structure of the ANN component by adding an affine transformation network to the input [6], hidden [11], or output layer [22]. They are typically trained while keeping the rest of the network parameters fixed. Motivations for these approaches stem from the concept that only relatively few parameters could be learned during adaptation and therefore it is preferable to training the entire network when the adaptation set is limited. For linear hidden network (LHN) layer approaches, the last hidden layer is usually designed to be a bottleneck to ensure an affordable parameter size [23, 24, 25, 26].

However, adapting parameters in a CD-DNN-HMM is much more challenging than earlier connectionist adaptation schemes because of its huge parameter set size with a large number of network branches connected to a large set of tied HMM states, often referred to as senones [27]. Furthermore, DNN parameters are adapted by every sample frame regardless of its senone class. Therefore, the posterior probabilities for the unobserved and scarcely seen senones are often pushed towards zero during adaptation. Such a phenomenon is commonly referred to as *catastrophic forgetting* [28]. Conservative ad-hoc solutions for ANNs have been proposed to force the senone distribution estimated from the adapted model to be close to that of the unadapted model. For example, a Kullback-Leibler divergence (KLD) based objective criterion to be used during adaptation was devised in [8] in order to alleviate the catastrophic forgetting problem. A variation to the standard method of assigning the target values was instead discussed in [11]. Nonetheless, Bayesian solutions adopted in the CD-GMM-HMMs to address the same issue [29] have not been fully exploited.

In this study, we attempt to cast DNN adaptation within a Bayesian framework in the spirit of maximum a posteriori (MAP) adaptation [30]. The key goal is to re-estimate some DNN parameters by representing available information in an augmented linear hidden network (LHN) added after the last non-linear hidden layer. Experimental results on the 20,000-word open vocabulary Wall Street Journal task demonstrates the feasibility of the proposed approach. Under supervised adaptation, the proposed MAP adaptation scheme can provide a relative word error rate (WER) reduction of more than 10% from an already-strong speaker independent CD-DNN-HMM baseline and consistently outperform conventional transformation based adaptation schemes. It also compares favorably against the feature space maximum a posteriori linear regression approach to speaker adaptation proposed in [31]. We also present an initial attempt to generate hierarchical priors for improving adaptation efficiency with small amounts of adaptation data by exploiting the similarities among senones.

2. Training of Deep Models

In DNNs, hidden layers are usually constructed by sigmoid units, and the output layer is a softmax layer. The values of the nodes can therefore be expressed as:

$$\mathbf{x}_i = \begin{cases} \mathbf{W}_1 \mathbf{o}^t + \mathbf{b}_1, & i = 1 \\ \mathbf{W}_i \mathbf{y}_{i-1} + \mathbf{b}_i, & i > 1 \end{cases}, \quad (1)$$

$$\mathbf{y}_i = \begin{cases} \mathrm{sigmoid}(\mathbf{x}_i), & i < L \\ \mathrm{softmax}(\mathbf{x}_i), & i = L \end{cases}, \quad (2)$$

where $\mathbf{W}_1$, and $\mathbf{W}_i$ are the weight matrices, $\mathbf{b_1}$, and $\mathbf{b_i}$ are the bias vectors, $\mathbf{o}^t$ is the input frame at time t, L is the total number of the hidden layers, and both sigmoid and softmax functions are element-wise operations. The vector $\mathbf{x}_i$ corresponds to pre-nonlinearity activations, and $\mathbf{y}_i$ and $\mathbf{y}_L$ are the vectors of neuron outputs at the i^{th} hidden layer and the output layer, respectively. The softmax outputs were considered as an estimate of the senone posterior probability:

$$p(C_j|\mathbf{o}^t) = \mathbf{y}_L^t(j) = \frac{\exp(\mathbf{x}_L^t(j))}{\sum\limits_i \exp(\mathbf{x}_L^t(i))}, \quad (3)$$

where C_j represents the j^{th} senone and $\mathbf{y}_L(j)$ is the j^{th} element of $\mathbf{y}_L$.

The DNN is trained by maximizing the log posterior probability over the training frames. This is equivalent to minimizing the cross-entropy objective function. Let $\mathcal{X}$ be the whole training set, which contains T frames, *i.e.* $\mathbf{o}^{1:T} \in \mathcal{X}$, then the loss with respect to $\mathcal{X}$ is given by

$$\mathcal{L}^{1:T} = -\sum_{t=1}^{T} \sum_{j=1}^{J} \tilde{\mathbf{p}}^t(j) \log p(C_j|\mathbf{o}^t), \quad (4)$$

where $p(C_j|\mathbf{o}^t)$ is defined in Eq. (3); $\tilde{\mathbf{p}}^t$ is the target probability of frame t. In real practices of DNN systems, the target probability $\tilde{\mathbf{p}}^t$ is often obtained by a forced alignment with an existing system resulting in only the target entry that is equal to 1. Mini-batch stochastic gradient descent (SGD) [32], with a reasonable size of mini-batches to make all matrices fit into the GPU memory, was used to update all neural parameters during training. Pre-training methods was used for the initialisation of the DNN parameters [33].

3. Transformation Based Adaptation for Deep Models

For DNN adaptation, some researchers choose to add an affine transformation network between the last hidden layer and the output layer weights matrix, i.e., an LHN, and adapt only the LHN parameters while keeping fixed all of the other DNN parameters [11]. In order to reduce the amount of parameters to adapt, usually the last hidden layer is designed to be a bottleneck (less neurons) [23, 24, 25, 26]. Superior results were obtained by this kind of LHN formulation than other transformation based adaptation schemes, such as linear input network (LIN) and linear output network (LON).

The LIN approach performs adaptation by adding an augmented linear input layer and only adapts this set of LIN parameters. If we follow the common idea that the hidden layers of a DNN is actually learning a more suitable data representation and extracting better "feature" for the output layer that is serving as a log-linear model, then by transforming the raw input

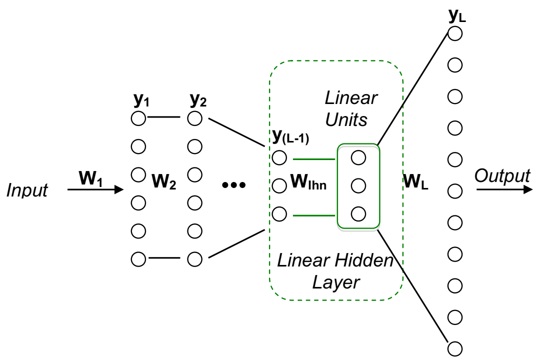

Figure 1: Basic neural architecture for adapting the HMM/ANN parameters: weights associated with the links in the dashed rectangles are estimated while all other weights remain unchanged. The activation function of each LHN units is a linear function.

using the LIN layer, we might harm the ability of data representation of the hidden layers. On the other hand for the LON approach, the issue is that usually we can't reduce the number of neurons of the output layer because we want to directly model the senones (the number of senones can be more than 10000 in practice), and that means we have to add a huge augmented layer with even more parameters to be adapted.

If we deem the hidden layers as a feature extractor and the output layer as the discriminative model. The model parameters are the weights of the output layer's affine transform matrix, $\mathbf{W}_L$. The output $\mathbf{y}_L$ can now be expressed as:

$$\mathbf{y}_L = \mathrm{softmax}(\mathbf{W}_L \mathbf{y}_{L-1}), \quad (5)$$

where the activation at the last hidden layer, $\mathbf{y}_{L-1}$, can be used as the new feature representation extracted by the hidden layers. When adding an augmented LHN after the last hidden layer, it is equivalent to applying a transformation matrix $\mathbf{W}_{lhn}$ to the model parameters to obtain an adapted model parameter set:

$$\mathbf{y}_L = \mathrm{softmax}(\mathbf{W}_{lhn} \mathbf{W}_L \mathbf{y}_{L-1}), \quad (6)$$

An LHN adaptation structure is shown in Figure 1. This formulation is quite similar to maximum likelihood linear regression (MLLR) [34]. The difference is that in MLLR the model parameters are Gaussian mean and variance while here the model parameters are the log-linear model's transformation matrix weights.

4. MAP Adaptation for Deep Models

Although conventional DNN adaptation approaches try to alleviate over-fitting issues by reducing the number of parameters to be adapted, such number could still be very big in some cases. Inspired by the MAP adaptation that address the problem effectively in GMM-HMM systems, in this section, we explain how to apply the MAP approach to the LHN adaptation. Note that though we choose LHN for demonstration, the proposed MAP approach can be easily applied to other DNN adaptation frameworks like [8, 14, 16, 17] as well.

4.1. Prior Estimation

In order to establish a MAP adaptation framework like in [30], a prior distribution over the weights of the affine transformation network need to be imposed. To analyze and estimate the prior density, we utilized the training data of the baseline DNN. We adopted an empirical Bayes approach [30, 29] and treated each speaker in the training set as a sample speaker and supervised LHN adaptation was performed. After that, we can get a particular LHN for each speaker. We observed that the histograms for weights of the adapted LHN over speakers are quite like Gaussian, so we assume that the distribution of the weights in $\mathbf{W}_{lhn}$ to be joint Gaussian [31]. By expressing the weights in the LHN transformation matrix $\mathbf{W}_{lhn}$ as a vector $\mathbf{w}$ with each entry representing a particular weight, we have the prior density in the following form:

$$p(\mathbf{W}_{lhn}) = \frac{1}{(2\pi)^{M/2}|\boldsymbol{\Sigma}|^{1/2}} \exp(-\frac{1}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu})) \tag{7}$$

where only the diagonal entries of the covariance matrix Σ are non-zero (from the independence assumption of the weights).

With N adapted speaker weight vectors, the maximum likelihood estimation of the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\Sigma}$ can be expressed as:

$$\boldsymbol{\mu}_{ML} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{w}_i \tag{8}$$

$$\boldsymbol{\Sigma}_{ML} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{w}_i - \boldsymbol{\mu}_{ML})(\mathbf{w}_i - \boldsymbol{\mu}_{ML})^T \tag{9}$$

where $\mathbf{w}_i$ is the vector consisting of the adapted transformation weights of speaker i.

4.2. MAP Formulation

Formal MAP adaptation is conducted following [31]. Eq. (10) formulates the MAP learning idea by adding the term of prior density $p(\mathbf{W}_{lhn})$ to the plain cross entropy objective function.

$$\mathcal{L}_{MAP}^{1:T} = -\lambda \log p(\mathbf{W}_{lhn}) + \mathcal{L}_{xent}^{1:T} \tag{10}$$

Applying the prior form of Eq. (7), the objective function for MAP LHN adaptation is in the form of Eq. (11).

$$\mathcal{L}_{MAP}^{1:T} = \frac{\lambda}{2}(\mathbf{w}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{w}-\boldsymbol{\mu}) + \mathcal{L}_{xent}^{1:T} \tag{11}$$

where only the diagonal entries of the covariance matrix $\boldsymbol{\Sigma}$ are non-zero (from the independence assumption of the weights).

A close look at Eq. (11), when the prior density is a standard Gaussian $\mathcal{N}(\mathbf{0},\mathbf{I})$, MAP learning will degenerate to conventional L2-regularized training. The gradient of $\mathcal{L}_{1:N}^{MAP}$ with respect to $\mathbf{w}$ can now be expressed as:

$$\frac{\partial \mathcal{L}_{MAP}^{1:T}}{\partial \mathbf{w}} = \lambda(\mathbf{w}-\boldsymbol{\mu})^T diag(\boldsymbol{\Sigma}^{-1}) + \frac{\partial \mathcal{L}_{xent}^{1:T}}{\partial \mathbf{w}}, \tag{12}$$

where $diag(\boldsymbol{\Sigma}^{-1})$ consists of the diagonal entries of $\boldsymbol{\Sigma}^{-1}$.

5. Experiments

5.1. Experimental Setup

This study is concerned with the problem of *speaker adaptation*, and experiments are reported on the 20k-word open vocabulary Wall Street Journal task [35] using the Kaldi toolkit [36]. The baseline CD-DNN-HMM system was trained using the WSJ0 material (SI-84). The standard adaptation set of WSJ0 (si_et_ad, 8 speakers, 40 sentences per speaker) was used to perform adaptation of the affine transformation added to the speaker-independent DNN. The standard open vocabulary 20,000-word (20K) read NVP Senneheiser microphone (si_et_20, 8 speakers x 40 sentences) data were used for evaluation. A standard tri-gram language model was adopted during decoding. The ASR performance was given in terms of the word error rate (WER).

The DNN has six hidden layers. The first five hidden layers have 2048 units, whereas the last hidden layer has 216 units. The output layer has 2022 softmax units. This DNN architecture follows conventional configurations used in the speech community except for the last hidden layer, which acts as a bottleneck layer. This configuration was chosen, because a too large dimension of the last non-linear hidden layer might have been harmful for LHN adaptation. The bottleneck based low rank methods have been widely used to achieve more compact DNN models with equivalent performance [23, 24, 25, 26]. The number units equal to 216 was chosen to simulate a sort of three-state phone layer thereby obtaining a kind of hierarchical structure between mono-phones in the hidden layer and senones at the output layer. The input feature vector is a 23-dimension mean-normalized log-filter bank feature with up to second-order derivatives and a context window of 11 frames, forming a vector of 759-dimension (69 $\times$ 11) input. The DNN was trained with an initial learning rate of 0.008 using the cross-entropy objective function. It was initialised with the stacked restricted Boltzmann machines by using layer by layer generative pre-training.

5.2. Experimental Results

The word error rate (WER) attained with different adaptation techniques are reported in Table 1. All available adaptation material was used for performing adaptation, namely 40 sentences per speaker. The term BASELINE refers to the speaker independent CD-DNN-HMM system. LIN, LIN-KLD, and MAP LIN refer to the adaptation technique based on the standard linear input network approach, the KLD regularisation technique[1] in combination with LIN, and the maximum a posteriori transformation based adaptation when a prior is defined over the LIN parameters [31], respectively. The terms LON and LON-KLD are used to denote, with a little abuse of terminology, the direct adaptation of the output layer weights matrix with or without KLD, respectively. LHN adaptation results are also reported along with the corresponding MAP version, MAP LHN, which is the adaptation approach proposed in this paper. Since LHN was inserted between the last hidden layer and the output layer weights matrix, its dimension is 216 $\times$ 216. LHN is initialised to an identity matrix with zero bias, which gives a starting point equivalent to the unadapted model. Supervised adaptation is then performed updating only the LHN parameters. MAP was performed as described in Section 4. For the sake of comparison, LHN-KLD, which denotes standard LHN combined with KLD, was also evaluated.

Indeed LIN and LHN outperforms LON, which attains the worst performance improvement. KLD always improves over affine transformation based adaptation techniques, as expected. Furthermore, the proposed MAP LHN outperforms all other techniques in the given task, and it attains the best recognition results with a relative improvement of 10.4% over the BASELINE. Finally, we would like to remark that MAP LHN compares favourably against MAP LIN, and that confirms that the

[1]The best KLD results obtained in our laboratories are reported.

Table 1: Comparing WERs on the 20K word open vocabulary WSJ0 task for several adaptation approaches using all 40 available adaptation utterances.

System	WER (in %)
BASELINE	8.84%
LIN	8.22%
LIN-KLD	8.06%
MAP LIN	8.13%
LON	8.80%
LON-KLD	8.64%
LHN	8.22%
LHN-KLD	8.15%
MAP LHN	**7.92**%

Table 2: Comparing LHN and MAP LHN performance with different amounts of adaptation data.

# Adaptation Sentences	Standard LHN	MAP LHN
BASELINE	8.84%	
5	8.59%	8.54%
10	8.52%	8.52%
20	8.31%	8.12%
40	8.22%	7.92%

introduction of the bottleneck layer was the key for a proper deployment of MAP LHN.

Table 2 shows experimental results for LHN, and MAP LHN with different amounts of adaptation sentences, namely $\{5, 10, 20, 40\}$, in the second and third columns, respectively. These results confirm that MAP LHN adaptation almost always improves over standard LHN, with the best adaptation results at a WER of 7.92% using 40 utterances. But in very limited adaptation data cases, namely, $\{5, 10\}$, there is only slight or even no improvement by only using flat prior in the MAP adaptation, so we turned to our preliminary investigation of hierarchical priors for dealing with the data scarcity problem.

5.3. Hierarchical Priors: Preliminary Experiments

Hierarchical structures, such as trees, have long been used in the speech community to address the over-fitting issues during model parameters estimation. For instance, efficient adaptation with a limited amount of adaptation data was obtained through the use of a tree data structure to cluster model parameters of a CD-GMM-HMM system in [37]. Similar ideas have been recently explored in DNN learning for enhancing classification performance for classes with few examples in [38], where hierarchical priors where devised for the output layer weights matrix (top-level weights in a DNN) using a tree data structure either fixed or learnable during training.

Top-level DNN weights in a hybrid acoustic model can be regarded as senone embeddings [39], and hierarchical priors can be defined by organising those embedding in a tree data structure. Let $\mathbf{W}_{(D+1)\times L}$ denote the output layer weights matrix (including the bias terms). Each line in $\mathbf{W}_{(D+1)\times L}$ corresponds to a senone embedding. Specifically, the sth senone embedding can be denoted as $\mathbf{w}_s$, which is the sth row in $\mathbf{W}_{(D+1)\times L}$. The tree structure used to generate hierarchical priors can be either learnt during training or given. Here, we used a fixed two-layer tree shown in Figure 2: there are L leaf nodes, with each leaf corresponding to a senone embedding, and S parent nodes clustering together similar leaf nodes. Each parent node clusters senone embeddings sharing the same central phone-state; therefore, S is equal to 130 in this work. Hierarchical priors can now be established by associating a vector $\mathbf{w}_s$ to a leaf node, and a vector $\boldsymbol{\theta}_s$ to each parent node, and imposing a Gaussian probability density distribution over these two vectors as follows: $\boldsymbol{\theta}_s \sim \mathcal{N}(\mathbf{0}, \frac{1}{\sigma_1^2}\mathbf{I}_{(D+1)})$, and $\mathbf{w}_s \sim \mathcal{N}(\boldsymbol{\theta}_s, \frac{1}{\sigma_2^2}\mathbf{I}_{(D+1)})$.

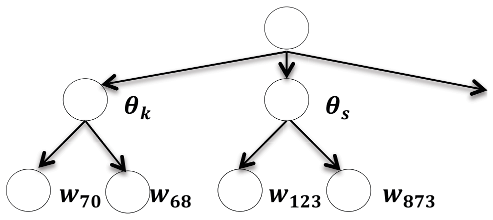

Figure 2: Fixed two-layer tree for hierarchical priors generation. Each leaf node represents a senone embedding, and it is thereby a row of $\mathbf{W}_{(D+1)\times L}$.

Table 3: WER comparisons of flat and hierarchical priors for MAP LHN with a small amount of adaptation utterances.

# Adaptation Sentences	MAP LHN Flat Priors	MAP LHN Hierarchical Priors
5	8.54%	8.48%
10	8.52%	8.45%

The objective function with hierarchical priors is in the form of Eq. (13).

$$\mathcal{L}_{1:N}^{MAP} = \mathcal{L}_{1:N}^{xent} + \frac{\lambda_2}{2}\sum \|\mathbf{w}_s - \boldsymbol{\theta}_s\|^2 + \frac{\lambda_1}{2}\|\boldsymbol{\theta}\|^2. \quad (13)$$

It is can be verified that $\boldsymbol{\theta}_s$ is a scaled average of all $\mathbf{w}_s$ associated to the sth leaf node by minimizing Eq. 13 over $\boldsymbol{\theta}_s$ with fixed DNN weights (see [38]). We focus our attention on experimental results with very small adaptation data amounts, as shown in Table 3. With limited adaptation data, namely 5, 10 utterances, small performance improvements are observed against using flat priors when adaptation is carried out with hierarchical priors. Although the current improvement is still quite small, we believe more sophisticated trees can be adopted for better performance in future studies.

6. Conclusion

We have investigated a *maximum a posteriori* (MAP) adaptation approach for linear hidden networks. The key idea is to treat the parameters of the augmented affine transformation as random Gaussian variables and incorporate prior information obtained from the training data. Speaker adaptation results show that the proposed MAP approaches can lead to a consistent performance improvement over conventional LHN adaptation. Furthermore, MAP LHN outperforms other regularisation schemes.

A first attempt to use hierarchical-based priors with a fixed two-layer tree structure was also studied, and small improvements were observed in a set of preliminary ASR experiments using a limited amount of adaptation sentences. Better results might still be hindered by the current fixed tree hierarchy structure employed in this preliminary work. Indeed, it was demonstrated that learning the tree hierarchy during training improves the classification performance [38]. Finally, from the objective function perspective, we are still relying on cross-entropy. Other forms of frame-level and sequence-level discriminative objectives [3, 40] can also be applied.

7. References

[1] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. Mohamed, “Making deep belief networks effective for large vocabulary continuous speech recognition,” in *Proc. ASRU*, 2011, pp. 30–35.

[2] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition,” *IEEE Trans. Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[3] K. Veselỳ, A. Ghoshal, L. Burget, and D. Povey, “Sequence-discriminative training of deep neural networks,” in *Proc. INTERSPEECH*, 2013, pp. 2345–2349.

[4] L. Rabiner, “A tutorial on hidden Markov models and selected applications in speech recognition,” *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[5] H. Bourlard and N. Morgan, *Connectionist speech recognition: A hybrid approach*. Kluwer Academic Publishers, 1994.

[6] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, and T. Robinson, “Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system,” in *Proc. Eurospeech*, 1995.

[7] X. Li and J. Bilmes, “Regularized adaptation of discriminative classifiers,” in *Proc. ICASSP*, vol. 1, 2006, pp. I–I.

[8] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, “KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition,” in *Proc. ICASSP*, 2013, pp. 7893–7897.

[9] D. Yu, X. Chen, and L. Deng, “Factorized deep neural networks for adaptive speech recognition,” in *Proc. Int. Workshop on Statistical Machine Learning for Speech Processing*, 2012.

[10] D. Yu, L. Deng, and S. Seide, “The deep tensor neural network with applications to large vocabulary speech recognition,” *IEEE Trans. Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 388–396, 2013.

[11] R. Gemello, F. Mana, S. Scanzio, P. Laface, and R. D. Mori, “Linear hidden transformations for adaptation of hybrid ANN/HMM models,” *Speech Communication*, vol. 49, no. 10, pp. 827–835, 2007.

[12] F. Seide, G. Li, X. Chen, and D. Yu, “Feature engineering in context-dependent deep neural networks for conversational speech transcription,” in *Proc. ASRU*, 2011, pp. 24–29.

[13] K. Yao, D. Yu, F. Seide, H. Su, L. Deng, and Y. Gong, “Adaptation of context-dependent deep neural networks for automatic speech recognition,” in *Proc. Spoken Language Technology Workshop*, 2012, pp. 366–369.

[14] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, “Speaker adaptation of neural network acoustic models using i-vectors,” in *Proc. ASRU*, 2013, pp. 55–59.

[15] S. M. Siniscalchi, J. Li, and C.-H. Lee, “Hermitian polynomial for speaker adaptation of connectionist speech recognition systems,” *IEEE Trans. Audio, Speech, and Language Processing*, vol. 21, no. 10, pp. 2152–2161, 2013.

[16] O. Abdel-Hamid and H. Jiang, “Rapid and effective speaker adaptation of convolutional neural network based models for speech recognition,” in *Proc. INTERSPEECH*, 2013, pp. 1248–1252.

[17] P. Swietojanski and S. Renals, “Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models,” in *Proc. IEEE STL*, 2014.

[18] J. Li, J.-T. Huang, and Y. Gong, “Factorized adaptation for deep neural network,” in *Proc. ICASSP*, 2014.

[19] O. Abdel-Hamid and H. Jiang, “Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code,” in *Proc. ICASSP*, 2013, pp. 7942–7946.

[20] S. Xue, O. Abdel-Hamid, H. Jiang, and L. Dai, “Direct adaptation of hybrid DNN/HMM model for fast speaker adaptation in LVCSR based on speaker code,” in *Proc. ICASSP*, 2014, pp. 6339–6343.

[21] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, “Fast adaptation of deep neural network based on discriminant codes for speech recognition,” *IEEE/ACM Trans. on Audio, Speech and Lang. Proc.*, vol. 22, no. 12, pp. 1713–1725, 2014.

[22] B. Li and K. C. Sim, “Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems,” in *Proc. INTERSPEECH*, 2010, pp. 526–529.

[23] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, “Low-rank matrix factorization for deep neural network training with high-dimensional output targets,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6655–6659.

[24] J. Xue, J. Li, and Y. Gong, “Restructuring of deep neural network acoustic models with singular value decomposition.” in *INTERSPEECH*, 2013, pp. 2365–2369.

[25] J. Xue, J. Li, D. Yu, M. Seltzer, and Y. Gong, “Singular value decomposition based low-footprint speaker adaptation and personalization for deep neural network,” in *Proc. ICASSP*, 2014.

[26] S. Xue, H. Jiang, and L. Dai, “Speaker adaptation of hybrid NN/HMM model for speech recognition based on singular value decomposition,” in *Proc. ISCSLP*, 2014.

[27] M.-Y. M.-Y. Hwang and X. Huang, “Shared-distribution hidden markov models for speech recognition,” *IEEE trans. Speech and Audio Processing*, vol. 1, no. 4, pp. 414–420, 1993.

[28] M. Franch, “Catastrophic forgetting in connectionist networks: causes, consequences and solutions,” *Trends in Cognitive Sciences*, vol. 3, no. 4, 1994.

[29] C.-H. Lee and Q. Huo, “On adaptive decision rules and decision parameter adaptation for automatic speech recognition,” *Proc. IEEE*, vol. 88, no. 8, 2000.

[30] J. Gauvain and C.-H. Lee, “Maximum a posteriori estimation for multivariate gaussian mixture observations of Markov chains,” *IEEE Trans. Speech and audio processing*, vol. 2, no. 2, pp. 291–298, 1994.

[31] Z. Huang, J. Li, S. M. Siniscalchi, I.-F. Chen, C. Weng, and C.-H. Lee, “Feature space maximum a posteriori linear regression for adaptation of deep neural networks,” in *Proc. INTERSPEECH*, 2014, pp. 2992–2996.

[32] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, “Optimal distributed online prediction,” in *Proc. ICML*, 2011, pp. 713–720.

[33] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[34] C. J. Leggetter and P. C. Woodland, “Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models,” *Computer Speech & Language*, vol. 9, no. 2, pp. 171–185, 1995.

[35] D. B. Paul and J. M. Baker, “The design for the wall street journal-based CSR corpus,” in *Proc. Workshop on Speech and Natural Language*, 1992, pp. 899–902.

[36] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovskỳ, G. Stemmer, and K. Veselỳ, “The Kaldi speech recognition toolkit,” in *Proc. ASRU*, 2011.

[37] K. Shinoda and C.-H. Lee, “A structural Bayes approach to speaker adaptation,” *IEEE trans. Speech and Audio Processing*, vol. 9, no. 3, pp. 276–287, 2001.

[38] N. Srivastava and R. Salakhutdinov, “Discriminative transfer learning with tree-based priors,” in *Proc. NIST*, 2013.

[39] X. Li and X. Wu, “Decision tree based state tying for speech recognition using DNN derived embeddings,” in *Proc. ISCSLP*, 2014, pp. 123–127.

[40] Z. Huang, J. Li, C. Weng, and C.-H. Lee, “Beyond cross-entropy: Towards better frame-level objective functions for deep neural network training in automatic speech recognition,” in *Proc. INTERSPEECH*, 2014.