DNN SPEAKER ADAPTATION USING PARAMETERISED SIGMOID AND RELU HIDDEN ACTIVATION FUNCTIONS

C. Zhang & P. C. Woodland

Cambridge University Engineering Dept., Trumpington St., Cambridge, CB2 1PZ U.K.

{cz277,pcw}@eng.cam.ac.uk

ABSTRACT

This paper investigates the use of parameterised sigmoid and rectified linear unit (ReLU) hidden activation functions in deep neural network (DNN) speaker adaptation. The sigmoid and ReLU parameterisation schemes from a previous study for speaker independent (SI) training are used. An adaptive linear factor associated with each sigmoid or ReLU hidden unit is used to scale the unit output value and create a speaker dependent (SD) model. Hence, DNN adaptation becomes re-weighting the importance of different hidden units for every speaker. This adaptation scheme is applied to both hybrid DNN acoustic modelling and DNN-based bottleneck (BN) feature extraction. Experiments using multi-genre British English television broadcast data show that the technique is effective in both directly adapting DNN acoustic models and the BN features, and combines well with other DNN adaptation techniques. Reductions in word error rate are consistently obtained using parameterised sigmoid and ReLU activation function for multiple hidden layer adaptation.

1. INTRODUCTION

In automatic speech recognition (ASR), adaptation techniques can reduce the distortion between training and testing acoustic conditions, which are caused by the different characteristics of different speakers/accents/styles as well as by channel and noise conditions. In particular, speaker adaptation is found to improve ASR performance by transforming speaker independent (SI) model parameters to speaker dependent (SD) versions using only a small amount of speaker-specific data [1].

While, there exists a large body of adaptation approaches developed for Gaussian mixture model (GMM) based hidden Markov models (HMMs) [1], adaptation methods for deep neural network (DNN) based HMMs is less mature. A convenient way of adapting DNNs is to use existing GMM-HMM adaptation methods such as constrained maximum likelihood linear regression (CMLLR) [2] to create SD input features [3–5]. Alternatively, different DNN oriented adaptation techniques have also been developed, which usually uses a discriminative criterion sometimes with additional regularisation [6–9]. There are various choices as to which DNN parameters to adapt including the weights and biases of standard DNN layers [6, 7, 10–14] along with extra input transforms [3, 12, 15] or the use of combination weights of several speaker cluster based DNNs [16, 17]. Another set of approaches is to supply the DNN with additional inputs to make a speaker-conditioned model such as the use of additional i-vector [18–21] and speaker code [22] input features.

Chao Zhang is supported by Cambridge International Scholarship from the Cambridge Commonwealth, European & International Trust and by the EPSRC Programme Grant EP/I031022/1 (Natural Speech Technology). Supplementary data is at https://www.repository.cam.ac.uk/handle/1810/253407.

Recently, there has been interest in using parameterised hidden activation functions for speaker adaptation [23–25]. Since the activation function is the final stage as the speech data moves through each DNN layer, activation function parameters can have a direct impact on the output of the layer. Furthermore, since there are relatively few hidden layer nodes, compared to the very large number of weights, to adapt with limited data, the use of activation function adaptation may suffer less from limited adaptation data. Previously [26], we showed that parameterised sigmoid and ReLU functions can improve SI DNN acoustic model performance. In this paper, we investigate the application of these parameterised activation functions in DNN speaker adaptation. The proposed adaptation methods are evaluated for both direct DNN-HMM acoustic modelling [27] and BN DNN feature extraction purposes [28]. All experiments are based on transcription of multi-genre broadcast audio and use unsupervised adaptation performed in batch mode.

The rest of the paper is organised as follows. Initially DNN-HMMs and BN DNNs for feature extraction are reviewed and notation introduced. Then parameterised sigmoid and ReLU functions and their use in adaptation are described. The experimental setup and results are then given followed by conclusions.

2. REVIEW OF DNNS

2.1. DNN Acoustic Feature Classifier

A DNN is a multi-layer classifier that maps an input vector $\mathbf{x}_{\text{in}}(t)$ at time t to an output vector that defines the class. $\mathbf{x}_{\text{in}}(t)$ is usually formed by stacking the acoustic feature vector $\mathbf{o}(t+c)$. c is any integer in a *context shift set* $\mathbf{c}$ that represents a time shift [29].

In each DNN layer, the input to each node j is called the activation, denoted as a_j, and a_j is defined as a weighted sum of the input vector to this layer, $\mathbf{x}_j$, based on the weights and bias associated with that node. If it is the input layer, $\mathbf{x}_j = \mathbf{x}_{\text{in}}(t)$; otherwise $\mathbf{x}_j = \mathbf{y}_i$, where $\mathbf{y}_i$ is the output from the previous layer. Node j transforms its input a_j with an activation function $f_j(\cdot)$,

$$y_j = f_j(a_j),$$

and y_j constitutes the output vector of current layer, $\mathbf{y}_j$. If $f_j(\cdot)$ has no adaptive parameters, then it is a standard activation function, such as sigmoid and ReLU. Otherwise $f_j(\cdot)$ is a parameterised function with learnable parameters.

At the output layer, the inputs to node k, a_k, are normalised to be the posterior probability of its associated class $\mathcal{C}_k$, using the *softmax function*. A DNN with softmax output function is often trained by minimising the cross entropy (CE) criterion,

$$\mathcal{F}_{\text{CE}}(t) = -\sum_k \hat{y}_k \ln \frac{y_k}{\hat{y}_k}, \tag{1}$$

where $\hat{y}_k$ is the label of $\mathcal{C}_k$ at time t. Since $\partial \mathcal{F}_{\text{CE}}(t)/\partial a_k = y_k - \hat{y}_k$ and $y_k - \hat{y}_k$ acts as the error backpropagated from the output layer, the algorithm used to compute the gradients $\partial \mathcal{F}_{\text{CE}}/\partial a_k$ is referred to as error backpropagation (EBP), and is often used together with the stochastic gradient descent (SGD) optimisation method. For SI DNN training, $\hat{y}_k$ comes from the training reference transcription [27]; for unsupervised adaptation, $\hat{y}_k$ is acquired as the output hypotheses from a previous decoding pass [1].

The partial derivative of the training objective function of a DNN parameter, e.g., an activation function parameter φ_i, is computed by backpropagting $\mathcal{F}_{\text{CE}}/\partial a_k$ from the output to the input layer. That is,

$$\frac{\partial \mathcal{F}_{\text{CE}}}{\partial a_i} = \frac{\partial f_i(a_i)}{\partial a_i} \sum_j \frac{\partial \mathcal{F}_{\text{CE}}}{\partial a_j} w_{ji}, \tag{2}$$

where w_{ji} is the weight associated with the connection between node i and j. Therefore, the gradient of φ_i that is needed for SGD is found by

$$\frac{\partial \mathcal{F}_{\text{CE}}}{\partial \varphi_i} = \frac{\partial f_i(a_i)}{\partial \varphi_i} \sum_j \frac{\partial \mathcal{F}_{\text{CE}}}{\partial a_j} w_{ji}. \tag{3}$$

2.2. DNN-HMMs and BN DNN Features

In state-of-the-art ASR systems, a DNN acoustic feature classifier is usually used either as DNN-HMM acoustic model [27] or for feature extraction [28]. If DNNs are used directly to provide state log likelihoods for HMM acoustic models, the DNN posterior probabilities $p(\mathcal{C}_k|\mathbf{x}_{\text{in}}(t))$ are converted to the log-likelihood of $\mathbf{x}_{\text{in}}(t)$ being generated by s_k as

$$\ln p(\mathbf{x}_{\text{in}}(t)|s_k) = \ln p(s_k|\mathbf{x}_{\text{in}}(t)) + \ln p(\mathbf{x}_{\text{in}}(t)) - \ln P(s_k), \tag{4}$$

where s_k is the HMM state relevant to $\mathcal{C}_k$, $P(s_k) = T_k / \sum_{k'} T_{k'}$, T_k is the number of frames labelled as s_k, and $p(\mathbf{x}_{\text{in}}(t))$ is independent of the HMM state [27].

A DNN can be used to extract discriminative features since it is a non-linear feature transform that maps its input $\mathbf{x}_{\text{in}}(t)$ to an output vector. A commonly used configuration for feature extraction is to have a reduced dimension hidden layer i.e. a BN layer [28]. Since the BN layer is normally much smaller in size than the other hidden layers, the output vector $\mathbf{y}_{\text{bn}}(t)$ is very compact and suitable to be used as features in e.g. a GMM-HMM. The training procedure for BN DNNs is usually the same as for DNN acoustic models [4, 14, 28].[1] Once the model is trained, the activation function of the BN layer is changed to a linear function $y_j = a_j$, making $\mathbf{y}_{\text{bn}}(t)$. There are different ways of using the BN features. In this paper, BN feature vectors $\mathbf{y}_{\text{bn}}(t)$ are used as a part of a BN tandem feature vector $(\mathbf{o}(t), \mathbf{y}_{\text{bn}}(t))$, which can be used in GMM-HMMs, or as input to another DNN giving stacked DNN-HMMs.

3. PARAMETERISED SIGMOID AND RELU FUNCTIONS FOR SPEAKER ADAPTATION

3.1. Parameterised Sigmoid and ReLU Functions

The parameterised sigmoid activation function introduces additional adaptive parameters to control the maximum value, steepness, and scaled horizontal displacement of the sigmoidal curve [26]. The best configuration found for SI training [26], was to only have a linear output scaling factor and this is adopted for speaker adaptation, i.e.,

$$f_i^s(a_i) = \alpha_i^s \cdot (1 + e^{-a_i})^{-1}, \tag{5}$$

where i is a hidden unit, s is a speaker, α_i^s is the SD hidden activation function parameter. According to Eqns. (2)-(3), learning α_i^s through adaptation requires $\partial f_i^s(a_i)/\partial a_i$ and $\partial f_i^s(a_i)/\partial \alpha_i^s$, which can be computed as

$$\begin{aligned}
\frac{\partial f_i^s(a_i)}{\partial a_i} &= \begin{cases} 0 & \text{if } \alpha_i^s = 0 \\ f_i^s(a_i) - (f_i^s(a_i))^2/\alpha_i^s & \text{if } \alpha_i^s \neq 0 \end{cases}, \\
\frac{\partial f_i^s(a_i)}{\partial \alpha_i^s} &= \begin{cases} (1 + e^{-a_i})^{-1} & \text{if } \alpha_i^s = 0 \\ f_i^s(a_i)/\alpha_i^s & \text{if } \alpha_i^s \neq 0 \end{cases}.
\end{aligned} \tag{6}$$

The ReLU function was parameterised by adding an individual linear factor to scale either part of its hinge-like shape [26]. The SI training experiments found that scaling only the positive part linearly gave the best performance [26]. This configuration is used for adaptation as

$$f_i^s(a_i) = \begin{cases} \alpha_i^s \cdot a_i & \text{if } a_i > 0 \\ 0 & \text{if } a_i \leqslant 0 \end{cases}, \tag{7}$$

where i and s denote the hidden unit and the speaker identity, α_i^s is the adaptive parameter for speaker adaptation. The α_i^s can be learned using

$$\begin{aligned}
\frac{\partial f_i^s(a_i)}{\partial a_i} &= \begin{cases} \alpha_i^s & \text{if } a_i > 0 \\ 0 & \text{if } a_i \leqslant 0 \end{cases}, \\
\frac{\partial f_i^s(a_i)}{\partial \alpha_i^s} &= \begin{cases} a_i & \text{if } a_i > 0 \\ 0 & \text{if } a_i \leqslant 0 \end{cases}.
\end{aligned} \tag{8}$$

The parameterised sigmoid and ReLU functions given by Eqns. (5) and (7) are denoted as p-Sigmoid(α_i^s) and p-ReLU(α_i^s) respectively. Furthermore, from Eqns. (6) and (8), it is necessary to save a_i for adaptation, which is an extra memory cost compared to standard sigmoid and ReLU functions. However, if we enforce $\partial f_i^s(a_i)/\partial \alpha_i^s = 0$ when $\alpha_i^s = 0$, the value of a_i no longer needs to be saved.

3.2. Speaker Adaptation using p-Sigmoid and p-ReLU

In order to adapt to a particular speaker s, the sigmoid or ReLU hidden activation function of the target SI hidden unit i is first replaced by a p-Sigmoid(α_i^s) or p-ReLU(α_i^s) function, with all α_i^s initialised to 1.0 so that the initial SD model is equivalent to the SI model. Then using the adaptation data from speaker s, all α^s's are jointly trained with the standard CE criterion using the EBP algorithm. If the supervision is provided as frame-state alignments, $\hat{y}_k$ follows the Bernoulli distribution and Eqn. (1) becomes

$$\begin{aligned}
\mathcal{F}_{\text{CE}}(t) &= -\sum_k \ln p(\mathcal{C}_k|\mathbf{x}_{\text{in}}(t)) \\
&\propto -\sum_k \ln p(\mathbf{x}_{\text{in}}(t)|\mathcal{C}_k).
\end{aligned}$$

Therefore, minimising CE minimises the per frame negative log posterior probabilitiy, and is equivalent to maximising the per frame log-likelihood. This follows a similar idea of the maximum log-likelihood adaptation methods [1]. Alternatively, other criteria and learning methods can also be applied to adapt the p-Sigmoid and p-ReLU parameters [6–8].

[1] One difference in the models used in this paper is that the DNN acoustic model has multiple output units corresponding to the HMM states for silence, while the BN DNN has only one unit for all silence samples.

It should be noted that learning α_i^s is equivalent to using standard activation functions with all SD weights associated with node i in the next layer scaled by $1/\alpha_i^s$ (if $\alpha_i^s \neq 0$), which if learned directly would require more parameters to be changed. This gives an insight of the power of the parameterised sigmoid and ReLU methods. Furthermore, since both Eqn. (5) and (7) can be re-written as

$$f_i^s(a_i) = \alpha_i^s \cdot f(a_i), \tag{9}$$

where $f(a_i)$ is the standard sigmoid or ReLU function, the configurations of the parameterised sigmoid and ReLU functions used in this paper fall into the learning hidden unit contribution (LHUC) framework [24], as shown by Eqn (9). A major difference between this work and LHUC adaptation [24] is that a linear scaling factor α_i^s, rather than a sigmoid function constraint scaling factor is used, which gives extra flexibility and means that these parameters can be more easily jointly learned with other DNN parameters [26].

While adaptation with an unconstrained scaling factor may be unsafe, in our experiments, we found a layer-by-layer adaptation that simulates the discriminative pre-training and "fine-tuning" method [27] is sufficient to stabilise adaptation. That is, once a hidden layer has completed adaptation for a single epoch, the next hidden layer is enabled to be jointly adapted with all α_i^s associated with the previous layers. This procedure is started from the input layer and can be continued until the desired number of hidden layers have been adapted. The next step is the "fine-tuning" that the α_i^s's of the whole network are adapted together for multiple epochs.

4. EXPERIMENTAL SETUP

The proposed techniques were evaluated by training systems on ASRU 2015 Multi-Genre Broadcast (MGB) challenge data [30]. The audio consists of seven weeks of BBC television programmes with a raw total duration of 1,600 hours. The data covers a full range of genres, e.g. news, comedy, drama, sports, quiz shows, documentaries *etc*. The audio was pre-processed using a lightly supervised decoding process, and 200 hours of data from 2,180 shows were randomly selected for which the difference between the sub-titles and the lightly supervised output had a phone matched error rate (PMER) $< 20\%$. Of the 115,932 training utterances, these were automatically clustered into 10,930 speaker clusters. A 4-gram word level language model (LM) with a 160k word vocabulary was used in all experiments. The LM was trained on 650M words of audio transcriptions and MGB additional subtitles, and was pruned with an entropy based beam of 1.0e-9. The testing set, **dev.sub**, contains 5.5 hours of audio data from 12 shows and is the official subset of the full MGB transcription development set [30]. Its manual segmentation was processed by automatic speaker clustering that resulted in 8,713 utterances and 285 speaker clusters. Further details of the data preparation, automatic DNN-based segmentation process, and the LM and dictionary can be found in [31].

All experiments were conducted with HTK 3.5 [29, 32]. Two types of acoustic features were used, 40d log-Mel filter bank (FBK) and 13d PLP coefficients, which were further expanded to 80d FBK_D and 52d PLP_D_A_T, where _D, _A, _T stand for the Δ, $\Delta\Delta$, and $\Delta\Delta\Delta$ coefficients [32]. The inputs to all DNNs were produced with $\mathbf{c} = [-4, +4]$ (equivalent to stacking $\mathbf{o}(t)$ with 4 frames in its left and right contexts) [29]. All DNNs and GMM-HMMs used the same clustered triphone state set with 6,000 non-silence tied-states. The GMM-HMMs have 16 Gaussians per state, except for the 3 silence states, which have 32 Gaussian components.

The supervision hypotheses for adaptations were generated by the joint decoding [29] of the SI tandem and DNN-HMM systems. For p-Sigmoid and p-ReLU methods, the DNNs had sigmoid and ReLU activation functions accordingly, and the resulted hypothesis WERs were 28.2% and 27.8%. The DNNs for adaptation were trained with the frame level CE criterion, and 10% of the training data was randomly selected as a held-out validation set. Parameter updates were averaged over a mini-batch with 800 frames and were smoothed by adding a "momentum" term of 0.5 times the previous update. Discriminative pre-training was used for sigmoid DNN training [27], with a learning rate of 1.0×10^{-3}. The "fine-tuning" stage uses a modified NewBob learning rate scheduler [29], with the initial learning rate of the sigmoid and ReLU DNNs set to 2.0×10^{-3} and 5.0×10^{-4} accordingly.

The SD DNN parameters were updated once per adaptation utterance, with the updates averaged over all samples in the utterance. The α_i^s parameters of both p-Sigmoid and p-ReLU were learnt using the layer-by-layer adaptation method described in Secion 3.2. The "fine-tuning" step of adaptation was performed for 6 epochs. A fixed learning rate was used throughout adaptation, which is 1.0×10^{-2} for p-Sigmoid and 2.5×10^{-3} for p-ReLU.

5. EXPERIMENTAL RESULTS

Three different sets of experiments were performed to evaluate the proposed adaptation methods. First, standard sigmoid and ReLU DNN-HMMs were adapted using p-Sigmoid and p-ReLU. Second, adaptation was applied to BN DNN features for use in GMM-HMMs. Finally, BN DNN and DNN-HMM adaptation methods were combined based on stacked DNN-HMM systems. Note that only test-time p-Sigmoid and p-ReLU adaptation is used here, so it is not used in any DNN and GMM-HMM training.

5.1. DNN-HMM Adaptation

Sigmoid and ReLU SI DNN-HMM systems with FBK_D input features were built to evaluate the hidden activation function adaptation approaches. Configurations with different hidden layers adapted are evaluated in Table 1, to show the role of each hidden layer in speaker adaptation. Comparing the different adaptation configurations, more hidden layers are progressively involved in adaptation, from the input to the output of the DNN. The SD parameters of all the different layers involved in adaptation, the α_i^s's, are jointly adapted as mentioned in Section 3.2.

From Table 1, although adapting more hidden layers consistently reduces the WER, the improvement from extra layers decreases as the new hidden layer is closer to the DNN output units. This coincides with previous findings that the low level characteristics of speech, such as speaker dependencies, are mainly modelled by the input layers, which makes them more important for speaker adaptation [24, 33]. Both p-Sigmoid and p-ReLU gave the lowest WERs by adapting all hidden layers.

n layers	p-Sigmoid	p-ReLU
0	30.6	29.9
1	28.9	28.2
2	28.5	28.0
3	28.4	28.0
4	28.3	27.8
5	28.3	27.8

Table 1. *DNN-HMM system %WERs on dev.sub with different numbers of hidden layers adapted.*

5.2. Adaptation for BN GMM-HMMs

In this section we explore the use of p-Sigmoid and p-ReLU in BN DNN adaptation for GMM-HMMs. The input features to the BN DNNs are FBK_D, and the DNN structure is $720 \times 1000^4 \times 39 \times 1000 \times 6001$. For the BN DNN, only the preceding hidden layers to the BN layer were involved in p-Sigmoid and p-ReLU speaker adaptation, as the SD parameters of the last two hidden layers were not involved in BN feature generation.

Two BN DNNs were trained separately for sigmoid and ReLU. The 91d BN tandem feature vector is formed as (PLP_D_A_T,BN). The PLP and BN features are projected by a 52d to 39d by heteroscedastic linear discriminant analysis and a semi-tied covariance matrix used for de-correlation. The BN DNN adaptation results are listed in Table 2. If CMLLR is used, BN GMM-HMMs are trained by applying CMLLR-based speaker adaptive training (SAT) [2], which have the GMM-HMMs and the per speaker CMLLR transforms estimated iteratively. Both SAT and non-SAT GMM-HMMs finally use minimum phone error (MPE) training. Extra model-based maximum likelihood linear regression (MLLR) transforms are also used for the SAT-based models. Note that once the BN features were changed due to the BN DNN adaptation, the CMLLR and MLLR transforms were re-estimated to accommodate the changes.

p-Sigmoid	CMLLR	MLLR	%WER
×	×	×	29.3
√	×	×	28.3
×	√	√	27.8
√	√	√	27.6
p-ReLU	CMLLR	MLLR	%WER
×	×	×	29.5
√	×	×	27.6
×	√	√	27.6
√	√	√	27.3

Table 2. *BN GMM-HMMs %WERs on dev.sub using BN DNN and GMM-HMM adaptation methods.*

Table 2 gives the results of applying p-Sigmoid and p-ReLU for test-time adaptation of BN DNNs. It can be seen that it can improve the BN GMM-HMM system performance, as it improves the overall quality of the testing features. Comparing p-Sigmoid and p-ReLU adaptation to the joint use of CMLLR and MLLR, although p-Sigmoid and p-ReLU resulted in smaller WER reductions, they only adapted the BN features at testing time, while CMLLR transforms were also estimated during GMM-HMM training. Meanwhile, CMLLR transformed features match the GMM-HMMs better due to the CMLLR based SAT, which is more computationally expensive than test-time only adaptation. In addition, it can also be seen that p-ReLU adaptation is complementary to the CMLLR and MLLR transforms. It should be noted that if the BN features are adapted, it is better to re-compute mean and variance normalisations, as the adaptation can cause the BN features to have a large change in range.

5.3. Stacked DNN-HMM Adaptation

A combination of the adaptation methods is finally investigated based on stacked DNN-HMM systems. The BN tandem features described in Section 5.2 were used to train the stacked DNN-HMMs. The DNN acoustic model structure is the same as that used in Section 5.1, except for the input vector size. If the de-correlation matrix and CMLLRs are used to transform the BN tandem features, the DNN input size is 702; otherwise, it is 819.

BN p-Sigmoid	CMLLR	p-Sigmoid	%WER
×	×	×	28.9
×	×	√	28.1
×	√	×	28.2
√	×	×	28.0
√	√	×	27.9
√	√	√	28.1
BN p-ReLU	CMLLR	p-ReLU	%WER
×	×	×	28.6
×	×	√	27.4
×	√	×	27.8
√	×	×	27.1
√	√	×	26.8
√	√	√	27.4

Table 3. *Stacked DNN-HMM system %WERs on dev.sub using different BN feature and DNN acoustic model adaptation methods. The 1st, 2nd, and 3rd column means the adaptation applied to the BN features, BN tandem features, and the DNN-HMM acoustic models.*

From the results in Table 3, it is clear that all of the three adaptation methods can reduce the WER. It is not surprising that for BN DNN adaptation, p-Sigmoid and p-ReLU clearly performed better than CMLLR, since adaptation was applied to multiple hidden layers and should be more powerful than using just linear transforms. Meanwhile, p-Sigmoid and p-ReLU resulted in slightly lower WERs when they were used for BN DNN adaptation rather than DNN-HMM adaptation. Perhaps ideally, we would expect the opposite pattern since both BN and PLP features can be involved in acoustic model adaptation. Therefore, this indicates the stacked DNN adaptation is more prone to overfitting than the BN DNN. Furthermore, p-Sigmoid and p-ReLU are complementary with CMLLR, since the joint use of these methods further improved the performance. Finally, if the input vectors are already well adapted, test-time only DNN-HMM adaptation is no longer useful, perhaps due to overfitting. However, SAT training may improve performance further, as in Table 2, p-Sigmoid with CMLLR based SAT achieves a WER of 27.6%, which is better than 27.9% in Table 3.

6. CONCLUSIONS

In this paper, sigmoid and ReLU hidden activation functions were parameterised with linear output value scaling factors for speaker adaptation. Several thousand parameters from the DNN-HMM, BN GMM-HMM, and stacked DNN-HMM systems were adapted for each speaker using the proposed methods at test time. A layer-by-layer adaptation scheme was used to stabilise the multi-layer speaker dependent hidden activation function parameter learning. It was found that learning activation function parameters is an effective method for speaker adaptation, which is not only complementary with the standard CMLLR feature transforms, but also more powerful in DNN adaptation scenarios.

7. REFERENCES

[1] P.C. Woodland, "Speaker adaptation for continuous density HMMs: A review," *Proc. ISCA ITR-Workshop on Adaptation Methods for Speech Recognition*, 2001.

[2] M.J.F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer speech & language*, vol. 12, pp. 75–98, 1998.

[3] F. Seide, G. Li, X. Chen, & D. Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription", *Proc. ASRU*, Waikoloa, 2011.

[4] K. Knill, M. Gales, S. Rath, P. Woodland, C. Zhang, & S.-X. Zhang, "Investigation of multilingual deep neural networks for spoken term detection", *Proc. ASRU*, Olomouc, 2013.

[5] M. Karafiát, F. Grézl, K. Veselý, M. Hannemann, I. Szokem, & J. Černocký, "BUT 2014 Babel system: Analysis of adaptation in NN based systems", *Proc. Interspeech*, Singapore, 2014.

[6] D. Yu, K. Yao, H. Su, G. Li, & F. Seide, "KL-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition", *Proc. ICASSP*, Vancouver, 2013.

[7] H. Liao, "Speaker adaptation of context dependent deep neural networks", *Proc. ICASSP*,Vancouver, 2013.

[8] Z. Huang, S.M. Siniscalchi, I.-F. Chen, J. Li, J. Wu, & C.-H. Lee, "Maximum a posteriori adaptation of network parameters in deep models", *Proc. Interspeech*, Dresden, 2015.

[9] P. Swietojanski, P. Bell, & S. Renals, "Structured output layer with auxiliary targets for context-dependent acoustic modelling," *Proc. Interspeech*, Dresden, 2015.

[10] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, & T. Robinson, "Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system", *Proc. Eurospeech*, Madrid,1995.

[11] R. Gemello, F. Mana, S. Scanzio, P. Laface, & R. De Mori, "Linear hidden transformations for adaptation of hybrid ANN/HMM models", *Speech Communication*, vol. 49, 827–835, 2007.

[12] B. Li & K.C. Sim, "Comparison of discriminative input and output transformations for speaker adaptation in the hybrid NN/HMM systems", *Proc. Interspeech*, Makuhari, 2010.

[13] T. Ochiai, S. Matsuda, X. Lu, C. Hori, & S. Katagiri, "Speaker adaptive training using deep neural networks", *Proc. ICASSP*, Florence, 2014.

[14] R.S. Doddipatla, M. Hasan, & T. Hain, "Speaker dependent bottleneck layer training for speaker adaptation in automatic speech recognition", *Proc. Interspeech*, Singapore, 2014.

[15] X. Cui & V. Goel, "Maximum likelihood nonlinear transformations based on deep neural networks", *Proc. ICASSP*, Brisbane, 2015.

[16] C. Wu & M.J.F. Gales, "Multi-basis adaptive neural network for rapid adaptation in speech recognition", *Proc. ICASSP*, Brisbane, 2015.

[17] T. Tan, Y. Qian, M. Yin, Y. Zhuang, & K. Yu, "Cluster adaptive training for deep neural network", *Proc. ICASSP*, Brisbane, 2015.

[18] G. Saon, H. Soltau, D. Nahamoo, & M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors", *Proc. ASRU*, Olomouc, 2013.

[19] Y. Miao, H. Zhang, & F. Metze, "Towards speaker adaptive training of deep neural network acoustic models", *Proc. Interspeech*, Singapore, 2014.

[20] A. Senior & I. Lopez-Moreno, "Improving DNN speaker independence with i-vector inputs", *Proc. ICASSP*, Florence, 2014.

[21] P. Karanasou, Y. Wang, M.J.F. Gales, & P.C. Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors", *Proc. Interspeech*, Singapore, 2014.

[22] O. Abdel-Hamid & H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code", *Proc. ICASSP*, Vancouver, 2013.

[23] S.M. Siniscalchi, J. Li, & C.-H. Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems", *IEEE Trans. ASLP*, vol. 21, pp. 2152–2161, 2013.

[24] P. Swietojanski & S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models", *Proc. SLT*, Lake Tahoe, 2014.

[25] Y. Zhao, J. Li, J. Xue, & Y. Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data," *Proc. ICASSP*, Brisbane, 2015.

[26] C. Zhang & P.C. Woodland, "Parameterised sigmoid and ReLU hidden activation functions for DNN acoustic modelling", *Proc. Interspeech*, Dresden, 2015.

[27] G. Hinton, L. Deng, D. Yu, D. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, & B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition", *IEEE Signal Processing Magazine*, pp. 2–17, Nov. 2012.

[28] F. Grézl, M. Karafiát, S. Kontár, & J. Černocký, "Probabilistic and bottle-neck features for LVCSR of meetings,", *Proc. ICASSP*, Honolulu, 2007.

[29] C. Zhang & P.C. Woodland, "A general artificial neural network extension for HTK", *Proc. Interspeech*, Dresden, 2015.

[30] P. Bell, M.J.F. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Wester, & P.C. Woodland, "The MGB challenge: Evaluating multi-genre broadcast media transcription", *Proc. ASRU*, Scottsdale, 2015.

[31] P.C. Woodland, X. Liu, Y. Qian, C. Zhang, M.J.F. Gales, P. Karanasou, P. Lanchantin, & L. Wang, "Cambridge University transcription systems for the multi-genre broadcast challenge", *Proc. ASRU*, Scottsdale, 2015.

[32] S. Young, G. Evermann, M. Gales, T. Hain., D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, A. Ragni, V. Valtchev, P. Woodland, & C. Zhang, *The HTK book (for HTK version 3.5)*, Cambridge University Engineering Department, 2015.

[33] A.-R. Mohamed, G. Hinton, & G. Penn, "Understanding how deep belief networks perform acoustic modelling", *Proc. ICASSP*, Kyoto, 2012.