Learning Hidden Unit Contributions for Unsupervised Speaker Adaptation of Neural Network Acoustic Models

LEARNING HIDDEN UNIT CONTRIBUTIONS FOR UNSUPERVISED SPEAKER ADAPTATION OF NEURAL NETWORK ACOUSTIC MODELS

Pawel Swietojanski and Steve Renals

Centre for Speech Technology Research, University of Edinburgh, Edinburgh EH8 9AB, UK

{p.swietojanski, s.renals}@ed.ac.uk

ABSTRACT

This paper proposes a simple yet effective model-based neural network speaker adaptation technique that learns speaker-specific hidden unit contributions given adaptation data, without requiring any form of speaker-adaptive training, or labelled adaptation data. An additional amplitude parameter is defined for each hidden unit; the amplitude parameters are tied for each speaker, and are learned using unsupervised adaptation. We conducted experiments on the TED talks data, as used in the International Workshop on Spoken Language Translation (IWSLT) evaluations. Our results indicate that the approach can reduce word error rates on standard IWSLT test sets by about 8–15% relative compared to unadapted systems, with a further reduction of 4–6% relative when combined with feature-space maximum likelihood linear regression (fMLLR). The approach can be employed in most existing feed-forward neural network architectures, and we report results using various hidden unit activation functions: sigmoid, maxout, and rectifying linear units (ReLU).

Index Terms— Speaker Adaptation, Deep Neural Networks, TED, IWSLT, LHUC

1. INTRODUCTION

In the past three years, speech recognition accuracy has been substantially improved through the use of (deep) neural network (DNN) acoustic models. Hinton et al [1] report word error rate (WER) reductions between 10–32% across a wide variety of tasks, compared with discriminatively trained Gaussian mixture model (GMM) based systems. These results use neural networks as both hybrid systems [2,3] where the neural network provides a scaled likelihood estimate which replaces the GMM, and as tandem/bottleneck systems [4, 5] in which the neural network is used as a discriminative feature extractor for a GMM-based system. Hybrid systems can provide a more powerful probability model – or can at least approximate a score a probability model would provide – compared with GMM-based systems, owing to automatically learned nonlinear feature extraction (hidden units) and better modelling of statistical dependences within and across frames.

A variety of additional acoustic model compensation and adaptation methods have been developed, to better deal with unseen speakers and mismatched acoustic backgrounds. The most successful of these is the maximum likelihood linear regression (MLLR) family of techniques developed for GMM-based systems [6, 7, 8]. In cases where speaker or channel adaptation can lead to significant improvements in accuracy, tandem systems often provide similar or reduced WERs compared with hybrid systems, owing to the applicability of GMM-based adaptation techniques [1, 9, 10].

DNNs can learn invariances through many layers of non-linear transformations, although accurate recognition of data from various acoustic conditions requires specific training approaches, such as multi-condition training [11]. Explicit adaptation to speaker or acoustic characteristics can further improve accuracy [12, 13, 14, 15, 16]. A good adaptation technique should have a compact representation – this allows the speaker-dependent parameters to be estimated from small amounts of adaptation data, and minimises storage requirements, since a different set of adaptation parameters needs to be stored for each speaker or condition. In addition, it is desirable to adapt the DNN in an unsupervised fashion without requiring labelled adaptation data.

In this paper we introduce a modified feed-forward neural network acoustic model in which there is a set of speaker-dependent parameters (one per hidden unit). Each speaker-dependent parameter corresponds to a hidden unit amplitude, and adaptation involves optimising these parameters for each speaker. We have evaluated the proposed adaptation technique across three IWSLT test sets of TED talks using DNN acoustic models with sigmoid units, maxout units, and rectifying linear units (ReLU).

2. NEURAL NETWORK ACOUSTIC ADAPTATION

Adaptation techniques for neural networks fall into three classes: feature-space transforms (speaker normalisation); auxiliary features; and model-based adaptation.

The dominant technique for estimating **feature space transforms** is constrained (feature-space) MLLR, referred to

This research was supported by EPSRC Programme Grant grant, no. EP/I031022/1 (Natural Speech Technology). Thanks to Peter Bell of University of Edinburgh for helpful discussion on GMM adaptation techniques.

as CMLLR or fMLLR [8]. fMLLR is a fully unsupervised adaptation method, which has been transferred from GMM-based to DNN-based acoustic models, in which a linear affine transform of the input acoustic features is estimated by maximising the log-likelihood that the model generates adaptation data based on first pass alignments. To use fMLLR with a DNN-based system, it is first necessary to train a complete GMM-based system, which is then used to estimate a single input transform per speaker. The transformed feature vectors are then used for DNN training and evaluation. This technique has been shown to be effective in reducing WER across several different data sets, in both hybrid and tandem approaches [17, 1, 9, 10]. Similar approaches have also been developed for neural networks. The linear input network (LIN) [12, 18] defines an additional speaker dependent layer between the input features and the first hidden layer, and thus has a similar effect to fMLLR. This technique has been further explored [19], including the use of a tied variant of LIN in which each of the input frames is constrained to have the same linear transform – feature-space discriminative linear regression (fDLR) [13, 20]. LIN/fDLR have been used in the context of both speaker-adaptive training and test-time adaptation only.

The use of augmented or **auxiliary features** is an approach to speaker-adaptive training in which the acoustic feature vectors are augmented with additional speaker-specific features computed for each speaker at both training and test stages. There has been considerable recent work exploring the use of i-vectors [21] for this purpose. The i-vectors can be regarded as the basis vectors which span a subspace of speaker variability, and were first used for adaptation in a GMM framework by Karafiat et al [22]. Saon et al [23] used i-vectors to augment the input features of DNN-based acoustic models, and showed that augmenting the input features with 100-dimensional i-vectors for each speaker resulted in a 10% relative reduction in WER on Switchboard (and a 6% reduction when the input features had been transformed using fMLLR). Gupta et al [24] obtained similar results, and Karanasou et al [25] presented an approach in which the i-vectors were factorised into speaker and environment parts. Other examples of auxiliary features include the use of speaker-specific bottleneck features obtained from a speaker separation DNN used in a distant speech recognition task [26] and the use of out-of-domain tandem features [14].

In **model-based adaptation**, the DNN parameters are adapted directly. Liao [16] investigated supervised and unsupervised adaptation of different weight subsets using a few minutes of adaptation data. On a large net (60M weights), up to 5% relative improvement was observed for unsupervised adaptation when all weights were adapted. Yu et al [15] have explored the use of regularisation for adapting the weights of a DNN, using the Kullback-Liebler (KL) divergence between the speaker-independent output distribution and the speaker-adapted output distributions, resulting in a 3% relative improvement on Switchboard. A variant of this approach reduces the number of speaker-specific parameters through a factorisation based on singular value decomposition [27]. Ochiai et al [28] have also explored regularised adaptive training of subsets of DNN parameters.

Directly adapting the weights of a large DNN results in extremely large speaker-dependent parameter sets, and a computationally intensive adaptation process. Smaller subsets of the DNN weights may be modified, including adaptation of output layer biases [20], and adaptation of the bias and slope of hidden units (a contrast experiment in [29]). Siniscalchi et al [29] also investigated the use of Hermite polynomial activation functions, whose parameters are set in a speaker adaptive fashion. Other approaches, related to the use of auxiliary features, are based on speaker codes [30, 31] in which a specific set of units for each speaker is optimised. Speaker-codes require speaker adaptive (re)-training, owing to the additional connection weights between codes and hidden units

Our goal is to develop a DNN adaptation technique which results in substantial and consistent reductions in WER, while remaining computationally efficient at adaptation time, with a compact set of speaker-specific parameters, and without requiring speaker adaptive training.

3. SPEAKER-DEPENDENT DNN

The feed-forward multi-layer perceptron (MLP) learns a nested nonlinear function $\mathbf{u}(\cdot)$ about data $\mathbf{x}$ formulated as a sequence of $L+1$ layers (hidden plus output),

$$\mathbf{u}(\mathbf{x};\theta) = \phi\left(\mathbf{U}^\top \phi^L\left(\mathbf{W}^{L\top}\phi^{L-1}\ \cdots\ \phi^1\left(\mathbf{W}^{1\top}\mathbf{x}\right)\ \cdots\right)\right), \tag{1}$$

where ϕ^l is the nonlinear transfer function at the l-th hidden layer and ϕ is the output layer transformation. We can write the output of hidden layer l as, $\mathbf{h}^l$:

$$\mathbf{h}^l = \phi^l\left(\mathbf{W}^{l\top}\mathbf{h}^{l-l}\right). \tag{2}$$

The model parameters are given by $\theta = \{\mathbf{W}_1, \ldots, \mathbf{W}_L, \mathbf{U}\}$, where we assume that the biases are included in the weight matrices $\mathbf{W}^l$. ϕ^l may take the form of a sigmoid $\phi^l = 1/(1+\exp(-c))$, rectifying linear units (ReLU) [32] $\phi^l = \max(0, c)$, or maxout units [33, 34] $\phi^l = \max_{j=i}^{i+G} c_j$.

For acoustic modelling, activations at the output layer are normalised by a *softmax* operation to produce posterior distribution over tied states at time t, s_t:

$$P(s_t|\mathbf{x}_t;\theta) = \frac{\exp(\mathbf{U}_{s_t}^\top \phi^L)}{\sum_{s'}\exp(\mathbf{U}_{s'}^\top \phi^L)}, \tag{3}$$

This model is usually trained in a speaker independent (SI) fashion: a set of training speech examples $\{(\mathbf{x}_t, s_t)\}_{t=1}^T$ produced by some number of distinct speakers is used to train the network. The objective of speaker adaptation is to adjust the parameters such that the acoustic model generalises better

to unseen talkers. This is achieved by using some amount of adaptation data $\{(\mathbf{x}_t^m, s_t^m)\}_{t=1}^{T^m}$, $T^m << T$ for speaker m in order to refine the model such that it better approximates the posterior distribution $P(s_t|\mathbf{x}_t^m; \theta^m, \theta)$ for a given speaker.

We modify the speaker independent model (1) by defining a set of speaker-dependent (SD) parameters for speaker m, $\theta^m = \{\mathbf{r}_m^1, \ldots, \mathbf{r}_m^L\}$, where $\mathbf{r}_m^l \in \mathbb{R}^{M^l}$ is the vector of SD parameters for the lth hidden layer. If $a(\mathbf{r}_m^l)$ is element-wise function that constrains the range of $\mathbf{r}_m^l$, then we can modify (2) to define an SD hidden layer output:

$$\mathbf{h}_m^l = a(\mathbf{r}_m^l) \circ \phi^l \left(\mathbf{W}^{l\top} \mathbf{h}_m^{l-l} \right) , \tag{4}$$

where $\circ$ is an element-wise multiplication. We have chosen to define $a(\cdot)$ as a sigmoid with amplitude 2, $a(c) = 2/(1 + \exp(-c))$. (Other options are also possible, e.g. $\max(0, \min(2, c))$; preliminary experiments indicated that this did not affect the WER.). This re-parametrisation is for optimisation purposes only; at runtime $a(\cdot)$ can be evaluated once for a given set of θ^m and directly used as a scaling factor. The SD term can be viewed as weighting the hidden unit contributions; if the gain of $a(\mathbf{r}_m^{\cdot})$ is set to 1.0, then the SI and SD models are equivalent.

We refer to this approach as Learning Hidden Unit Contributions (LHUC). The function $a(\cdot)$ has been chosen to constrain the range of $\mathbf{r}_m$ to $[0, 2]$ which simplifies interpretability and the next layer receives the expected re-weighted average input – some activations can be turned off entirely but some have chance to compensate with doubled amplitude while some other may remain unchanged. This formulation has several advantages: first, the total number of adaptation parameters stays relatively low - at most the total number of hidden units, $\sum_l^L M^l$; second, since it does not make assumptions about the form of nonlinearity, neither the internal structure of a layer, it may be applied to any feed-forward neural network; third, it does not rely on speaker adaptive training which makes the training and adaptation processes simpler allowing standard SI DNN components to be reused; finally, it does not change the learned feature detectors, which makes the technique more robust against over-fitting and catastrophic forgetting [35].

The SD parameters are optimised with respect to the negative log posterior probability $\mathcal{F}(\theta^m)$ over T^m adaptation datapoints of the m-th speaker, similar to the SI case:

$$\mathcal{F}(\theta^m) = -\sum_t^{T^m} \log P(s_t|\mathbf{x}_t^m; \theta^m) . \tag{5}$$

(Other cost functions could also be used such as regularising using the KL divergence between SI and SD models [15].)

For the top hidden layer the SD parameters' gradient is:

$$\frac{\partial \mathcal{F}}{\partial \mathbf{r}_m^L} = \frac{\partial \mathcal{F}}{\partial a(\mathbf{r}_m^L)} \frac{\partial a(\mathbf{r}_m^L)}{\partial \mathbf{r}_m^L}, \tag{6}$$

where $\partial a(\mathbf{r}_m^L)/\partial \mathbf{r}_m^L$ depends on $a(\cdot)$ and $\partial \mathcal{F}/\partial a(\mathbf{r}_m^L)$ is obtained using (3) and (4):

$$\begin{aligned}
\frac{\partial \mathcal{F}}{\partial a(\mathbf{r}_m^L)} &= -\sum_t \frac{\partial}{\partial a(\mathbf{r}_m^L)} \Bigg(\exp(\mathbf{U}_s^\top a(\mathbf{r}_m^L) \circ \phi_t^L) \\
&\qquad - \log \sum_{s'} \exp(\mathbf{U}_{s'}^\top a(\mathbf{r}_m^L) \circ \phi_t^L) \Bigg) \\
&= -\sum_t \left(\mathbf{U}_{s_t}^\top - \sum_s \frac{\exp(\mathbf{U}_s^\top a(\mathbf{r}_m^L) \circ \phi_t^L)}{\sum_{s'} \exp(\mathbf{U}_{s'}^\top a(\mathbf{r}_m^L) \circ \phi_t^L)} \mathbf{U}_s^\top \right) \phi_t^L \\
&= -\sum_t \left(\mathbf{U}_{s_t}^\top - \sum_s P(s|\mathbf{x}_t^m) \mathbf{U}_s^\top \right) \phi_t^L .
\end{aligned} \tag{7}$$

The remaining gradients for the SD parameters in the lower layers $\mathbf{r}_m^l$, $l \in \{1 \ldots L-1\}$ can be computed using the chain rule

$$\frac{\partial \mathcal{F}}{\partial \mathbf{r}_m^l} = \frac{\partial \mathcal{F}}{\partial \phi^L} \frac{\partial \phi^L}{\partial \phi^{L-1}} \cdots \frac{\partial \phi^{l+2}}{\partial \phi^{l+1}} \frac{\partial \phi^{l+1}}{\partial a(\mathbf{r}_m^l)} \frac{\partial a(\mathbf{r}_m^l)}{\partial \mathbf{r}_m^l}, \tag{8}$$

where $\partial \mathcal{F}/\partial \phi^L$ is obtained similar to (7)

$$\frac{\partial \mathcal{F}}{\partial \phi^L} = -\sum_t \left(\mathbf{U}_{s_t}^\top - \sum_s P(s|\mathbf{x}_t^m) \mathbf{U}_s^\top \right) a(\mathbf{r}_m^L), \tag{9}$$

and by using an auxiliary identity $\mathbf{z}_t^{l+1} = \mathbf{W}^{l+1\top} a(\mathbf{r}_m^l) \circ \phi_t^l$ the remaining partials for (8) are given by (10) and (11).

$$\frac{\partial \phi_t^{l+1}}{\partial \phi_t^l} = \frac{\partial \phi_t^{l+1}}{\partial \mathbf{z}_t^{l+1}} \frac{\partial \mathbf{z}_t^{l+1}}{\partial \phi_t^l} = \frac{\partial \phi_t^{l+1}}{\partial \mathbf{z}_t^{l+1}} \mathbf{W}^{l+1\top} a(\mathbf{r}_m^l) \tag{10}$$

$$\frac{\partial \phi_t^{l+1}}{\partial a(\mathbf{r}_m^l)} = \frac{\partial \phi_t^{l+1}}{\partial \mathbf{z}_t^{l+1}} \frac{\partial \mathbf{z}_t^{l+1}}{\partial a(\mathbf{r}_m^l)} = \frac{\partial \phi_t^{l+1}}{\partial \mathbf{z}_t^{l+1}} \mathbf{W}^{l+1\top} \phi_t^l \tag{11}$$

$\partial \phi_t^{l+1}/\partial \mathbf{z}_t^{l+1}$ depends on the activation function in the given layer: it is 1.0 for maxout and ReLU (positive slope, otherwise 0), for sigmoid it is $\phi^{l+1}(1 - \phi^{l+1})$.

Learning the amplitides of hidden unit activation functions has been previously suggested [36]: however, in that work the amplitudes were not shared between speakers (or some other adaptation class), and the amplitudes were estimated on the the training set, similar to the other weights.

4. EXPERIMENTAL SETUP

We performed experiments using a corpus of publicity available TED talks (`http://www.ted.com`) following the IWSLT ASR evaluation protocol [37] (`http://iwslt.org`). Our baseline systems are very similar to [14] with some minor acoustic model refinements. We used a lightweight pruned language model that is inferior to a later

model estimated on more data [38]. The training data consisted of 813 publicly available TED talks published before the end of 2010. After automatic segmentation and lightly-supervised alignment 143 hours of speech remained for training purposes. We present results on three predefined IWSLT test sets: `dev2010`, `tst2010` and `tst2011` containing 8, 11 and 8 talks of about 10 minutes duration, respectively.

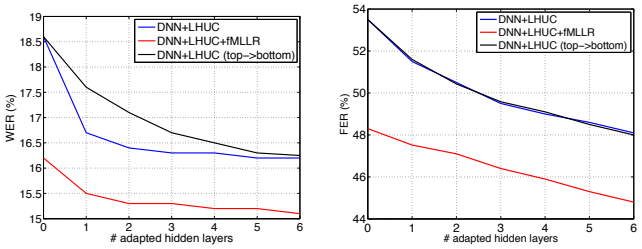
For acoustic modelling we use a DNN with 6 hidden layers and 2048 units per layer for element-wise activations (sigmoid, ReLU), together with 12000 tied states (outputs). The input features had a dimension of 351: PLP-12 (including C0), with first and second derivatives, with ± 4 frames of context. For the maxout pooling nonlinearity we set the number of hidden maxout units to 1500 with a group size of 2, following our previous work [39]. For each neural network we sampled initial weights from a uniform distribution with range $\pm k$. For ReLU and maxout we used $k = 0.005$, while for sigmoid we used a normalised initialisation with $k = 4\sqrt{6/(M^l + M^{l+1})}$ [40]. All models are fine-tuned with the exponentially decaying "newbob" learning rate schedule[1] staring from an initial learning rate of 0.08 (for sigmoid) and 0.01 for piece-wise linear activation functions. Models are adapted with a large learning rate of 0.8. We used the open source Kaldi toolkit [41], with DNNs estimated using PyLearn2 [42].

5. RESULTS

Most of the analyses in this section use `tst2010` and DNN models with a sigmoid nonlinearity. A summary of results using all three test sets and the sigmoid, maxout, and ReLU hidden layers is presented in Table 1. All adaptation experiments, unless explicitly stated otherwise, were unsupervised.

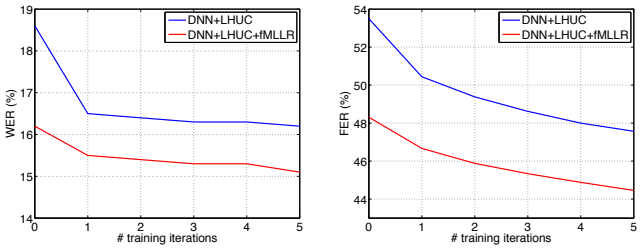
First, we investigated how many and which hidden layers should be adapted on a per speaker basis. Figure 1 (a) shows the effect of LHUC, by progressively adapting layers from the bottom (closest to the input) to the top. The lowest hidden layer is most important, with the overall WER flattening after 3–4 adapted layers. Learning hidden unit contributions from the top hidden layer yields worse WER – 17.5% if only the top hidden layer is adapted compared with 16.6% when the bottom hidden layer is adapted. The frame error rate (FER, Figure 1 (b)) steadily decreases with the number of adapted hidden layers; interestingly from the order in which layers were inserted was irrelevant with respect to FER.

Figure 2 plots WER and FER against the number of iterations of adaptation, using the same alignments obtained from the first pass decoding lattices. Most of the WER decrease was obtained after one iteration of adaptation; further iterations brought only small reductions in WER, although FER steadily decreases. It may also be observed that LHUC adaptation is partially complementary with fMLLR-based speaker normalisation.



(a) (b)

Fig. 1. a) WER(%) and b) FER(%) on `tst2010` as a function of the number of layers with learned hidden unit contributions. The SD parameters were inserted from the bottom (closest to input) to the top.



(a) (b)

Fig. 2. a) WER(%) and b) FER(%) on `tst2010` as a function of the number of adaptation iterations. All hidden unit contributions are adapted.

In the following experiments we carry out LHUC adaptation on all layers and adapt for 3 iterations. We investigated how the amount of adaptation data affects WER by randomly selecting adaptation utterances to give totals of 10s, 30s, 60s, 120s and 300s of speaker-specific adaptation data for each talker. Figure 3 (a) shows average WERs after repeating the experiments 5 times. 10s of unsupervised adaptation data decreases the WER of the large (46M weights) speaker independent model by 3% relative. This is further improved when adapting with more data to 5.4% relative with 30s and 7% relative with 60s. A full two-pass decoding yields a WER of 16.2% (13% relative improvement) – which is comparable to the result obtained with speaker adaptive fMLLR training[2]. Combining both methods further decreased WER to 15.1% (18.7% relative improvement).

Figure 3 also presents an oracle experiment in which the adaptation targets were obtained by aligning the audio data with reference transcripts. We performed this experiment for analysis, rather than considering it in terms of system

[1]Developed as part of ICSI QuickNet: `http://www.icsi.berkeley.edu/Speech/qn.html`

[2]Note, fMLLR transforms were estimated once based on GMM first pass alignments using all data for a given talker

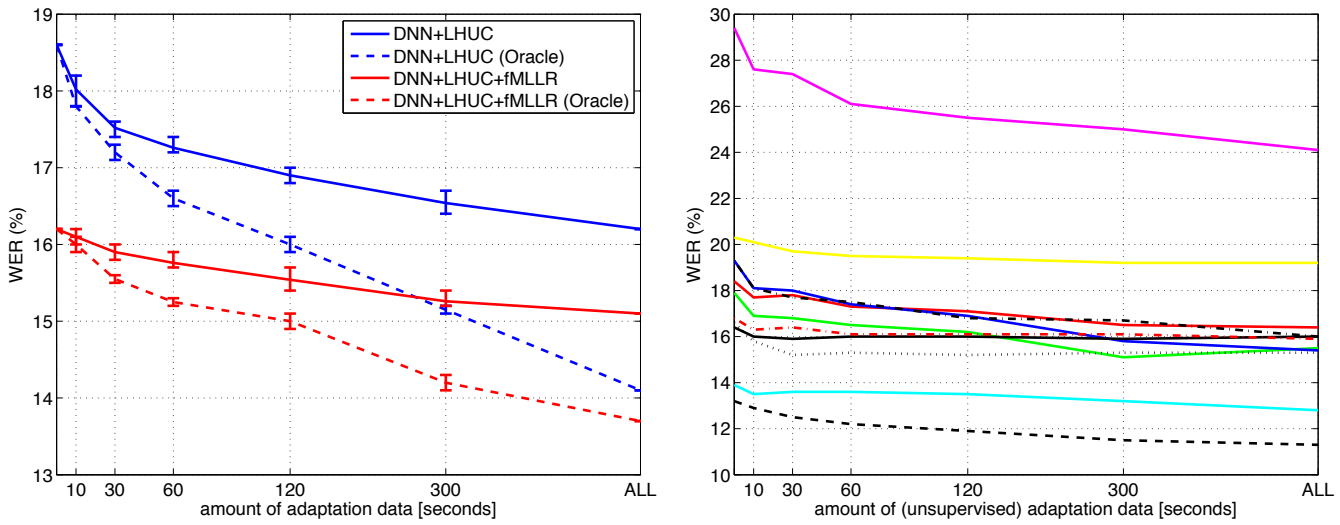


Fig. 3. WER(%) for different amounts of adaptation data a) average scores b) split for 11 speakers of `tst2010`

Table 1. Summary of WER (%) results on IWSLT12 TED evaluation sets. Relative improvements are given in parentheses w.r.t. the corresponding speaker independent model.

Model	`dev2010`	`tst2010`	`tst2011`
DNN	19.3	18.6	15.2
+LHUC	17.3 (-10.4)	16.2 (-12.9)	13.7 (-9.9)
+fMLLR	17.4 (-9.8)	16.2 (-12.9)	13.9 (-8.5)
+LHUC	16.2 (-16)	15.1 (-18.7)	12.9 (-15.1)
ReLU	19.3	18.4	15.2
+LHUC	17.8 (-7.8)	15.7 (-14.7)	13.5 (-11.2)
+fMLLR	17.7 (-8.3)	15.7 (-14.7)	13.6 (-10.5)
+LHUC	16.9 (-12.4)	14.6 (-21.2)	12.7 (-16.4)
Maxout	19.0	18.0	14.3
+LHUC	17.1 (-10)	15.6 (-13.3)	12.8 (-10.4)
+fMLLR	16.9 (-11.1)	15.4 (-14.4)	12.5 (-12.6)
+LHUC	16.3 (-14.2)	14.6 (-18.9)	11.9 (-16.8)

improvements, to demonstrate the modelling capacity of LHUC adaptation. Without refining what the model knows about speech, nor the way it classifies it (since the feature receptors and output layer are fixed during adaptation and remain speaker independent), we show that the recomposition of these "basis functions" is able to decrease the WER by 24.7% relative for LHUC-only adaptation and 26.3% relative when combined with fMLLR. This indicates that effective adaptation methods can be designed in the space of speaker-independent speech components (which can be robustly estimated on hundreds of hours of training data) while the final SD model is composed by an appropriate selection of a relatively small number of weighting coefficients.

Figure 3 (b) also shows the WERs separately for each of the 11 speakers from `tst2010`. We can see very good adaptation results can be also obtained for more noisy adaptation targets: the largest decreases in WER is for the speaker with the highest WER. Although much more evidence is required, this is an indication that the LHUC method is not very sensitive to inaccuracies in adaptation targets.

Finally, we report the complete results for the three predefined IWSLT12 test sets and different model types in Table 1. We can see the LHUC works well also with non- or partially-bounded activation functions (Maxout, ReLU) giving up to 21% relative improvement for combined feature- and model-based adaptation. The average gain from the combined LHUC and fMLLR adaptation of Maxout models is consistently smaller when compared to other models; we note that the Maxout hidden dimensionality is smaller (1500 hidden units per layer versus 2048) and as such there are fewer SD adaptation parameters.

Figure 4 shows the maximum, mean, and minimum values of $\mathbf{r}$ for an example speaker from `tst2010` – note the close to symmetrical distribution of gains across layers with nearly zero mean.

6. CONCLUSIONS

We have presented an adaptation technique that learns hidden unit contributions on a per speaker basis. The method is able to fully reuse most SI DNN model architectures. Our results across three IWSLT test sets indicate that the approach consistently reduces word error rates by 8–15% relative compared to unadapted systems, with a further reduction of 4–6% relative when combined with fMLLR. Our experiments also show that the technique may be used with various hidden unit activation functions: sigmoid, maxout, and ReLU.

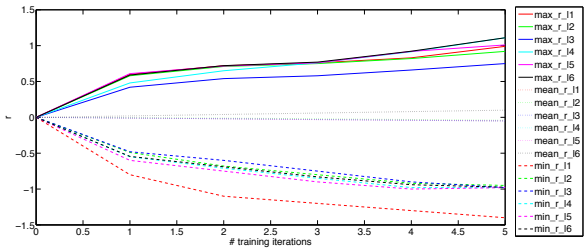


Fig. 4. Maximum, mean, and minimum values of $\mathbf{r}$ for one of the speakers in `tst2010`; $\mathbf{r} = 0$ corresponds to a gain of 1.

7. REFERENCES

[1] G Hinton, L Deng, D Yu, GE Dahl, A Mohamed, N Jaitly, A Senior, V Vanhoucke, P Nguyen, TN Sainath, and B Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, Nov 2012.

[2] H Bourlard and N Morgan, *Connectionist Speech Recognition: A Hybrid Approach*, Kluwer Academic Publishers, 1994.

[3] S Renals, N Morgan, H Bourlard, M Cohen, and H Franco, "Connectionist probability estimators in HMM speech recognition," *IEEE Trans Speech and Audio Processing*, vol. 2, pp. 161–174, 1994.

[4] H Hermansky, DPW Ellis, and S Sharma, "Tandem connectionist feature extraction for conventional HMM systems," in *Proc IEEE ICASSP*, 2000, pp. 1635–1638.

[5] F Grézl, M Karafiát, S Kontár, and J Černocký, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc IEEE ICASSP*, 2007, pp. IV–757–IV–760.

[6] CJ Leggetter and PC Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer Speech & Language*, vol. 9, pp. 171–185, 1995.

[7] VV Digalakis and LG Neumeyer, "Speaker adaptation using combined transformation and bayesian methods," *IEEE Transactions on Speech and Audio Processing*, vol. 4, pp. 294–300, 1996.

[8] MJF Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech and Language*, vol. 12, pp. 75–98, April 1998.

[9] TN Sainath, B Kingsbury, and B Ramabhadran, "Auto-encoder bottleneck features using deep belief networks.," in *Proc IEEE ICASSP*, 2012, pp. 4153–4156.

[10] P Bell, P Swietojanski, and S Renals, "Multi-level adaptive networks in tandem and hybrid ASR systems," in *Proc IEEE ICASSP*, 2013.

[11] M Seltzer, D Yu, and Y Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Proc IEEE ICASSP*, 2013.

[12] 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, pp. 2171–2174.

[13] F Seide, X Chen, and D Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proc IEEE ASRU*, 2011.

[14] P Swietojanski, A Ghoshal, and S Renals, "Revisiting hybrid and GMM-HMM system combination techniques," in *Proc IEEE ICASSP*, 2013.

[15] 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 IEEE ICASSP*, 2013, pp. 7893–7897.

[16] H. Liao, "Speaker adaptation of context dependent deep neural networks.," in *In Proc. ICASSP*. 2013, pp. 7947–7951, IEEE.

[17] T Hain, L Burget, J Dines, PN Garner, F Grézl, A El Hannani, M Karafíat, M Lincoln, and V Wan, "Transcribing meetings with the AMIDA systems," *IEEE Trans Audio, Speech and Language Processing*, vol. 20, pp. 486–498, 2012.

[18] V Abrash, H Franco, A Sankar, and M Cohen, "Connectionist speaker normalization and adaptation," in *Proc Eurospeech*, 1995, pp. 2183–2186.

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

[20] 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 IEEE SLT*, 2012.

[21] N Dehak, PJ Kenny, R Dehak, P Dumouchel, and P Ouellet, "Front end factor analysis for speaker verification," *IEEE Trans Audio, Speech and Language Processing*, vol. 19, pp. 788–798, 2010.

[22] M Karafiát, L Burget, P Matějka, O Glembek, and J Černozký, "iVector-based discriminative adaptation for automatic speech recognition," in *Proc IEEE ASRU*, 2011.

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

[24] V Gupta, P Kenny, P Ouellet, and T Stafylakis, "I-vector based speaker adaptation of deep neural networks for french broadcast audio transcription," in *Proc IEEE ICASSP*, 2014.

[25] P Karanasou, Y Wang, MJF Gales, and PC Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proc Interspeech*, 2014.

[26] Y Liu, P Zhang, and T Hain, "Using neural network front-ends on far field multiple microphones based speech recognition," in *Proc IEEE ICASSP*, 2014.

[27] 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 IEEE ICASSP*, 2014.

[28] T Ochiai, S Matsuda, X Lu, C Hori, and S Katagiri, "Speaker adaptive training using deep neural networks," in *Proc IEEE ICASSP*, 2014.

[29] SM Siniscalchi, J Li, and CH Lee, "Hermitian polynomial for speaker adaptation of connectionist speech recognition systems," *IEEE Trans Audio, Speech, and Language Processing*, vol. 21, pp. 2152–2161, 2013.

[30] JS Bridle and S Cox, "Recnorm: Simultaneous normalisation and classification applied to speech recognition," in *Advances in Neural Information Processing Systems 3*, 1990, pp. 234–240.

[31] O Abdel-Hamid and J Hui, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *Proc IEEE ICASSP*, 2013, pp. 4277–4280.

[32] V Nair and G Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 131–136.

[33] M Riesenhuber and T Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

[34] IJ Goodfellow, D Warde-Farley, M Mirza, A Courville, and Y Bengio, "Maxout networks," *arXiv:1302.4389*, 2013.

[35] RM French, "Catastrophic forgetting in connectionist networks: Causes, consequences and solutions," *Trends in Cognitive Sciences*, vol. 3, pp. 128–135, 1999.

[36] E Trentin, "Networks with trainable amplitude of activation functions," *Neural Networs*, vol. 14, pp. 471–493, 2001.

[37] M Federico, M Cettolo, L Bentivogli, M Paul, and S Stüker, "Overview of the IWSLT 2012 evaluation campaign," in *Proc IWSLT*, 2012.

[38] P Bell, H Yamamoto, P Swietojanski, Y Wu, F McInnes, C Hori, and S Renals, "A lecture transcription system combining neural network acoustic and language models," in *Proc Interspeech*, 2013.

[39] P Swietojanski, J Li, and J-T Huang, "Investigation of maxout networks for speech recognition," in *Proc IEEE ICASSP*, 2014.

[40] X Glorot and Y Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc AISTATS*, 2010.

[41] D Povey, A Ghoshal, G Boulianne, L Burget, O Glembek, N Goel, M Hannemann, P Motlíček, Y Qian, P Schwarz, J Silovský, G Stemmer, and K Veselý, "The Kaldi speech recognition toolkit," in *Proc. IEEE ASRU*, December 2011.

[42] IJ Goodfellow, D Warde-Farley, P Lamblin, V Dumoulin, M Mirza, R Pascanu, J Bergstra, F Bastien, and Y Bengio, "Pylearn2: a machine learning research library," *arXiv preprint arXiv:1308.4214*, 2013.