Differentiable Pooling for Unsupervised Acoustic Model Adaptation

Pawel Swietojanski, *Member, IEEE,* and Steve Renals, *Fellow, IEEE*

***Abstract*—We present a deep neural network (DNN) acoustic model that includes parametrised and differentiable pooling operators. Unsupervised acoustic model adaptation is cast as the problem of updating the decision boundaries implemented by each pooling operator. In particular, we experiment with two types of pooling parametrisations: learned L_p-norm pooling and weighted Gaussian pooling, in which the weights of both operators are treated as speaker-dependent. We perform investigations using three different large vocabulary speech recognition corpora: AMI meetings, TED talks and Switchboard conversational telephone speech. We demonstrate that differentiable pooling operators provide a robust and relatively low-dimensional way to adapt acoustic models, with relative word error rates reductions ranging from 5–20% with respect to unadapted systems, which themselves are better than the baseline fully-connected DNN-based acoustic models. We also investigate how the proposed techniques work under various adaptation conditions including the quality of adaptation data and complementarity to other feature- and model-space adaptation methods, as well as providing an analysis of the characteristics of each of the proposed approaches.**

I. Introduction and Summary

DEEP neural network (DNN) acoustic models have significantly extended the state-of-the-art in speech recognition [1] and are known to be able to learn significant invariances through many layers of non-linear transformations [2]. If the training and deployment conditions of the acoustic model are mismatched then the runtime data distribution can differ from the training distribution, bringing a degradation in accuracy, which may be addressed through explicit adaptation to the test conditions [2]–[9].

In this paper we explore the use of parametrised and differentiable pooling operators for acoustic adaptation. We introduce the approach of differentiable pooling using speaker-dependent pooling operators, specifically L_p-norm pooling and weighted Gaussian pooling (Section III), showing how the pooling parameters may be optimised by minimising the negative log probability of the class given the input data (Section IV), and providing a justification for the use of pooling operators in adaptation (Section V). To evaluate this novel adaptation approach we performed experiments on three corpora – TED talks, Switchboard conversational telephone speech, and AMI meetings – presenting results on using differentiable pooling for speaker independent acoustic modelling, followed by unsupervised speaker adaptation experiments in which adaptation of the pooling operators is compared (and combined) with learning hidden unit contributions (LHUC) [10], [11] and constrained/feature-space maximum likelihood linear regression (fMLLR) [12].

II. DNN Acoustic Modelling and Adaptation

DNN acoustic models typically estimate the posterior distribution over a set of context-dependent tied states s of a hidden Markov model (HMM) [13] given an acoustic observation $\mathbf{o}$, $P(s|\mathbf{o}) = \mathrm{DNN}(\mathbf{o};\boldsymbol{\theta})$ [1], [14], [15]. The DNN is implemented as a nested function comprising L processing layers (non-linear transformations):

$$\mathrm{DNN}(\mathbf{o};\boldsymbol{\theta}) = f^L\left(f^{L-1}\left(\dots f^1\left(\mathbf{o};\theta^1\right)\dots;\theta^{L-1}\right);\theta^L\right) \quad (1)$$

The model is thus parametrised by a set of weights $\boldsymbol{\theta} = \{\boldsymbol{\theta}^l\}_{l=1}^L$ in which the lth layer consists of a weight matrix and bias vector, $\boldsymbol{\theta}^l = \{\mathbf{W}^l, \mathbf{b}^l\}$, followed by a non-linear transformation ϕ, acting on arbitrary input $\mathbf{x}$:

$$f^l(\mathbf{x};\boldsymbol{\theta}^l) = \phi^l\left(\mathbf{W}^{l\top}\mathbf{x} + \mathbf{b}^l\right) \quad (2)$$

To form a probability distribution, the output layer employs a *softmax* transformation [16] $\phi_i^L(\mathbf{x}) = \exp(x_i)/\sum_j \exp(x_j)$, whereas the hidden layer activation functions are typically chosen to be either sigmoid $\phi^l(x) = 1/(1+\exp(-x))$ or rectified linear $\phi^l(x) = \max(0,x)$ units (ReLU) [17].

Yu et al [2] experimentally demonstrated that the invariance of the internal representations with respect to variabilities in the input space increases with depth (the number of layers) and that the DNN can interpolate well around training samples but fails to extrapolate if the data mismatch increases. Therefore one often explicitly compensates for unseen variabilities in the acoustic space.

Feature-space normalisation increases the invariance to unseen data by transforming the data such that it better matches the training data. In this approach the DNN learns an additional transform of the input features conditioned on the speaker or the environment. The transform, which is typically affine, is parametrised by an additional set of adaptation parameters. The most effective form of feature-space normalisation is constrained (feature-space) maximum-likelihood linear regression (MLLR), referred to as fMLLR [12], in which the linear transform parameters are estimated by maximising the likelihood of the adaptation data under a Gaussian Mixture

P Swietojanski and S Renals are with The Centre for Speech Technology Research, University of Edinburgh, Edinburgh EH89AB, U.K., (e-mail: p.swietojanski@ed.ac.uk, s.renals@ed.ac.uk)

The authors were supported by EPSRC Programme Grant grant EP/I031022/1 *Natural Speech Technology* (NST) and the European Union under H2020 project *SUMMA*, grant agreement 688139.

10.1109/TASLP.2016.2584700 © 2016 IEEE.

Model (GMM) / HMM acoustic model. To use fMLLR with a DNN acoustic model it is necessary to estimate a single input transform per speaker (using a trained GMM), using the resultant transformed data to train a DNN in a speaker adaptive training (SAT) manner. At runtime another set of fMLLR parameters is estimated for each speaker and the data transformed accordingly. This technique has consistently and significantly reduced the word error rate (WER) across several different benchmarks for both hybrid [1], [14] and tandem [18], [19] approaches. There are many successful examples of fMLLR adaptation of DNN acoustic models [1], [6], [8], [20]–[25]. One can also estimate the linear transform as an input layer of the DNN, often referred to as a linear input network (LIN) [3], [4], [6], [26]. LIN-based approaches have been mostly used in test-only adaptation schemes, whereas fMLLR requires SAT, but usually results in lower WERs.

Auxiliary feature approaches augment the acoustic feature vectors with additional speaker-dependent information computed for each speaker at both training and runtime stages – this is a form of SAT in which the model learns the distribution over tied states conditioned on some additional speaker-specific information. There has been considerable recent work exploring the use of i-vectors [27] for this purpose. I-vectors, which can be regarded as basis vectors spanning a subspace of speaker variability, were first used for adaptation in a GMM framework by Karafiat et al [28], and were later successfully employed for DNN adaptation [29]–[34]. Other examples of auxiliary features include the use of speaker-specific bottleneck features obtained from a speaker separation DNN [35], the use of out-of-domain tandem features [24], as well as speaker codes [36]–[38] in which a specific set of units for each speaker is optimised. Kundu et al. [39] present an approach using auxiliary input features derived from the bottleneck layer of a DNN which is combined with i-vectors.

Model-based approaches adapt the DNN parameters using data from the target speaker. Liao [40] investigated this approach in both supervised and unsupervised settings using a few minutes of adaptation data. On a large DNN, when all weights were adapted, up to 5% relative improvement was observed for unsupervised adaptation, using a speaker independent decoding to obtain DNN targets. Yu et al [9] have explored the use of regularisation for adapting the weights of a DNN, using the Kullback-Liebler (KL) divergence between the output distributions produced by speaker-independent and the speaker-adapted models. This approach was also recently used to adapt parameters of sequence-trained models [41]. LIN may also be regarded as a form of model-based adaptation, and related approaches include adaptation using a linear output network (LON) or linear hidden network (LHN) [4], [7], [42].

Directly adapting all the weights of a large DNN is computationally and data intensive, and results in large speaker-dependent parameter sets. Smaller subsets of the DNN weights may be modified, including biases and slopes of hidden units [7], [34], [43], [44]. Another recently developed approach relies on learning hidden unit contributions (LHUC) for test-only adaptation [10], [11] as well as in a SAT framework [45]. One can also adapt the top layer using Bayesian methods resulting in a maximum a posteriori (MAP) approach [46], or address the sparsity of context-dependent tied-states when few adaptation data-points are available by modelling both monophones and context-dependent tied states using multi-task adaptation [47], [48] or a hierarchical output layer [49].

III. Differentiable Pooling

Building on our initial work [50], we present an approach to adaptation by learning hidden layer pooling operators with parameters that can be learned and adapted in a similar way to the other model parameters. The idea of feature pooling originates from Hubel and Wiesel's pioneering study on visual cortex in cats [51], and was first used in computer vision to combine spatially local features [52]. Pooling in DNNs involves the combination of a set of hidden unit outputs into a summary statistic. Fixed poolings are typically used, such as average pooling (used in the original formulation of convolutional neural networks – CNNs) [53], [54] and max pooling (used in the context of feature hierarchies [55] and later applied to CNNs [56], [57]).

Reducing the dimensionality of hidden layers by pooling some subsets of hidden unit activations has become well investigated beyond computer vision, and the $\max$ operator has been interpreted as a way to learn piecewise linear activation functions – referred to as Maxout [58]. Maxout has been widely investigated for both fully-connected [59]–[61] and convolutional [62], [63] DNN-based acoustic models. Max pooling, although differentiable, performs a one-from-K selection, and hence does not allow hidden unit outputs to be interpolated, or their combination to be learned within a pool.

There have been a number of approaches to pooling with differentiable operators – *differentiable pooling* – a notion introduced by Zeiler and Fergus [64] in the context of constructing unsupervised feature extract for support vector machines in computer vision tasks. There has been some interest in the use of L_p-norm pooling with CNN models [57], [65] in which the sufficient statistic is the p-norm of the group of (spatially-related) hidden unit activations. Fixed order L_p-norm pooling was recently applied within the context of a convolutional neural network acoustic model [66], where it did not reduce the WER over max-pooling, and as an activation function in a fully-connected DNNs [67], where it was found to improve over maxout and ReLU.

A. L_p-norm (`Diff-`L_p) pooling

In this approach we pool a set of activations using an L_p-norm. A hidden unit pool is formed by a set R_k of K affine projections which form the input to the kth pooling unit, which we write as an ordered set (vector) $\mathbf{a}^k = \{\mathbf{w}_i^\top \mathbf{x} + b_i\}_{i \in R_k}$. The output of the kth pooling unit is produced as an L_p norm:

$$f_{L_p}\left(\mathbf{a}^k; p_k\right) = ||\mathbf{a}^k||_{p_k} = \left(\frac{1}{K}\sum_{i \in R_k} |a_i^k|^{p_k}\right)^{\frac{1}{p_k}}, \quad (3)$$

where p_k is the learnable norm order for the kth unit, that can be jointly optimised with the other parameters in the model. To ensure that (3) satisfies a triangle inequality ($p_k \geq 1$; a

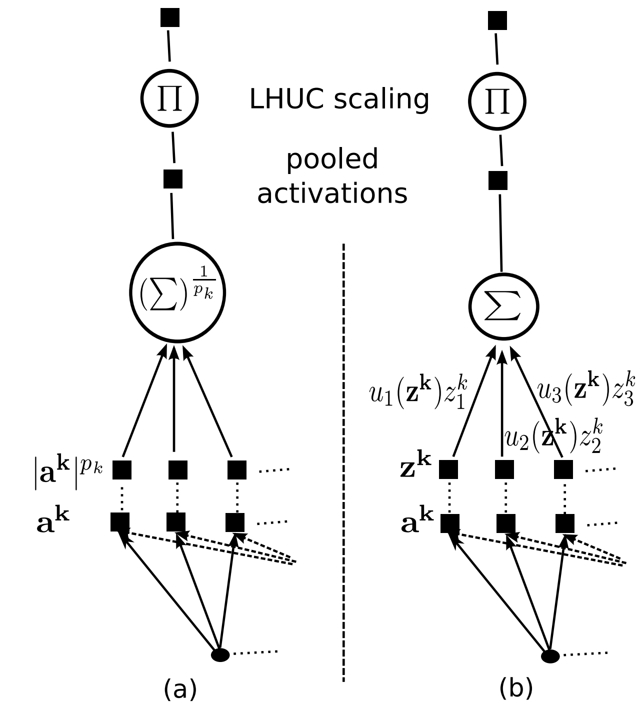


Fig. 1. Illustration of Diff-L_p (a) and Diff-Gauss (b) pooling operators. See Sections III-A and III-B for further details and explanations of the symbols. LHUC scaling follows [68] and is used only during adaptation.

necessary property of the norm), during optimisation p_k is re-parametrised as $p_k = \zeta(\rho_k) = \max(1, \rho_k)$, where ρ_k is the actual learned parameter. For the case when $p_k = \infty$ we obtain the max-pooling operator [55]:

$$||\mathbf{a}^k||_\infty = \max\left(\{|a_i^k|\}_{i \in R_k}\right). \quad (4)$$

Similarly, if $p_k = 1$ we obtain absolute average pooling (assuming the pool is normalised by K). We refer to this model as Diff-L_p, and it is parametrised by $\theta_{L_p} = \{\{\mathbf{W}^l, \mathbf{b}^l, \boldsymbol{\rho}^l\}_{l=1}^{L-1}, \mathbf{W}^L, \mathbf{b}^L\}$. Serement et al [65] investigated fixed-order L_p pooling for image classification, which was applied to speaker independent acoustic modelling [67]. Here we allow each L_p unit in the model to have a learnable order p [69], and we use the pooling parameters to perform model-based test-only acoustic adaptation.

B. Gaussian kernel (Diff-Gauss) pooling

The second pooling approach estimates the pooling coefficients using a Gaussian kernel. We generate the pooling inputs at each layer as:

$$\mathbf{z}^k = \left\{\eta_k \cdot \phi(\mathbf{w}_i^\top \mathbf{x} + b_i)\right\}_{i \in R_k} = \left\{\eta_k \cdot \phi(\mathbf{a}_i^k)\right\}_{i \in R_k}, \quad (5)$$

where ϕ is a non-linearity (tanh in this work) and $\mathbf{a}^k$ is a set of affine projections as before. A non-linearity is essential as otherwise (contrary to L_p pooling) we would produce a linear transformation through a linear combination of linear projections. η_k is the pool amplitude; this parameter is tied and learned per-pool as this was found to give similar results to per-unit amplitudes (but with fewer parameters), and better results compared to setting to a fixed value $\eta_k = 1.0$ [50].

Given the activation (5), the pooling operation is defined as a weighted average over a set R_k of hidden units, where the k-th pooling unit $f_G(\cdot; \boldsymbol{\vartheta}^k)$ is expressed as:

$$f_G\left(\mathbf{z}^k; \boldsymbol{\vartheta}^k\right) = \sum_{i \in R_k} u_i(\mathbf{z}^k; \boldsymbol{\vartheta}^k) z_i^k. \quad (6)$$

The pooling contributions $\mathbf{u}(\mathbf{z}^k; \boldsymbol{\vartheta}^k)$ are normalised to sum to one within each pooling region R_k (7) and each weight $u_i(z_i^k; \boldsymbol{\vartheta}^k)$ is coupled with the corresponding value of z_i^k by a Gaussian kernel (8) (one per pooling unit) parameterised by the mean and precision, $\boldsymbol{\vartheta}^k = \{\mu_k, \beta_k\}$:

$$u_i(\mathbf{z}^k; \boldsymbol{\vartheta}^k) = \frac{v(z_i^k; \boldsymbol{\vartheta}^k)}{\sum_{i' \in R_k} v(z_{i'}^k; \boldsymbol{\vartheta}^k)}, \quad (7)$$

$$v(z_i^k; \boldsymbol{\vartheta}^k) = \exp\left(-\frac{\beta_k}{2}\left(z_i^k - \mu_k\right)^2\right). \quad (8)$$

Similar to L_p-norm pooling, this formulation allows a generalised pooling to be learned – from average ($\beta \to 0$) to max ($\beta \to \infty$) – separately for each pooling unit $f_G(\mathbf{z}^k; \boldsymbol{\vartheta}^k)$ within a model. The Diff-Gauss model is thus parametrised by $\theta_G = \{\{\mathbf{W}^l, \mathbf{b}^l, \boldsymbol{\mu}^l, \boldsymbol{\beta}^l, \boldsymbol{\eta}^l\}_{l=1}^{L-1}, \mathbf{W}^L, \mathbf{b}^L\}$.

IV. Learning Differentiable Poolers

We optimise the acoustic model parameters by minimising the negative log probability of the target HMM tied state given the acoustic observations using gradient descent and error back-propagation [70]; the pooling parameters may be updated in a speaker-dependent manner, to adapt the acoustic model to unseen data. In this section we give the necessary partial derivatives for Diff-L_p and Diff-Gauss pooling.

A. Learning and adapting Diff-L_p pooling

In Diff-L_p pooling we learn p_k which we express in terms of ρ, $p_k = \zeta(\rho_k)$. Error back-propagation requires the partial derivative of the pooling region $f_{L_p}(\mathbf{a}^k; \rho_k)$ with respect to ρ_k, which is given as:

$$\frac{\partial f_{L_p}(\mathbf{a}^k; \rho_k)}{\partial \rho_k} = \left(\frac{\sum_{i \in R_k} \log(|a_i^k|) \cdot |a_i^k|^{p_k}}{p_k \sum_{i \in R_k} |a_i^k|^{p_k}} - \frac{\log \sum_{i \in R_k} |a_i^k|^{p_k}}{p_k^2}\right) \frac{\partial \zeta(\rho_k)}{\partial \rho_k} f_{L_p}(\mathbf{a}^k; \rho_k), \quad (9)$$

where $\partial\zeta(\rho_k)/\partial\rho_k = 1$ when $p_k > 1$ and 0 otherwise. The back-propagation through the norm itself is implemented as:

$$\frac{\partial f_{L_p}(\mathbf{a}^k; p_k)}{\partial \mathbf{a}^k} = \frac{\mathbf{a}^k \circ |\mathbf{a}^k|^{p_k - 2}}{\sum_{i \in R_k} |a_i^k|^{p_k}} \circ \mathbf{G}^k, \quad (10)$$

where $\circ$ represents the element-wise Hadamard product, and $\mathbf{G}^k$ is a vector of $f_{L_p}(\mathbf{a}^k; p_k)$ activations repeated K times, so the resulting operation can be fully vectorised:

$$\mathbf{G}^k = \left[f_{L_p}(\mathbf{a}^k; p_k)^1, \ldots, f_{L_p}(\mathbf{a}^k; p_k)^K\right]^\top. \quad (11)$$

Normalisation by K in (3) is optional (see also Section VII-A) and the partial derivatives in (9) and (10) hold for the un-normalised case also: the effect of this is taken into account in the forward activation $f_{L_p}(\mathbf{a}^k; p_k)$.

Since (9) and (10) are not continuous everywhere, they need to be stabilised when $\sum_{i \in R_k} |a_i^k|^{p_k} = 0$. When computing logarithm in the numerator of (9) it is also necessary to ensure that each $a_i^k > 0$. In practise, we threshold each element to have at least a value $\epsilon = 10^{-8}$ if $a_i^k < \epsilon$. Note, this numerical stabilisation of $\mathbf{a}^k$ only applies to L_p units, not `Diff-Gauss`.

B. Learning and adapting `Diff-Gauss` pooling regions

To learn the `Diff-Gauss` pooling parameters $\boldsymbol{\vartheta}^k = \{\mu^k, \beta^k\}$, we require the partial derivatives $\partial f_G(\mathbf{z}^k)/\partial \mu_k$ and $\partial f_G(\mathbf{z}^k)/\partial \beta_k$ to update pooling parameters, as well as $\partial f_G(\mathbf{z}^k)/\partial \mathbf{z}^k$ in order to back-propagate error signals to lower layers.

One can compute the partial derivative of (6) with respect to the input activations $\mathbf{z}^k$ as:

$$\frac{\partial f_G(\mathbf{z}^k)}{\partial \mathbf{z}^k} = \left[(\mathbf{z}^k)^\top \left(\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))\mathbf{J}_\mathbf{v}(\mathbf{z}^k)\right) + \mathbf{u}(\mathbf{z}^k)^\top\right]^\top, \tag{12}$$

where $\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))$ is the Jacobian representing the partial derivative $\partial u(\mathbf{z}^k)/\partial v(\mathbf{z}^k)$:

$$\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k)) = \frac{\partial \mathbf{u}(\mathbf{z}^k)}{\partial \mathbf{v}(\mathbf{z}^k)} = \begin{bmatrix} \frac{\partial u(z_1^k)}{\partial v(z_1^k)} & \cdots & \frac{\partial u(z_1^k)}{\partial v(z_K^k)} \\ \vdots & \ddots & \vdots \\ \frac{\partial u(z_K^k)}{\partial v(z_1^k)} & \cdots & \frac{\partial u(z_K^k)}{\partial v(z_K^k)} \end{bmatrix}, \tag{13}$$

whose elements can be computed as:

$$\frac{\partial u(z_i^k)}{\partial v(z_i^k)} = \left(\sum_{m \in R_k} v(z_m)\right)^{-1} \left(1 - u(z_i^k)\right), \tag{14}$$

$$\frac{\partial u(z_i^k)}{\partial v(z_{i'}^k)} = \left(\sum_{m \in R_k} v(z_m)\right)^{-1} \left(-u(z_i^k)\right). \tag{15}$$

Likewise, $\mathbf{J}_\mathbf{v}(\mathbf{z}^k)$ represents the Jacobian of the kernel function $v(\mathbf{z}^k)$ in (8) with respect to $\mathbf{z}^k$:

$$\mathbf{J}_\mathbf{v}(\mathbf{z}^k) = \frac{\partial \mathbf{v}(\mathbf{z}^k)}{\partial \mathbf{z}^k} = \begin{bmatrix} \frac{\partial v(z_1^k)}{\partial z_1^k} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \frac{\partial v(z_K^k)}{\partial z_K^k} \end{bmatrix}, \tag{16}$$

and the elements of $\mathbf{J}_\mathbf{v}(\mathbf{z}^k)$ can be computed as:

$$\frac{\partial v(z_i^k)}{\partial z_i^k} = -\beta_k (z_i^k - \mu_k) v(z_i^k). \tag{17}$$

Similarly, one can obtain the gradients with respect to the pooling parameters $\boldsymbol{\vartheta}^k$. In particular, for β_k, the gradient is:

$$\frac{\partial f_G(\mathbf{z}^k)}{\partial \beta_k} = \sum_{i \in R_k} \left[(\mathbf{z}^k)^\top \left(\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))\mathbf{J}_\mathbf{v}(\beta_k)\right)\right]_i, \tag{18}$$

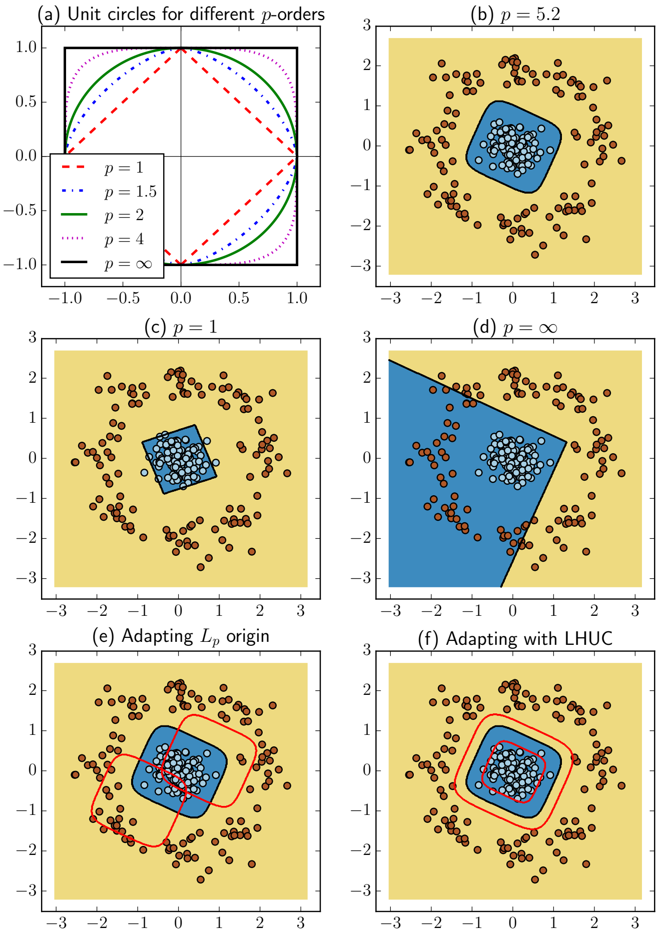
where $\mathbf{J}_\mathbf{v}(\beta_k) = \partial \mathbf{v}(\mathbf{z}^k)/\partial \beta_k$ and $\partial v(z_i^k)/\partial \beta_k$ is:

$$\frac{\partial v(z_i^k)}{\partial \beta_k} = -\frac{1}{2}\left(z_i^k - \mu_k\right)^2 v(z_i). \tag{19}$$

The corresponding gradient for $\partial f_G(\mathbf{z}^k)/\partial \mu_k$ is obtained below (20). Notice, that $\partial v(z_i^k)/\partial z_i^k$ (17) and $\partial v(z_i^k)/\partial \mu_k$ (21) are symmetric, hence $\mathbf{J}_\mathbf{v}(\mu_k) = -\mathbf{J}_\mathbf{v}(\mathbf{z}^k)$, and to compute $\partial f_G(\mathbf{z}^k)/\partial \mu_k$ one can reuse the $(\mathbf{z}^k)^\top \mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))\mathbf{J}_\mathbf{v}(\mathbf{z}^k)$ term in (12), as follows:

$$\begin{aligned} \frac{\partial f_G(\mathbf{z}^k)}{\partial \mu_k} &= \sum_{i \in R_k} \left[(\mathbf{z}^k)^\top \left(\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))\mathbf{J}_\mathbf{v}(\mu_k)\right)\right]_i \\ &= -\sum_{i \in R_k} \left[(\mathbf{z}^k)^\top \left(\mathbf{J}_\mathbf{u}(\mathbf{v}(\mathbf{z}^k))\mathbf{J}_\mathbf{v}(\mathbf{z}^k)\right)\right]_i, \end{aligned} \tag{20}$$

$$\frac{\partial v(z_i^k)}{\partial \mu_k} = -\frac{\partial v(z_i^k)}{\partial z_i^k} = \beta_k (z_i^k - \mu_k) v(z_i^k). \tag{21}$$


(a) Unit circles for different p-orders

(b) $p = 5.2$

(c) $p = 1$

(d) $p = \infty$

(e) Adapting L_p origin

(f) Adapting with LHUC

Fig. 2. Illustration of the representational efficiency and adaptation principles of an L_p unit. (a) Unit circles as obtained under different norm p-orders. (b) An example decision boundary for two-class toy data (red and blue dots). The `Diff-`L_p model is build out of one L_p unit (with $K = 2$ linear inputs) and is able to draw highly non-linear decision-regions. (c) The model from (b) with $p = 1.0$ and (d) $p = \infty$. Red contours of the bottom two plots illustrate (e) the effect of adaptation of the origin (biases) of the linear inputs $\mathbf{a}^k$ and (f) the effect of `LHUC` scaling. Further description in Section V. (Best viewed in colour.)

V. Representational Efficiency of Pooling Units

The aim of model-based DNN adaptation is to alter the learned speaker independent representation in order to improve the classification accuracy for data from a possibly mismatched test distribution. Owing to the highly distributed representations that are characteristic of DNNs, it is rarely clear which parameters should be adapted in order generalise well to a new speaker or acoustic condition.

Pooling enables decision boundaries to be altered, through the selection of relevant hidden features, while keeping the parameters of the feature extractors (the hidden units) fixed: this is similar to LHUC adaptation [68]. The pooling operators allow for a geometrical interpretation of the decision boundaries and how they will be affected by a constrained adaptation – the units within the pool are jointly optimised given the pooling parametrisation, and share some underlying relationship within the pool.

This is visualised for L_p units in Fig. 2. Fig. 2 (a) illustrates the unit circles obtained by solving $||\mathbf{a}^k||_p = d$ for different orders p, with $d = 1.0$ and a pool of $K = 2$ linear inputs $\mathbf{a}^k$. Such an L_p unit is capable of closed-region decision boundaries, illustrated in Fig. 2 (b). The distance threshold d is implicitly learned from data (through the $\mathbf{a}^k$ parameters given p), resulting in an efficient representation [67], [69] compared with representing such boundaries using sigmoid units or ReLUs, which would require more parameters. Figs. 2 (c) and (d) show how those boundaries are affected when $p = 1$ (average pooling) and $p = \infty$ (max pooling), while keeping $\mathbf{a}^k$ fixed. As shown in Section VII we found that updating p is an efficient and relatively low-dimensional way to adjust decision boundaries such that the the model's accuracy on the adaptation data distribution improves.

It is also possible to update the biases (Fig. 2 (e), red contours) and the LHUC amplitudes (Fig. 2 (f), red contours). We experimentally investigate how each approach impacts adaptation WER in Section VII-B. Although models implementing Diff-Gauss units are theoretically less efficient in terms of SI representations compared to L_p units, and comparable to standard fully-connected models, the pooling mechanism still allows for more efficient (in terms of number of SD parameters) speaker adaptation.

VI. Experimental setups

We have carried out experiments on three corpora: the TED talks corpus [71] following the IWSLT evaluation protocol (www.iwslt.org); the Switchboard corpus of conversational telephone speech [72] (ldc.upenn.edu) and the AMI meetings corpus [73], [74] (corpus.amiproject.org). Unless explicitly stated otherwise, our baseline models share similar structure across the tasks – DNNs with 6 hidden layers (2,048 units per layer) using a sigmoid non-linearity. The output softmax layer models the distribution of context-dependent clustered tied states [75]. The features are presented in 11 (±5) frame long context windows. All the adaptation experiments, if not stated otherwise, were performed unsupervised using adaptation targets obtained from first-pass speaker-independent decoding of the corresponding SI system.

TED: The training data consisted of 143 hours of speech (813 talks) and the systems follow our previously described recipe [8]. However, compared to our previous work [8], [11], [50], our systems here make use of more accurate language models developed for our IWSLT–2014 systems [76]: in particular, the final reported results use a 4-gram language model estimated from 751 million words. The baseline TED acoustic models were trained on unadapted PLP features with first and second order time derivatives. We present results on four IWSLT test sets: dev2010, tst2010, tst2011 and tst2013 containing 8, 11, 8, and 28 talks respectively.

AMI: We follow a Kaldi GMM recipe [77] and use the individual headset microphone (IHM) recordings. On this corpus, we train the acoustic models using 40 mel-filter-bank (FBANK) features. We decode with a pruned 3-gram language model estimated from 800k words of AMI training transcripts interpolated with an LM trained on Fisher conversational telephone speech transcripts (1M words) [78].

Switchboard (SWBD): We follow a Kaldi GMM recipe [79], [80][1], using Switchboard–1 Release 2 (LDC97S62). Our baseline unadapted acoustic models were trained on MFCC features, while the SAT trained fMLLR variants utilise the usual Kaldi feature preprocessing pipeline, which is MFCC+LDA/MLLT+fMLLR[2]. The results are reported on the full Hub5'00 set (LDC2002S09) – eval2000. eval2000 contains two types of data: Switchboard – which is better matched to the training data; and CallHome (CHE) English. Our reported results use 3-gram LMs estimated from the Switchboard and Fisher Corpus transcripts.

VII. Results

A. Baseline speaker independent models

The structures of the differentiable pooling models were selected such that the number of parameters was comparable to the corresponding baseline DNN models, described in detail in [68]. For the Diff-L_p and Diff-L_2 types, the resulting models utilised non-overlapping pooling regions of size $K = 5$, with 900 L_p-norm units per layer. The Diff-Gauss models had pool sizes set to $K = 3$ (this was found to work best in our previous work [50]) which (assuming non-overlapping regions) results in 1175 pooling units per layer.

Training speaker independent Diff-L_2 and Diff-L_p models: For both Diff-L_p and Diff-L_2 we trained with an initial learning rate of .008 (for MFCC, PLP, FBANK features) and .006 (for fMLLR features). The learning rate was adjusted using the newbob learning scheme [81] based on the validation frame error rate. We found that applying explicit pool normalisation (dividing by K in (3)) gives consistently higher error rates (typically an absolute increase of 0.3% WER): hence we used un-normalised L_p units in all experiments. We did not apply post-layer normalisation [67]. Instead, we use max-norm approach – after each update we scaled the columns (i.e. each a_i^k) of the fully connected weight matrices such that their L_2 norms were below a given threshold (set to 1.0 in this work) [82]. For Diff-L_p models we initialised $p = 2.0$. Those parameters were optimised on TED and directly applied without further tuning for the other two corpora. In this work we have focussed on adaptation; Zhang et

[1] To stay compatible with our previous adaptation work on Switchboard [45], [68] we are using the older set of Kaldi recipe scripts called s5b, and our baseline results are comparable with the corresponding baseline numbers previously reported. A newer set of improved scripts exists under s5c which, in comparison to s5b, offer about 1.5% absolute lower WER.

[2] MFCC-Mel-frequency Cepstral Coefficients, LDA - Linear Discriminant Analysis, MLLT - Maximum Likelihood Linear Transform

TABLE I
BASELINE WER(%) RESULTS ON SELECTED TEST SETS OF OUR BENCHMARK CORPORA

Model	TED `tst2010`	AMI `eval`	SWBD `eval2000`
DNN	15.0	29.1	22.1
`Diff-Gauss`	14.6	29.0	21.4
`Diff-`L_2	14.6	28.5	21.3
`Diff-`L_p	14.5	27.6	21.3

al [67] have reported further speaker independent experiments for fixed order L_p units.

Training speaker independent `Diff-Gauss` models: The initial learning rate was set to 0.08 (regardless of the feature type), again adjusted using `newbob`. Initial pooling parameters were sampled randomly from normal distribution: $\mu \sim \mathcal{N}(0,1)$ and $\beta \sim \mathcal{N}(1,0.5)$. Otherwise, the hyper-parameters were the same as for the baseline DNN models.

Baseline speaker independent results: Table I gives speaker independent results for each of the considered model types. The `Diff-Gauss` and `Diff-`L_2/`Diff-`L_p models have comparable WERs, with a small preference towards `Diff-`L_p in terms of the final WER on TED and AMI; all have lower average WER than the baseline DNN. The gap between the pooled models increases on AMI data where `Diff-`L_p has a substantially lower WER (3.2% relative) than the fixed order `Diff-`L_2 which is in turn has a lower WER than the other two models (`Diff-Gauss` and baseline DNN) by 2.1% relative.

Fig. 3 gives more insight into the `Diff-`L_p models by showing how the final distributions of the learned order p differ across AMI, TED and SWBD corpora. p deviates more from its initialisation in the lower layers of the model; there is also a difference across corpora. This follows the intuition of how a multi-layer network builds its representation: lower layers are more dependent on acoustic variabilities, normalising for such effects, and hence feature extractors may differ across datasets – in contrast to the upper layers which rely on features abstracted away from the acoustic data. For these corpora, the order p rarely exceeded 3, sometimes dropping below 2 – especially for layer 1 with SWBD data. However, most L_p units, especially in higher layers, tend to have $p \sim 2$. This corresponds to previous work [67] in which fixed $L_{p=2}$ units tended to obtain lower WER. A similar analysis of `Diff-Gauss` pooling does not show large data-dependent differences in the learned pooling parameters.

Training speed: Table II shows the average training speeds for each of the considered models. Training pooling units is significantly more expensive than training baseline DNN models. This is to be expected as the pooling operations cannot be easily and fully vectorised. In our implementation training the `Diff-Gauss` or `Diff-`L_p models is about 40% slower than training a baseline DNN. Not optimising p during training (9) decreases the gap to about 20% slower. This indicates that training using fixed L_2 units, and then adapting the order p in a speaker adaptive manner could make a good compromise.

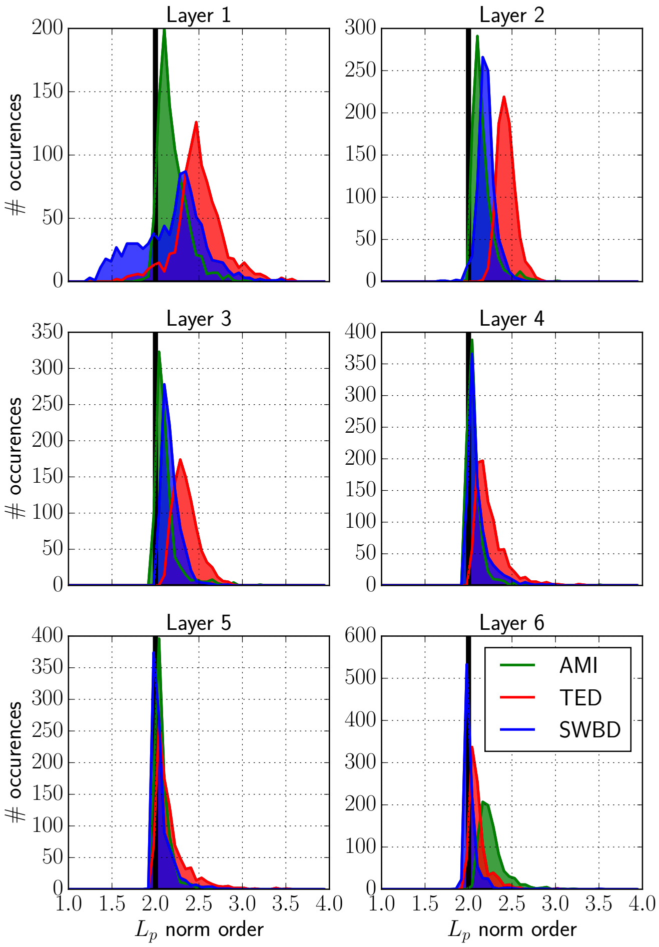


Fig. 3. L_p orders for the three corpora used in this work. Particular models share the same structure of hidden layers (the same number of L_p units per layer – 900), though both dimensionality of the output layers as well as the acoustic features used to train each model, are different. Vertical black line at 2 denotes an initial p setting of L_p units.

TABLE II
AVERAGE TRAINING SPEEDS [FRAMES/SECOND] AS OBTAINED FOR EACH MODEL TYPE ON SWBD DATA AND GTX980 GPGPU BOARDS.

DNN	`Diff-Gauss`	`Diff-`L_2	`Diff-`L_p
9k	5.2k	7.1k	5.4k

B. Adaptation experiments

We initially used the TED talks corpus to investigate how WERs are affected by adapting different layers in the model. The results indicated that adapting only the bottom layer brings the largest drop in WER; however, adapting more layers further improves the accuracy for both `Diff-`L_p and `Diff-Gauss` models (Fig. 4 (a)). Since obtaining the gradients for the pooling parameters at each layer is inexpensive compared to the overall back-propagation, and adapting bottom layer gives largest gains, in the remainder of this work we adapt all pooling units. Similar trends hold when pooling adaptation is combined with `LHUC` adaptation, which on `tst2010` improves the accuracies by 0.2-0.3% absolute.

Fig. 4 (b) shows WER vs. the number of adaptation iterations. The results indicate that one adaptation iteration is

TABLE III
WER(%) RESULTS FOR DIFFERENT SUBSETS OF ADAPTED PARAMETERS OF `Diff`-L_p MODEL ON TED (`tst2010`), AMI(`eval`) AND SWBD (`eval2000`) TEST SETS

Model	#SD Parameters	TED	AMI	SWBD
`Diff`-L_p	-	14.5	27.6	21.3
+ `LHUC`	$P(L-1)$	12.8	25.8	20.5
+ Update p	$P(L-1)$	12.5	25.8	20.1
++ Update b	$(P+PK)(L-1)$	12.3	25.5	20.5
++ `LHUC`	$2P(L-1)$	12.3	25.6	20.0

L - #layers, P - #pooling units in layer, K - pool size

TABLE IV
WER(%) RESULTS FOR DIFFERENT SUBSETS OF ADAPTED PARAMETERS OF `Diff-Gauss` MODEL ON TED (`tst2010`), AMI(`eval`) AND SWBD (`eval2000`) TEST-SETS.

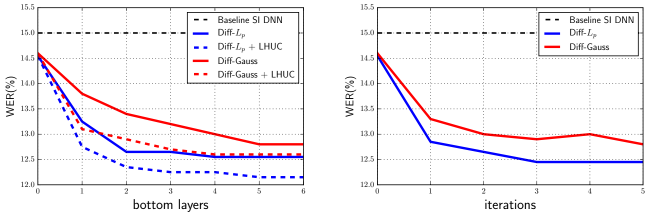
Model	#SD Parameters	TED	AMI	SWBD
`Diff-Gauss`	-	14.6	29.0	21.4
+ `LHUC`	$P(L-1)$	12.8	-	-
+ Update μ	$P(L-1)$	13.1	-	-
+ Update β	$P(L-1)$	13.1	-	-
+ Update η	$P(L-1)$	12.7	-	-
+ Update μ, β	$2P(L-1)$	12.8	27.3	20.7
++ `LHUC`	$3P(L-1)$	12.5	27.0	20.4
++ Update η	$3P(L-1)$	12.3	26.9	20.3

L - #layers, P - #pooling units in layer

sufficient and, more importantly, the model does not overfit when more iterations are used. This suggests that it is not necessary to regularise the model carefully (by Kullback-Leibler divergence [9], for instance) which is usually required when weights that directly transform the data are adapted. In the remainder, we adapt all models with a learning rate of 0.8 for three iterations (optimised on `dev2010`).

Table III shows the effect of adapting different pooling parameters (including `LHUC` amplitudes) for L_p units. Updating only p, rather than any other stand-alone pooling parameter, gives a lower WER than `LHUC` adaptation with the same number of parameters (cf Fig. 2); however, updating both brings further reductions in WER. Adapting the bias is more data-dependent with a substantial increase in WER for SWBD; this also significantly increases the number of adapted parameters. Hence we adapted either p alone, or p with `LHUC` in the remaining experiments

Table IV shows similar analysis but for `Diff-Gauss` model. For `Diff-Gauss`, it is beneficial to update both $\boldsymbol{\mu}$ and $\boldsymbol{\beta}$ (as in [50]), and `LHUC` was also found to be complementary. Notice, adapting with `LHUC` scalers is similar to altering η in eq. (5) (assuming η is tied per pool, as mentioned in Section III-B). As such, new parameters need not be introduced to adapt `Diff-Gauss` with `LHUC` as it is the case for `Diff`-L_p units. In fact, last two rows of Table IV show that jointly updating μ, β and η gives lower WER than updating μ, β and applying `LHUC` after pooling (see Fig. 1).



(a) (b)

Fig. 4. WER(%) on `tst2010` as a function of a) number of bottom layers adapted with pooling operators and (optional) `LHUC` transforms and b) number of adaptation iterations

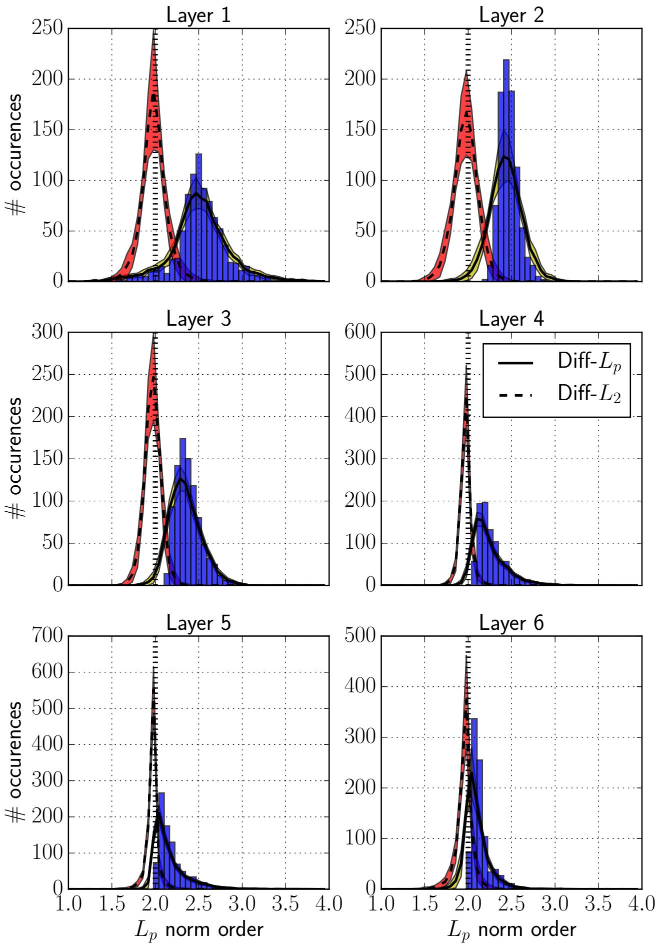


Fig. 5. Layer-wise histograms of learned L_p orders (speaker independent) on TED. The vertical line (dashed-black) at 2 is the initial value of p; the black solid line denotes the mean contour (± standard deviations in yellow) of the distribution of p obtained after adaptation to 28 speakers of `tst2013`. Likewise, the dashed black line is the mean of the adapted L_p orders (± standard deviations in red) starting from a fixed-order `Diff`-L_2 speaker independent model. (Best viewed in colour.)

Analysis of `Diff`-L_p: Fig. 5 shows how the distribution of p changes after the `Diff`-L_p model adapts to each of the 28 speakers of `tst2013`. We plot the speaker independent histograms as well as the contours of the mean bin frequencies for each layer. For the adapted models the distributions of p become less dispersed, especially in higher layers, which can be interpreted as shrinking the decision regions of particular L_p units (cf Fig. 2). This follows the intuition that speaker adaptation involves reducing the variability that needs to be modelled, in contrast to the speaker independent model.

Taking into account the increased training time of `Diff`-L_p models, one can also consider training fixed order `Diff`-L_2 [67], adapting p using (9). The results in Fig.

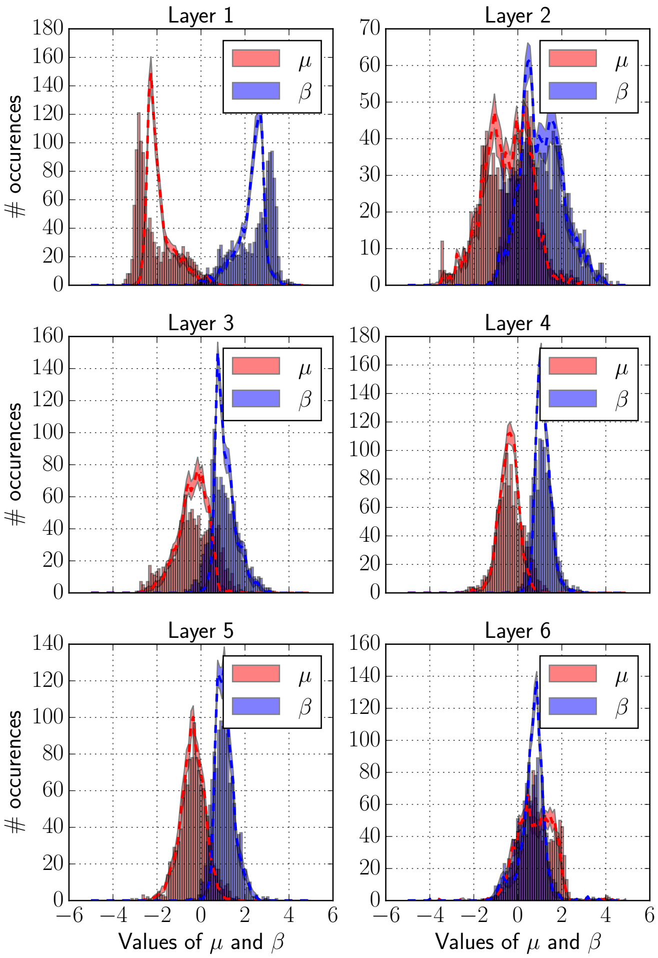


Fig. 6. Layer-wise histograms of learned `Diff-Gauss` pooling parameters $\{\mu, \beta\}$ during speaker independent training on TED. We also plot the altered mean contours (± standard deviation) of the adapted pooling parameters on 28 speakers of `tst2013`. (Best viewed in color.)

5, as well as later results, cover this scenario. The adapted `Diff-`L_2 models display a similar trend in the distribution of p to the `Diff-`L_p models.

Analysis of `Diff-Gauss`: We performed a similar investigation on the learned `Diff-Gauss` pooling parameters (Fig. 6). In the bottom layers they are characterised by a large negative means and positive precisions which has the effect of turning off many units. After adaptation, some of them become more active, which can be seen based on shifted distributions of adapted pooling parameters in Fig. 6. The adaptation with `Diff-Gauss` has a similar effect as the adaptation of slopes and amplitudes [44], [83], but adapts K times fewer parameters.

Amount of adaptation data and quality of targets: We investigated the effect of the amount of adaptation data by randomly selecting adaptation utterances from `tst2010` to give totals of 10s, 30s, 60s, 120s, 300s and more speaker-specific adaptation data per talker (Fig. 7 (a)). The WERs are an average over three independent runs, each sampling a different set of adaptation utterances (we did more passes in our previous work [11], [50], however, both `LHUC` and differentiable pooling operators were not sensitive to this aspect, resulting in small error bars between different results obtained with different random utterances). The `Diff-`L_p models offer lower WER and more rapid adaptation, with 10s of adaptation data resulting in a decrease in WER by 0.6% absolute (3.6% relative) which further increases up to 2.1% absolute (14.4% relative) when using all the speaker's data in an unsupervised manner. `Diff-Gauss` is comparable in terms of WER to a DNN adapted with `LHUC`. In addition, both methods are complementary to `LHUC` adaptation, and to feature-space adaptation with fMLLR (Tables VI and VII).

In order to demonstrate the modelling capacities of the different model-based adaptation techniques, we carried out a supervised adaptation (oracle) experiment in which the adaptation targets were obtained by aligning the audio data with reference transcripts (Fig. 7 (b)). We do not refine what the model knows about speech, nor the way it classifies it (the feature receptors and output layer are fixed during adaptation and remain speaker independent), but show that the re-composition and interpolation of these basis functions to approximate the unseen distribution of adaptation data is able to decrease the WER by 26.7% relative for `Diff-`L_p + `LHUC` scenario.

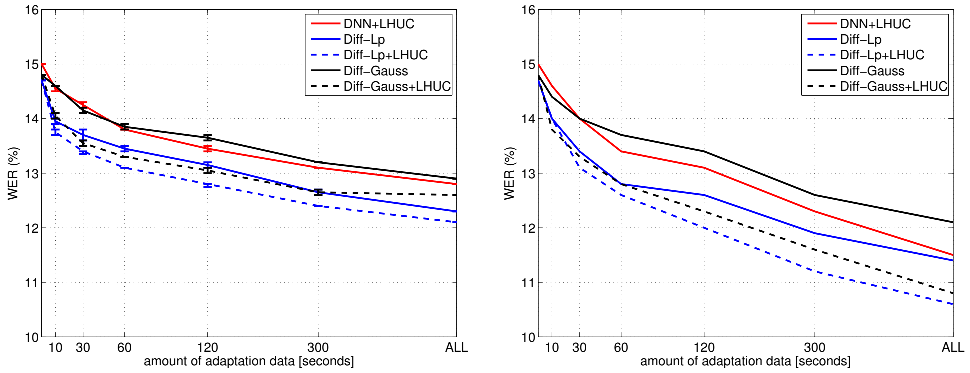
The methods are also not very sensitive to the quality of adaptation targets, they show very similar trends as `LHUC`, for which exact results for different qualities of adaptation targets resulting from re-scoring adaptation hypotheses with different language models were reported in [68].

Summary: Results for the proposed techniques are summarised in Tables V, VI, and VII for AMI, TED, and SWBD, respectively. The overall observed trends are as follows: (I) speaker independent pooling models return lower WERs than the baseline DNNs: `Diff-Gauss` $<$ `Diff-`$L_2 \leq$ `Diff-`L_p (although the last two seem to be data-dependent); (II) the pooling models (`Diff-Gauss`, `Diff-`L_2 and `Diff-`L_p) are complementary to both fMLLR and `LHUC` adaptation – as expected, the final gain depends on the degree of data mismatch; (III) one can effectively train speaker independent `Diff-`L_2 models and later alter p in a speaker dependent manner; (IV) the average relative improvement across all tasks with respect to baseline unadapted DNN models were 6.8% for `Diff-Gauss`, 9.1% for `Diff-`L_2 and 10.4% for `Diff-`L_p; and (V) when comparing `LHUC` adapted DNN to `LHUC` adapted differentiable pooling models, the relative reductions in WER for the pooling models were 2%, 3.4% and 4.8% for `Diff-Gauss`, `Diff-`L_2 and `Diff-`L_p, respectively.

VIII. Discussion and Conclusions

We have proposed the use of differentiable pooling operators with DNN acoustic models to perform unsupervised speaker adaptation. Differentiable pooling operators offer a relatively-low dimensional set of parameters which may be adapted in a speaker-dependent fashion.

We investigated the complementarity of differentiable pooling adaptation with two other approaches – model-based `LHUC` adaptation and feature-space fMLLR adaptation. We have not performed an explicit comparison with an i-vector approach to adaptation. However, some recent papers have compared i-vector adaptation with either `LHUC` and/or fMLLR on similar



(a) (b)

Fig. 7. WER(%) on tst2010 for different amounts of adaptation data with (a) unsupervised and (b) oracle adaptation targets.

TABLE V
WER(%) ON AMI - INDIVIDUAL HEADSET MICROPHONES AND AM TRAINED ON FBANK FEATURES

Model	dev	eval
DNN	26.8	29.1
+LHUC	25.6	27.1
Diff-Gauss	26.7	29.0
+ Update μ, β	26.0	27.3
++LHUC	25.7	27.0
Diff-L_2	26.1	28.5
+ Update p	25.5	26.9
++LHUC	25.3	26.7
Diff-L_p	25.4	27.6
+ Update p	24.7	25.8
++LHUC	24.7	25.6

TABLE VI
SUMMARY WER(%) RESULTS ON TED TEST SETS FROM IWSLT12 AND IWSLT13 EVALUATIONS.

Model	dev2010	tst2010	tst2011	tst2013
DNN	15.4	15.0	12.1	22.1
+LHUC	14.5	12.8	11.0	19.2
+fMLLR	14.5	12.9	10.9	20.8
++LHUC	14.1	11.8	10.3	18.4
Diff-Gauss	15.4	14.6	11.9	21.8
+ Update μ, β	14.5	12.8	11.2	19.5
++LHUC	14.1	12.5	10.8	18.7
+fMLLR	14.6	13.1	10.9	21.1
++ Update μ, β	14.3	12.4	10.7	19.4
+++LHUC	14.1	12.1	10.5	18.9
Diff-L_2	15.0	14.6	11.8	21.7
+ Update p	14.1	12.6	11.0	18.5
++LHUC	13.9	12.3	10.8	18.1
Diff-L_p	14.9	14.5	11.7	21.6
+ Update p	14.2	12.5	10.8	18.4
++LHUC	14.0	12.2	10.6	17.9
+fMLLR	14.0	12.5	10.6	20.3
++ Update p	13.7	11.5	10.0	18.0
+++LHUC	13.4	11.4	9.8	17.6

TABLE VII
SUMMARY WER(%) RESULTS ON SWITCHBOARD EVAL2000

	eval2000		
Model	SWB	CHE	TOTAL
Baseline models			
DNN	15.8	28.4	22.1
+LHUC	15.4	27.0	21.2
+fMLLR	14.3	26.1	20.3
++LHUC	14.2	25.6	19.9
Diff-Gauss models			
Diff-Gauss	15.1	27.8	21.4
+ Update μ, β	14.8	26.6	20.7
++LHUC	14.6	26.2	20.4
+fMLLR	14.4	26.1	20.3
++ Update μ, β	14.3	25.5	19.9
Diff-L_2 models			
Diff-L_2	14.9	28.0	21.3
+ Update p	14.2	26.0	20.1
++LHUC	14.2	25.9	20.1
+fMLLR	13.9	25.5	19.7
++ Update p	13.5	24.9	19.2
Diff-L_p models			
Diff-L_p	14.8	28.0	21.3
+ Update p	14.2	26.0	20.1
++LHUC	14.1	25.9	20.0
+fMLLR	13.7	25.3	19.5
++ Update p	13.5	24.6	19.0

data which enables us some make indirect comparisons. For example, Samarakoon and Sim [34] showed that speaker-adaptive training with i-vectors gives a comparable results to test-only LHUC using TED data, and Miao et. al [33] suggested that LHUC is better than a standard use of i-vectors (as in Saon et al. [29]) on TED data, with a more sophisticated i-vector post-processing needed to equal LHUC. Since the proposed Diff-L_p and Diff-Gauss techniques resulted in WERs that were at least as good as LHUC (and were found to be complementary to fMLLR) we conclude that the proposed pooling-based adaptation techniques are competitive.

In the future, one could investigate extending the proposed techniques to speaker adaptive training (SAT) [84], [85], for example in a similar spirit as proposed in the context of SAT-LHUC [45]. In addition it would be interesting to investigate the suitability of adapting pooling regions in the framework of sequence discriminative training [79], [86], [87]. Our experience of LHUC in this framework [68], together with the observation that the pooling models are not prone to over-fitting in the case of small amounts of adaptation data, suggests that adaptation based on differentiable pooling is a promising technique for sequence trained models.

ACKNOWLEDGEMENT

The NST research data collection may be accessed at http://datashare.is.ed.ac.uk/handle/10283/786. This research utilised a K40 GPGPU board donated by NVIDA Corporation. The authors would like to thank the reviewers for insightful comments that helped to improve the manuscript.

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," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, Nov 2012.

[2] D Yu, M Seltzer, J Li, J-T Huang, and F Seide, "Feature learning in deep neural networks - studies on speech recognition," in *Proc. ICLR*, 2013.

[3] 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.

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

[5] J Trmal, J Zelinka, and L Müller, "On speaker adaptive training of artificial neural networks," in *Proc. Interspeech*, 2010.

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

[7] 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, pp. 366–369.

[8] P Swietojanski, A Ghoshal, and S Renals, "Revisiting hybrid and GMM-HMM system combination techniques," in *Proc. IEEE ICASSP*, 2013, pp. 6744–6748.

[9] 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.

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

[11] P Swietojanski and S Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Proc. IEEE SLT*, 2014.

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

[13] SJ Young and PC Woodland, "State clustering in hidden Markov model-based continuous speech recognition," *Computer Speech and Language*, vol. 8, no. 4, pp. 369–383, 1994.

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

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

[16] JS Bridle, "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition," in *Neurocomputing*, F Fogelman Soulié and J Hérault, Eds., pp. 227–236. Springer, 1990.

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

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

[19] F Grezl, M Karafiat, S Kontar, and J Cernocky, "Probabilistic and bottle-neck features for LVCSR of meetings," in *Proc. IEEE ICASSP*, 2007, pp. IV–757–IV–760.

[20] A Mohamed, TN Sainath, G Dahl, B Ramabhadran, GE Hinton, and MA Picheny, "Deep belief networks using discriminative features for phone recognition," in *Proc. IEEE ICASSP*, 2011, pp. 5060–5063.

[21] 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 Transactions on Audio, Speech and Language Processing*, vol. 20, pp. 486–498, 2012.

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

[23] TN Sainath, A Mohamed, B Kingsbury, and B Ramabhadran, "Deep convolutional neural networks for lvcsr," in *Proc. IEEE ICASSP*, 2013, pp. 8614–8618.

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

[25] T Yoshioka, A Ragni, and MJF Gales, "Investigation of unsupervised adaptation of dnn acoustic models with filter bank input," in *Proc. IEEE ICASSP*, 2014, pp. 6344–6348.

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

[27] 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.

[28] M Karafiat, L Burget, P Matejka, O Glembek, and J Cernozky, "I-vector-based discriminative adaptation for automatic speech recognition," in *Proc. IEEE ASRU*, 2011.

[29] 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.

[30] A Senior and I Lopez-Moreno, "Improving DNN speaker independence with i-vector inputs.," in *Proc. IEEE ICASSP*, 2014, pp. 225–229.

[31] 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.

[32] P Karanasou, Y Wang, MJF Gales, and PC Woodland, "Adaptation of deep neural network acoustic models using factorised i-vectors," in *Proc. ICSA Interspeech*, 2014, pp. 2180–2184.

[33] Y Miao, H Zhang, and F Metze, "Speaker adaptive training of deep neural network acoustic models using i-vectors," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 11, pp. 1938–1949, Nov 2015.

[34] L Samarakoon and K C Sim, "On combining i-vectors and discriminative adaptation methods for unsupervised speaker normalization in dnn acoustic models," in *Proc. IEEE ICASSP*, 2016, pp. 5275–5279.

[35] 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, pp. 5542–5546.

[36] JS Bridle and S Cox, "Recnorm: Simultaneous normalisation and classification applied to speech recognition," in *Advances in Neural Information and Processing Systems*, 1990.

[37] 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. IEEE ICASSP*, 2013, pp. 4277–4280.

[38] S Xue, O Abdel-Hamid, J Hui, L Dai, and Q Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1713–1725, Dec 2014.

[39] S Kundu, G Mantena, Y Qian, T Tan, M Delcroix, and KC Sim, "Joint acoustic factor learning for robust deep neural network based automatic speech recognition," in *Proc. IEEE ICASSP*, March 2016, pp. 5025–5029.

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

[41] Y Huang and Y Gong, "Regularized sequence-level deep neural network model adaptation," in *Proc. ICSA Interspeech*, 2015, pp. 1081–1085.

[42] T Ochiai, S Matsuda, X Lu, C Hori, and S Katagiri, "Speaker adaptive training using deep neural networks," in *Proc. IEEE ICASSP*, 2014, pp. 6349–6353.

[43] 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.

[44] Y Zhao, J Li, J Xue, and Y Gong, "Investigating online low-footprint speaker adaptation using generalized linear regression and click-through data," in *Proc. IEEE ICASSP*, 2015, pp. 4310–4314.

[45] P Swietojanski and S Renals, "SAT-LHUC: Speaker adaptive training for learning hidden unit contributions," in *Proc. IEEE ICASSP*, 2016, pp. 5010–5014.

[46] Z Huang, S M Siniscalchi, I-F Chen, J Wu, and C-H Lee, "Maximum a-posteriori adaptation of network parameters in deep models," *arXiv preprint arXiv:1503.02108*, 2015.

[47] Z Huang, J Li, SM Siniscalchi, I-F Chen, J Wu, and C-H Lee, "Rapid adaptation for deep neural networks through multi-task learning," in *Proc. ICSA Interspeech*, 2015, pp. 3625–3629.

[48] P Swietojanski, P Bell, and S Renals, "Structured output layer with auxiliary targets for context-dependent acoustic modelling," in *Proc. ICSA Interspeech*, 2015, pp. 3605–3609.

[49] R Price, K Iso, and K Shinoda, "Speaker adaptation of deep neural networks using a hierarchy of output layers," in *Proc. IEEE SLT*, 2014.

[50] P Swietojanski and S Renals, "Differentiable pooling for unsupervised speaker adaptation," in *Proc. IEEE ICASSP*, 2015, pp. 4305–4309.

[51] D Hubel and T Wiesel, "Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex," *Journal of Physiology*, vol. 160, pp. 106–154, 1962.

[52] K Fukushima and S Miyake, "Neocognitron: A new algoriothm for pattern recognition tolerant of deformations," *Pattern Recognition*, vol. 15, pp. 455–469, 1982.

[53] Y LeCun, B Boser, JS Denker, D Henderson, RE Howard, W Hubbard, and LD Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, pp. 541–551, 1989.

[54] Y LeCun, L Bottou, Y Bengio, and P Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

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

[56] MA Ranzato, FJ Huang, Y-L Boureau, and Y LeCun, "Unsupervised learning of invariant feature hierarchies with applications to object recognition," in *Proc. IEEE CVPR*, 2007.

[57] Y-L Boureau, J Ponce, and Y LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proc. ICML*, 2010.

[58] IJ Goodfellow, D Warde-Farley, M Mirza, A Courville, and Y Bengio, "Maxout networks," in *Proc. ICML*, 2013, pp. 1319–1327.

[59] Y Miao, F Metze, and S Rawat, "Deep maxout networks for low-resource speech recognition," in *Proc. IEEE ASRU*, 2013.

[60] M Cai, Y Shi, and J Liu, "Deep maxout neural networks for speech recognition," in *Proc. IEEE ASRU*, 2013, pp. 291–296.

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

[62] S Renals and P Swietojanski, "Neural networks for distant speech recognition," in *Proc. HSCMA*, 2014.

[63] L Toth, "Convolutional deep maxout networks for phone recognition," in *Proc. ICSA Interspeech*, 2014.

[64] MD Zeiler and R Fergus, "Differentiable pooling for hierarchical feature learning," *CoRR*, vol. abs/1207.0151, 2012.

[65] P Sermanet, S Chintala, and Y LeCun, "Convolutional neural networks applied to house numbers digit classification," *CoRR*, vol. abs/1204.3968, 2012.

[66] TN Sainath, B Kingsbury, A Mohamed, GE Dahl, G Saon, H Soltau, T Beran, AY Aravkin, and B Ramabhadran, "Improvements to deep convolutional neural networks for LVCSR," in *Proc. IEEE ASRU*, 2013, pp. 315–320.

[67] X Zhang, J Trmal, D Povey, and S Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *Proc. IEEE ICASSP*, 2014.

[68] P Swietojanski, J Li, and S Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 8, pp. 1450–1463, 2016.

[69] C Gülçehre, K Cho, R Pascanu, and Y Bengio, "Learned-norm pooling for deep feedforward and recurrent neural networks," in *Proc. ECML and KDD*. 2014, pp. 530–546, Springer-Verlag.

[70] DE Rumelhart, GE Hinton, and RJ Williams, "Learning internal representations by error-propagation," in *Parallel Distributed Processing*, vol. 1, pp. 318–362. MIT Press, 1986.

[71] M Cettolo, C Girardi, and M Federico, "Wit^3: Web inventory of transcribed and translated talks," in *Proc. EAMT*, 2012, pp. 261–268.

[72] JJ Godfrey, EC Holliman, and J McDaniel, "SWITCHBOARD: Telephone speech corpus for research and development," in *Proc. IEEE ICASSP*. IEEE, 1992, pp. 517–520.

[73] J Carletta, "Unleashing the killer corpus: Experiences in creating the multi-everything AMI meeting corpus.," *Language Resources and Evaluation*, vol. 41, no. 2, pp. 181–190, 2007.

[74] S Renals, T Hain, and H Bourlard, "Recognition and understanding of meetings: The AMI and AMIDA projects," in *Proc. IEEE ASRU*, Kyoto, 12 2007, IDIAP-RR 07-46.

[75] GE Dahl, D Yu, L Deng, and A Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Transaction on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[76] P Bell, P Swietojanski, J Driesen, M Sinclair, F McInnes, and S Renals, "The UEDIN system for the IWSLT 2014 evaluation," in *Proc. IWSLT*, 2014, pp. 26–33.

[77] P Swietojanski, A Ghoshal, and S Renals, "Hybrid acoustic models for distant and multichannel large vocabulary speech recognition," in *Proc. IEEE ASRU*, 2013.

[78] C Cieri and D Millerand K Walker, "The Fisher corpus: a resource for the next generations of speech-to-text," in *Proc LREC*, 2004.

[79] K Vesely, A Ghoshal, L Burget, and D Povey, "Sequence-discriminative training of deep neural networks," in *Proc. ICSA Interspeech*, 2013, pp. 2345–2349.

[80] 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.

[81] S Renals, N Morgan, M Cohen, and H Franco, "Connectionist probability estimation in the DECIPHER speech recognition system," in *Proc. IEEE ICASSP*, 1992.

[82] N Srivastava, G Hinton, A Krizhevsky, I Sutskever, and R Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[83] C Zhang and PC Woodland, "Parameterised sigmoid and ReLU hidden activation functions for DNN acoustic modelling," in *Proc. ICSA Interspeech*, 2015, pp. 3224–3228.

[84] T Anastasakos, J McDonough, R Schwartz, and J Makhoul, "A compact model for speaker-adaptive training," in *Proc. ICSLP*, 1996, pp. 1137–1140.

[85] MJF Gales, "Cluster adaptive training of hidden markov models," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, no. 4, pp. 417–428, 2000.

[86] D Povey, *Discriminative training for large vocabulary speech recognition*, Ph.D. thesis, University of Cambridge, 2003.

[87] B Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Proc. IEEE ICASSP*, 2009, pp. 3761–3764.

Pawel Swietojanski received the M.Sc. degree from the AGH University of Science and Technology, Cracow, Poland, and is currently working toward the Ph.D. degree in informatics at the Centre for Speech Technology Research, School of Informatics, University of Edinburgh, U.K. His main research interests include machine learning and its applications to speech processing, with a particular focus on learning representations for acoustic modelling in speech recognition.

Steve Renals (M'91 — SM'11 – F'14) received the B.Sc. degree from the University of Shefeld, Shefeld, U.K., and the M.Sc. and Ph.D. degrees from Edinburgh. He is a Professor of speech technology at the University of Edinburgh, Edinburgh, U.K. He has previously had positions at ICSI Berkeley, the University of Cambridge, and the University of Shefeld. He is a Senior Area Editor of IEEE/ACM Transactions on Audio, Speech and Language Processing.