INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Factorised representations for neural network adaptation to diverse acoustic environments

Joachim Fainberg, Steve Renals, Peter Bell

The Centre for Speech Technology Research, University of Edinburgh, United Kingdom
{j.fainberg, s.renals, peter.bell}@ed.ac.uk

Abstract

Adapting acoustic models jointly to both speaker and environment has been shown to be effective. In many realistic scenarios, however, either the speaker or environment at test time might be unknown, or there may be insufficient data to learn a joint transform. Generating independent speaker and environment transforms improves the match of an acoustic model to unseen combinations. Using i-vectors, we demonstrate that it is possible to factorise speaker or environment information using multi-condition training with neural networks. Specifically, we extract bottleneck features from networks trained to classify either speakers or environments. We perform experiments on the Wall Street Journal corpus combined with environment noise from the Diverse Environments Multichannel Acoustic Noise Database. Using the factorised i-vectors we show improvements in word error rates on perturbed versions of the eval92 and dev93 test sets, both when one factor is missing and when the factors are seen but not in the desired combination.
Index Terms: speech recognition, adaptation, acoustic factorisation, i-vectors, deep neural networks

1. Introduction

Acoustic mismatch between training and test conditions may significantly affect acoustic models for speech recognition. Speaker adaptation has proven particularly effective [1, 2, 3, 4]. There are, however, other acoustic factors that affect the speech signal in addition to speakers. Modelling these factors, such as environments, may reduce acoustic mismatch and can yield additional reductions in Word Error Rates (WERs) [1, 5, 6, 7]. Particularly, learning a single transform for a joint combination of factors, such as a speaker and an environment combination, can reduce WERs over speaker adaptation alone [1]. This, however, requires sufficient adaptation data for each combination of factors. Further, it does not make efficient use of existing information, for example if a speaker is in an alternate combination of factors. When using transforms estimated in mismatched conditions, performance typically degrades [5].

A number of studies have investigated methods to adapt to each factor with separate transforms (e.g. [5, 6, 8, 9, 10]). For example Swietojanski et al. [1] interpolated transforms obtained independently using Learning Hidden Unit Contributions (LHUC). Seltzer and Acero [6] cascaded speaker and environment transforms using Constrained Maximum Likelihood Linear Regression (CMLLR). If the adaptation transforms for each factor are independent, then they can be combined in novel combinations not seen in the data, and existing transforms may be reused and recombined. Independence may occur implicitly when estimating the adaptation transforms if the nuisance factor is evenly distributed within the data. This is, however, an assumption not always found in real data and explicit approaches to obtaining independent transforms are often required. One explicit approach is making use of the observation that, if speaker and environment transformations are independent, then their first derivatives should be zero, which has been shown using constrained optimisation for cluster adaptive training [9] and i-vectors [5]. Karanasou et al. [5] obtained 10% relative reductions in WER using factorised i-vectors compared to normal speaker i-vectors on a perturbed set of Wall Street Journal (WSJ) [11] utterances. A somewhat similar approach was taken by Seo et al. [10], to obtain independent transforms by projecting adaptation transforms onto orthogonal, factor-dependent subspaces, resulting in 7.5% reduction in WER adapting to speakers using environment-independent transforms on Aurora4 [12].

There is also work that aims to normalise speaker information without explicitly referring to other factors. For example, Samarakoon and Sim [13] proposed to learn an input-dependent feature transformation using the i-vectors, by placing the i-vector in a diagonal matrix and embedding this matrix in a larger weight matrix. This may help normalise the speaker information and produced 3.5% relative WER reductions on top of normal (bias-only) i-vector adaptation. A related approach [14] models factors as additional inputs to the softmax layer of a neural network acoustic model, where each input has its own weight matrix dependent on a particular factor as input. Using a noise factor estimated using frames from the current utterance, roughly 10% relative improvements in WER were obtained using 20 adaptation utterances.

In this paper we investigate the use of neural networks to factorise adaptation transforms. Specifically, we extract bottleneck features from networks trained to classify speakers or environments given speaker or environment i-vectors, respectively (Section 2). The overall goal of the paper is similar to that of i-vector factorisation [5]. A key difference is that we experiment with factorising i-vectors after, rather than during, extraction. This means that it is possible to apply the technique to other feature representations as well. We show results on the WSJ corpus perturbed using the Diverse Environments Multichannel Acoustic Noise Database (DEMAND) [15]. To avoid implicit factorisation during i-vector extraction at test time, we explicitly avoid balancing environments across speakers and speakers across environments (Section 3). We first show that environment information in the factorised speaker i-vectors is significantly reduced, while maintaining speaker information, and similarly with speaker information in environment i-vectors (Section 4.1). We then show improvements when either speaker or environment information is absent at test time (Section 4.2). Finally, we experiment with the situation in which we reuse speaker and environment transforms which has been estimated in mismatched conditions and there is no adaptation data in the given combination (Section 4.3). For this experiment, factorising the i-vectors seems particularly important.

http://dx.doi.org/10.21437/Interspeech.2017-1365

2. Methods

i-vectors are fixed-dimensional representations, representing the coordinates, $\boldsymbol{\lambda}$, of a total variability subspace (or loading matrix) $\boldsymbol{M}$. In i-vector estimation we assume that the difference between the means of a Gaussian Mixture Model (GMM) trained on all the data, $\boldsymbol{\mu}_0$ and speaker-specific means, $\boldsymbol{\mu}^s$, is the matrix-vector product of the total variability matrix and the respective i-vector:

$$\boldsymbol{\mu}^s = \boldsymbol{\mu}_0 + \boldsymbol{M}\boldsymbol{\lambda}^s. \tag{1}$$

For more details on i-vectors we refer to Saon et al. [4].

For adaptation, the i-vectors are concatenated with the acoustic features, $\boldsymbol{x}$. This produces bias adaptation of the first hidden layer, $\boldsymbol{h}$:

$$\boldsymbol{h} = \sigma(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} + \boldsymbol{A}\boldsymbol{\lambda}), \tag{2}$$

where $\sigma(\cdot)$ denotes a nonlinearity, $\boldsymbol{W}$ and $\boldsymbol{b}$ are the corresponding weight matrix and bias, and $\boldsymbol{A}\boldsymbol{\lambda}$ is the bias contribution from the i-vector with weight matrix $\boldsymbol{A}$. In the case where we include both speaker and environment i-vectors there will be a second i-vector dependent bias term.

To factorise the i-vectors we experiment with multi-condition neural networks. The notion we test is that by learning to classify one factor, other nuisance factors are implicitly normalised out in the hidden representations since they are not relevant to the classification task. Specifically, we experiment with feedforward networks with as inputs either speaker or environment i-vectors and as outputs the corresponding speaker or environment classes, respectively. The extracted bottleneck features, `bn-iv`, should then mostly embody information about a particular speaker or a particular environment. These are concatenated with the acoustic features and used to train the neural network acoustic model. At test time we factorise the normal i-vectors, `iv`, by passing them through the existing networks. A diagram of the approach showing the use of either normal i-vectors or the bottleneck features is shown in Figure 1.

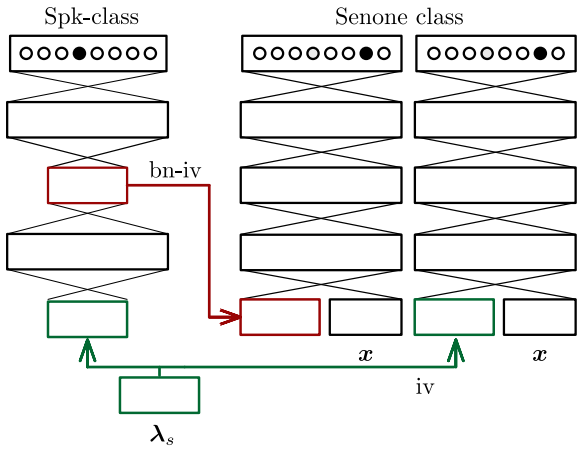

Figure 1: *Diagram of the setup shown for speaker i-vectors. The original i-vectors (`iv`) are either concatenated directly with the acoustic features, $\boldsymbol{x}$ (right), or first passed through a network that has been trained to classify speakers (left). The resulting bottleneck features (`bn-iv`) are then used in place of the normal i-vectors in another acoustic model (middle).*

3. Experimental setup

We performed experiments using the Wall Street Journal (WSJ) corpus [11] containing 282 speakers. As noise sources we used the Diverse Environments Multichannel Acoustic Noise Database (DEMAND) [15]. DEMAND provides 18 recordings of environments ranging from residential environments to offices and transportation (Table 1). Each recording was made with a 16 microphone array at 48kHz. To add noise to the single channel speech we have arbitrarily selected channel 1 of the array. We set the Signal to Noise Ratio (SNR) for each combination to 0 so that the results are more easily interpretable. Note that we do not include any clean examples in the perturbed dataset, though some environments are more noisy than others.

Table 1: *List of environments in DEMAND.*

Category	Environments
Domestic	Kitchen, Living Room, Washing
Nature	Field, Park, River
Office	Hallway, Meeting, Office
Public	Cafeteria, Restaurant, Station
Street	Cafe, Public square, Traffic
Transportation	Bus, Car, Metro

We randomly selected environments for each training utterance. For test utterances we applied the environments to utterances in an unbalanced manner. Otherwise we might learn implicit orthogonality between the factors, since for each speaker there would have been an even distribution of environments. Instead, we ensure a correlated set of environments per speaker, s, by for each utterance, u, sampling an environment k_u with respect to a categorical distribution on a probability vector $\mathbf{p}_s$:

$$k_u \sim \text{Cat}(\mathbf{p}_s), \tag{3}$$

where $\mathbf{p}_s \in \mathbb{R}^n$ and n is the number of environments. $\mathbf{p}_s$ is sampled from a symmetric Dirichlet distribution with concentration parameter α:

$$\mathbf{p}_s \sim \text{Dir}(\alpha). \tag{4}$$

This procedure ensures that speakers and environments are not independent, i.e.

$$p(s|n) \neq p(s), \quad p(n|s) \neq p(n) \qquad \forall s, n, \tag{5}$$

such that both speaker and environment i-vectors do not see an even balance of the other factor.

For $\alpha < 1.0$, the distributions will be highly peaked, while $\alpha = 1$ gives a flat Dirichlet distribution, effectively an 18-dimensional uniform simplex. In the limit where $\alpha \to \infty$, each distribution will be near-uniform. We chose $\alpha = 0.75$ as a reasonable compromise, i.e. each speaker is seen in several environments, but the distributions are clearly non-uniform.

The i-vectors are obtained as typical in Kaldi (e.g. [16]), with online and offline extraction for training and test data, respectively. Specifically, to obtain a suitable variety of i-vectors during training, we split each speaker (or environment) into sub-speakers with a maximum of two utterances for each sub-speaker. i-vectors were then extracted in an online fashion using only frames prior to the current frame within a sub-speaker. For

our setup this means that the training time i-vectors will effectively represent joint speaker-environment transforms. During decoding we estimated the i-vectors in an offline fashion for higher quality i-vectors. Normally this is split into subspeakers with 60 seconds minimum per speaker, but for more easily interpretable results we extracted a single i-vector across all the data for a given speaker or environment.

In the final experiment (Section 4.3) we address the situation where speaker and environment i-vectors are extracted in mismatched conditions, e.g. when adaptation data is unavailable. Specifically, we consider i-vectors extracted for a speaker in a single mismatched environment, or vice versa. To obtain results that are not biased towards particular environments we use test sets with an even balance of environments. The procedure for i-vector extraction is then as follows: For each possible speaker and environment combination, $(s \in S, n \in N_s)$, where N_s denotes the environments that have occurred with speaker s, we hold out each pair in turn. The speaker i-vector for the held-out pair is then determined by extracting an i-vector from the speaker seen in a different environment, e.g. $\boldsymbol{\lambda}_{s,n'}$ where n' is chosen such that $n' \in N_s$ and $n' \neq n$. Similarly, for the environment vector, we extract an i-vector $\boldsymbol{\lambda}_{s',n}$, where s' is chosen such that $s' \in S_n$ and $s' \neq s$. The i-vectors will now also contain information from s' and n'. The idea is then to factorise these vectors by generating bottleneck features using the multicondition networks, denoted by $f_{\{\texttt{spk},\texttt{env}\}}(\cdot)$:

$$\begin{aligned} f_{\texttt{spk}}(\boldsymbol{\lambda}_{s,n'}) &\approx \boldsymbol{\lambda}_s \\ f_{\texttt{env}}(\boldsymbol{\lambda}_{s',n}) &\approx \boldsymbol{\lambda}_n \end{aligned} \tag{6}$$

We trained Hidden Markov Model (HMM) - Gaussian Mixture Model (GMM) acoustic models with the Kaldi toolkit [17] and hybrid neural (NN) network models with `nnet3` package, using the standard WSJ recipe. Specifically, we trained monophone and triphone models on top of 13 mel-frequency cepstral coefficients (MFCCs) with delta and double deltas. We then trained on 40-dimensional features transformed with Linear Discriminant Analysis (LDA) and Maximum Likelihood Linear Transform (MLLT). Finally, we carried out speaker adaptive training with CMLLR.

We trained 6-layer time-delay feedforward neural networks [16] with p-norm activations [18] ($p = 2$) and input and output dimensions set to 2000 and 250, respectively. The splice indices were `-4,-3,-2,-1,0,1,2,3,4 0 -2,2 0 -4,4 0`. We trained with an initial learning rate set to 0.005, which was reduced exponentially to a tenth of the original rate over 8 epochs. The i-vectors were concatenated with the features for each frame using an i-vector period of 10.

For the bottleneck features we trained independent multicondition networks for speaker and environment classification, where each network had 282 or 18 output classes, respectively. The networks have three 500-dimensional layers with the exception of middle bottleneck layers the size of the original i-vectors (i.e. the input dimension). We used ReLU activations [19] and trained with RMSProp using the Keras deep learning toolkit [20]. We held out a random subset of 10% as validation data and used early stopping with a patience of 2. After training, we passed the i-vectors through the network to extract bottleneck features. We then trained acoustic models as above using the bottleneck features in place of the i-vectors.

4. Results

Table 2 shows baseline results training with clean data and testing on clean and perturbed test sets. Unsurprisingly, when the training and test conditions are not matched, the WERs are significantly higher. Training on matched, perturbed data significantly reduces errors on the perturbed test sets, as shown in Table 3. This is particularly evident for the neural network models, where the WERs on the perturbed sets drop by up to 62% relative, while increasing WERs on the non-perturbed sets by about 10%, or 1% absolute.

Table 2: *Baseline results WER (%) from models trained on WSJ. Test sets postfixed with* d *are noisy versions.*

Model	dev93	eval92	dev93d	eval92d
GMM	12.06	7.76	49.25	42.96
NN	10.36	6.72	48.17	41.52

Table 3: *Baseline results WER (%) from models trained on WSJ+DEMAND. Test sets postfixed with* d *are noisy versions.*

Model	dev93	eval92	dev93d	eval92d
GMM	13.52	9.50	32.77	25.09
NN	11.38	7.81	22.70	15.95

4.1. Multi-condition training

To confirm that the respective conditions are indeed factored out, we performed classification experiments on the original i-vectors and the (factorised) bottleneck features. These used networks with the same architectures and training procedure as above, but with 500 units in each hidden layer. Table 4 shows classification accuracies on held-out data. As shown in the first two rows, the original i-vectors (iv) contain a large amount of information about the opposite classes. In contrast, when training on top of the extracted bottleneck features (bn-iv), we observe large drops in accuracies. The differences in accuracies between the speaker and environment vectors may be due to the large difference in the number of speakers and environments (282 and 18, respectively). Crucially, the factorised representations still classify their respective classes with high accuracies. Figure 2 visualises the effect on the speaker i-vectors using t-SNE [21]. The original speaker i-vectors cluster into environments, whereas there is no evident clustering with the bottleneck features.

Table 4: *Classification validation accuracy (%) from 100 dimensional vectors to speaker or environment categories on a held-out evaluation set.*

Features	Spk-class	Env-class
iv spk	85.9	97.4
iv env	69.6	99.2
bn-iv spk	84.3	**41.2**
bn-iv env	**6.4**	99.3

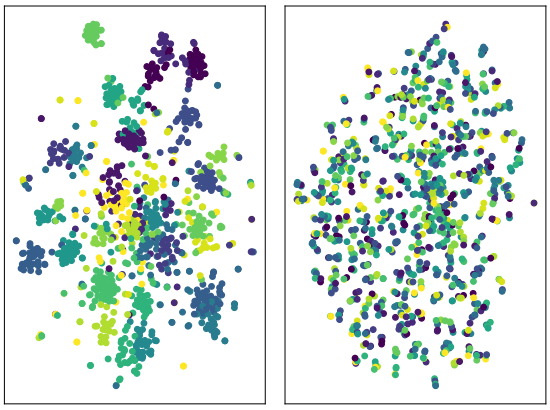

Figure 2: *t-SNE [21] of 1000 sampled 30-dimensional speaker i-vectors, before (left) and after (right) factorising the i-vectors. The colours indicate the 18 environments in the data. The loss of evident clustering shows that the bottleneck features have lost information about the environments.*

4.2. Results with bottleneck features

The results for using bottleneck features are shown in Table 5. When we rely solely on speaker or environment i-vectors, the use of the factorised bottleneck features provide about 12% relative reduction in WER on dev93d with 100-dimensional speaker i-vectors. Similarly, factorising the environment i-vectors reduces WER on eval92d by roughly 5.5% relative. It is interesting to note that reducing the speaker i-vector dimension to 30 improves WERs in the non-factorised case. The smaller dimension may lead to implicit factorisation.

When concatenating speaker and environment i-vectors we observe the opposite: the bottleneck features generally yield *increases* in WER. The non-factorised, 100 dimensional i-vectors provide the lowest WERs, whereas the lower dimensional i-vectors or the bottleneck features yield higher WERs. This may be because the non-factorised vectors can make use of correlations between speakers and environments during training, and because each target combination is present in the data.

Table 5: *i-vector WER (%) results on WSJ+DEMAND with perturbed test sets (dev93d / eval92d). Speaker vector sizes are provided in parentheses.*

Features	Spk	Env	Spk+Env
iv (100)	23.27 / 16.59	23.21 / 16.27	20.39 / 13.50
iv (30)	22.89 / 15.20	- / -	20.18 / 14.44
bn-iv (100)	20.34 / 14.46	22.48 / 15.36	20.08 / 14.85
bn-iv (30)	21.02 / 14.94	- / -	20.85 / 14.71

We investigated the sparsity of the vectors and observed a drop in the average number of nonzero elements, or L0 "norm" ($\sum_i |x_i|^0$, where $0^0 \equiv 0$). The 100-dimensional speaker bottleneck features had an average norm of 32.1 on the development set, yet only 5 units were consistently turned off. This suggests that the learned, factorised representations are still making use of the majority of the dimensions, but only about a third at any one time, perhaps improving the match at test time.

4.3. Results for unseen combinations

For this experiment we address the situation when joint adaptation data is not available, but we instead reuse transforms estimated in a single, mismatched condition. The results are shown in Table 6. Previously, concatenating speaker and environment vectors did not produce improvements in WERs with the bottleneck features (Table 5). This situation is now reversed, with up to 5% relative improvements. This is possibly because there is no longer any implicit averaging across environments or speakers, and because the true, joint combination is not present. The improvements when using only speaker or environment features are also more pronounced, with up to 21% relative for 100-dimensional speaker i-vectors and 14-17% relative for the environment i-vectors. There is now no possibility of implicit factorisation since the i-vectors are estimated in combination with exactly one mismatched condition. The results clearly demonstrate the requirement for factorisation when each factor has been estimated in strongly mismatched conditions.

Table 6: *Heldout experiments. i-vector WER (%) results on WSJ+DEMAND with perturbed test sets (dev93d / eval92d) with an equal distribution of environments for each speaker. Speaker vector sizes are given in parentheses.*

Features	Spk	Env	Spk+Env
iv (100)	25.21 / 17.26	25.47 / 19.03	21.12 / 14.48
iv (30)	24.71 / 17.19	- / -	21.29 / 15.33
bn-iv (100)	19.88 / 14.44	21.71 / 15.68	20.06 / 14.32
bn-iv (30)	19.71 / 14.80	- / -	19.57 / 14.19

5. Conclusions

We have demonstrated the possibility of factorising i-vectors using multicondition neural networks. Classification experiments on the (factorised) bottleneck features showed that information about the nuisance factor had been significantly reduced, while maintaining information about the class of interest. For speaker i-vectors we have shown up to roughly 12% relative improvements when the target environment is unknown or 5% when the target speaker is unknown. When adaptation data is unavailable and we reuse transforms extracted in mismatched conditions, we observe up to 21% relative improvements. A clear disadvantage with the presented approach is that we lose correlations between known speaker and environment combinations at training time, yielding lower frame accuracy. This is reflected in the WERs when using both speaker and environment i-vectors (`Spk+Env`). In future work we would like to investigate the speaker adaptive i-vector approach presented by Miao et al. [22] which might help mitigate this effect. Another limitation is that there is no explicit drive towards removing a factor from the bottleneck features: we are relying on this implicitly in the multicondition networks. Other architectures could embody this explicitly, through for example adversarial learning. Finally, we would like to perform experiments on more realistic data such as the Multi-Genre Broadcast corpus [23].

6. Acknowledgements

This work was partially supported by a PhD studentship funded by Bloomberg, and by the H2020 project SUMMA, under grant agreement 688139.

7. References

[1] 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*, pp. 1–1, 2016.

[2] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer Speech & Language*, vol. 12, no. 2, pp. 75–98, 1998.

[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," 1995.

[4] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 55–59.

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

[6] M. L. Seltzer and A. Acero, "Separating speaker and environmental variability using factored transforms." in *INTERSPEECH*, 2011, pp. 1097–1100.

[7] O. Saz and T. Hain, "Acoustic adaptation to dynamic background conditions with asynchronous transformations," *Computer Speech Language*, vol. 41, pp. 180 – 194, 2017.

[8] M. Gales, "Acoustic factorisation," in *Automatic Speech Recognition and Understanding, 2001. ASRU'01. IEEE Workshop on*. IEEE, 2001, pp. 77–80.

[9] Y. Q. Wang and M. J. F. Gales, "An explicit independence constraint for factorised adaptation in speech recognition." in *INTERSPEECH*, 2013, pp. 1233–1237.

[10] H. Seo, H.-G. Kang, and M. L. Seltzer, "Factored adaptation of speaker and environment using orthogonal subspace transforms," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3251–3255.

[11] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "CSR-I (WSJ0) Complete LDC93S6A," *Linguistic Data Consortium, Philadelphia*, 2007.

[12] N. Parihar, J. Picone, D. Pearce, and H.-G. Hirsch, "Performance analysis of the Aurora large vocabulary baseline system," in *Signal Processing Conference, 2004 12th European*. IEEE, 2004, pp. 553–556.

[13] L. Samarakoon and K. C. Sim, "Factorized hidden layer adaptation for deep neural network based acoustic modeling," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 12, pp. 2241–2250, 2016.

[14] J. Li, J. T. Huang, and Y. Gong, "Factorized adaptation for deep neural network," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5537–5541.

[15] J. Thiemann, N. Ito, and E. Vincent, "The Diverse Environments Multi-channel Acoustic Noise Database: A database of multichannel environmental noise recordings," *The Journal of the Acoustical Society of America*, vol. 133, no. 5, pp. 3591–3591, 2013.

[16] V. Peddinti, G. Chen, D. Povey, and S. Khudanpur, "Reverberation robust acoustic modeling using i-vectors with time delay neural networks," *Proceedings of INTERSPEECH. ISCA*, 2015.

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

[18] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 215–219.

[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[20] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[21] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[22] 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, 2015.

[23] P. Bell, M. J. F. Gales, T. Hain, J. Kilgour, P. Lanchantin, X. Liu, A. McParland, S. Renals, O. Saz, M. Webster *et al.*, "The MGB challenge: Evaluating multi-genre broadcast media transcription," in *Proceedings of IEEE workshop on Automatic Speech Recognition and Understanding, Scottsdale, AZ*, 2015.