Reverberation robust acoustic modeling using i-vectors with time delay neural networks

Vijayaditya Peddinti[1], *Guoguo Chen*[1], *Daniel Povey*[1,2], *Sanjeev Khudanpur*[1,2]

[1]Center for language and speech processing &
[2]Human Language Technology Center of Excellence
The Johns Hopkins University, Baltimore, MD 21218, USA
{vijay.p,guoguo,khudanpur}@jhu.edu, dpovey@gmail.com

Abstract

In reverberant environments there are long term interactions between speech and corrupting sources. In this paper a time delay neural network (TDNN) architecture, capable of learning long term temporal relationships and translation invariant representations, is used for reverberation robust acoustic modeling. Further, iVectors are used as an input to the neural network to perform instantaneous speaker and environment adaptation, providing 10% relative improvement in word error rate. By subsampling the outputs at TDNN layers across time steps, training time is reduced. Using a parallel training algorithm we show that the TDNN can be trained on $\sim$ 5500 hours of speech data in 3 days using up to 32 GPUs. The TDNN is shown to provide results competitive with state of the art systems in the IARPA ASpIRE challenge, with 27.7% WER on the *dev_test* set.

Index Terms: far field speech recognition, time delay neural networks, reverberation

1. Introduction

In reverberant environments the reflections of the signal affect the signal over several time frames. These long term interactions are due to multiple paths from each sound source to the microphone, each with its own delay. To tackle these longer term interactions between the direct speech signal and the corrupting sources, speech recognizers have account for long-term acoustic context [1].

Recurrent neural networks (RNNs) which use a dynamically changing contextual window over all of the sequence history rather than a fixed context window are a viable choice for learning reverberation robust representations. They have been shown to achieve state of the art performance on LVCSR tasks [2]. Weninger *et al* [3] have shown that deep RNNs are suitable for feature enhancement of reverberant speech signals. However due to recurrent connections in the network, parallelization during training cannot be exploited to the same extent as in feed forward neural networks.

In this paper we use a time delay neural network [4], which is a feed forward network architecture that is effective in modelling long term temporal contexts. In [5] it was shown that TDNNs can be trained with training times competitive with those of standard feed-forward DNNs, by sub-sampling the TDNN layer outputs. In this paper we use the TDNN architecture suggested in [5] for learning reverberation robust representations. The TDNN was able to benefit from increasing the input context up to 280 milliseconds. The ability to process such a wide temporal context enables the network to deal with late reverberations.

This work was partially supported by NSF Grant No IIA 0530118.

iVectors which capture both speaker and environment specific information have been shown to be useful for rapid adaptation of the neural network [6, 7, 8]. iVector based adaptation has also been shown to be effective in reverberant environments [9]. In this paper we use this adaptation technique.

We show experimental results on the ASpIRE far-field speech recognition challenge held by IARPA [10]. This challenge uses the English portion of the Fisher database [11] for acoustic and language model training. We show that in this large data scenario the proposed network architecture, combined with a parallel training technique [12], can train on multi-condition training data of $\sim$ 5500 hours, using up to 32 GPUs, in 3 days.

Using the TDNN architecture helps us to achieve results close to those of the best combined system submitted to the ASpIRE challenge, while using only a single system. Our system was able to achieve 27.7% WER on the *dev-test* set, while the best system achieved 27.2% WER.

The paper is organized as follows, Section 2 describes prior work, Section 3 describes the neural network architecture, Section 4 describes an online iVector extraction technique suitable for unsegmented audio recordings, Section 5 details the experimental setup, Section 6 analyses the results, and conclusions are presented in Section 7.

2. Relevant Prior Work

Signal and feature enhancement techniques [13, 14, 15] are widely used to tackle reverberation in ASR systems using standard acoustic models. However a potential drawback of the enhancement based approaches is the inevitability of estimation errors [1]. An alternative approach is to learn models which are robust to training-data distortions. Seltzer *et al.,* [16] have shown that DNN based acoustic models can learn distortion stable representations at higher layers in the network, when trained with multi-condition data. In this paper we use a combination of reverberated training data, a TDNN neural network, and iVector based adaptation, to learn a network that is robust to reverberations.

Reverberant speech is assumed to be composed of direct-path response, early reflections and late reverberations. Reflections within a delay of 50ms of the direct signal are categorized as early reflections. Late reverberations, which comprise of later reflections, have *reverberation time* from 200 to 1000 ms in typical office environments [1]. Early reflections can be effectively dealt with using DNN architectures, which operate on comparatively short temporal contexts. However in order to tackle late reverberations, DNNs should be able to model temporal relationships across wide acoustic contexts.

TDNNs [4], which are feed-forward neural networks, with the ability to model long term temporal relationships, were used. We used the sub-sampling technique proposed in [5] to achieve an acceptable training time.

In Section 3 we describe the time delay neural network architecture in more detail.

3. Neural network architecture

In a TDNN architecture the initial transforms are learnt on narrow contexts and the deeper layers process the hidden activations from increasingly wider contexts. Hence the higher layers have the ability to learn longer temporal relationships. The baseline TDNN network is shown in Figure 1. The figure shows the time steps at which activations are computed, at each layer, and the dependencies between activations across layers.

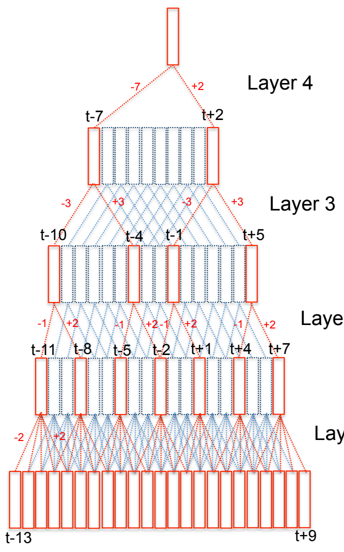

Figure 1: Computation in TDNN with sub-sampling (red) and without sub-sampling (blue+red)

The hyper-parameters which define the TDNN network structure are the set of frame offsets that we require as an input to each layer. In the case pictured, these are $\{-2, -1, 0, 1, 2\}$, $\{-2, 1\}$, $\{-3, 3\}$ and $\{-7, 2\}$. The output dimension at each hidden layer is constant. In a conventional TDNN, these input frame offsets would always be contiguous. However, in our work we sub-sample these; in our normal configuration, the frame splicing at the hidden layers splices together just two frames, separated by a delay that increases as we go to higher layers of the network [5].

In order to compute the network output at time t, the time steps at which at which activations are computed at each TDNN layer are shown in *red* in Figure 1. Our implementation of neural network training allows the evaluation of exactly the frames that are required at each layer. Compared with evaluating these frames in the smallest possible contiguous chunks at each layer, this reduces the overall computation required for training.

In this paper we were able to operate on input contexts of up to $\sim$ 280 ms without detriment in performance, using the TDNN. Thus the TDNN has the capability to tackle corruptions due to late reverberations.

Our TDNN uses the p-norm non-linearity [17]. We use a group size of of 10, and the 2-norm.

3.1. Input Features

Mel-frequency cepstral coefficients (MFCCs) [18], without cepstral truncation, were used as input to the neural network. 40 MFCCs were computed at each time index. MFCCs over a wide asymmetric temporal context were provided to the neural network. Different contexts were explored in this paper. 100 dimensional iVectors were also provided as an input to the network, every time frame. Section 4 describes the iVector extraction process during training and decoding in greater detail.

3.2. Training recipe

The paper follows the training recipe detailed in [17]. It uses greedy layer-wise supervised training, preconditioned stochastic gradient descent (SGD) updates, an exponentially decreasing learning rate schedule and *mixing-up*. Parallel training of the DNNs using up to 18 GPUs was done using the model averaging technique in [12].

3.2.1. Modified sMBR sequence training

Sequence training was done on the DNN, based on a state-level variant of the Minimum Phone Error (MPE) criterion, called sMBR [19] . The training recipe mostly follows [20], although it has been modified for the parallel-training method. Training is run in parallel using 12 GPUs, while periodically averaging the parameters, just as in the cross-entropy training phase.

Our previous sMBR-based training recipe degraded results on the ASpIRE setup, so we introduced a modification to the recipe which we have since found to be useful more generally.

In the sMBR objective function, as for MPE, insertion errors are not penalized. This can lead to larger number of insertion errors when decoding with sMBR trained acoustic models. Correcting this asymmetry in the sMBR objective function, by penalizing insertions, was shown to improve the WER performance of sMBR models by 10% relative. In standard sMBR training [19, 21], the frame error is always set to zero if the reference is silence, which means that insertions into silence regions are not penalized. In other words, frames where the reference alignment is silence are treated specially. (Note that in our implementation several phones, including silence, vocalized noise and non-spoken noise, are treated as silence for these purposes.) In our modified sMBR training method, we treat silence as any other phone, except that all pdfs of silence phones are collapsed into a single class for the frame-error computation. This means that replacing one silence phone with another silence phone is not penalized (e.g. replacing silence with vocalized-noise is not penalized), but insertion of a non-silence phone into a silence region is penalized. This is closer to the WER metric that actually we care about, since WER is generally computed after filtering out noises, but does penalize insertions. We call our modified criterion the “one-silence-class” modification of sMBR.

4. iVector Extraction

In this section we describe the iVector estimation process adopted during training and decoding. We discuss issues in estimating iVectors from noisy unsegmented speech recordings, and in using these noisy estimates of iVectors as input to neural networks.

On each frame we append a 100-dimensional iVector [22] to the 40-dimensional MFCC input. The MFCC input is not subject to cepstral mean normalization; the intention is to allow the iVector to supply the information about any mean offset of the speaker's data, so the network itself can do any feature normalization that is needed. In order for the mean-offset information to be encoded in the iVector, we estimate the iVector on top of features that have not been mean-normalized. However, the Gaussian posteriors used for the iVector estimation are based on features that have been mean normalized using a sliding window of 6 seconds.

We noticed that the iVector adaptation was not sufficiently effective in adapting to test signals that had substantially different energy levels than the training data. For the results reported here, this issue was resolved by normalizing the test-signal energies to be the same as the average of the training data. This normalization resulted in a relative improvement of 15% in WER. The reason why the network did not learn to do this normalization automatically is that the training data was too carefully normalized; in future we will randomize the volume of our training data to force the network to learn this type of normalization.

4.1. iVector Extraction during training

The iVector estimator was trained on a 100 hour subset of training data: this includes the training of the Gaussian mixture model used for the UBM, and the estimation of the T matrix. Then, for the entire training data, iVectors were estimated. In order to ensure sufficient variety of the iVectors in the training data, rather than estimating a separate iVector per speaker we estimate them in an online fashion, where we only use frames prior to the current frame (for some arbitrary ordering of the utterances). We re-set this history every two utterances, so that we still have some training-data variety even when there are only a few speakers.

4.2. iVector extraction during decoding

During decoding, the constraints of online extraction were not enforced and iVectors were estimated in an offline fashion from statistics accumulated over fairly large portions of the speaker's data (at least 60 seconds).

The prior term in the iVector extraction is quite important when applying these iVector based methods to data that is dissimilar to the training data. In our iVector estimation we always scale the per-frame posteriors by 0.1 (equivalent to scaling the prior term up by 10). For the ASpIRE challenge we made a further modification: if the total count of (scaled) statistics for iVector extraction exceeds a predefined limit (75 for these experiments), we scale the statistics down to that value, which again is equivalent to scaling the prior term up. Due to the posterior scale of 0.1, this effect kicks in after we exceed 750 frames of features.

4.2.1. iVectors from reliable speech segments

In the current LVCSR task (see below), audio recordings 5-10 minutes in length were provided without speech end-point information. The recordings had long durations of contiguous silence, similar to single channel recordings of conversational telephone speech. We found empirically that excluding the silence from the statistics for iVector estimation was very helpful. Even keeping a small amount of silence around every speech segment (similar to the amount we saw in training) was harmful; possibly the nature of the silence in the ASpIRE test data was so different from what was seen in the artificially reverberated and noise-added training data, that it affected the iVector in unexpected ways.

In order to detect regions containing speech, we perform a first-pass decode of the audio data using iVectors derived from both speech and non-speech regions. Reliable speech segments are identified from this first-pass decode. Audio segments corresponding to words with confidence measures of 1.0 (derived from lattice posteriors) and with durations less than one second were considered reliable (over half the words recognized had a confidence of at least 1.0). We also excluded the words "mm" and "mhm". A second pass decode was then performed using the iVectors estimated from these reliable speech segments. This led to 8.9% relative improvement in WER, versus using all the data for iVector estimation.

5. Experimental Setup

5.1. Acoustic Model

Speech from the English portion of the Fisher corpus [11] (LDC2004S13, LDC2005S13) was used to train the acoustic models. Multi-condition training data was created by distorting speech with real world room impulse response (RIR) and noise recordings available from three different databases *viz.,* the RWCP sound scene database [1] [23], the REVERB challenge database [24] and the Aachen impulse response database [25]. 320 muti-channel recordings of RIRs and corresponding noise recordings were selected from the three databases. The first channel from the multi-channel recordings was used for corruption. Three different copies of each recording in the Fisher corpus were created by randomly sampling three different pairs of RIRs and noise recordings from the above set. Overall ~ 5500 hours of training data was created, based on these copies. Another version of the acoustic model was trained on data that was processed as above, but also then speed perturbed as in [26]. We still produced 3 copies of each original recording, by combining a random RIR and noise recording with each different speed of the data. Speed perturbation, which emulates pitch and tempo variations in speech, was shown to provide on average 4.3% relative gain in a variety of LVCSR tasks. However this did not help in the current task, possibly because we already had enough training-data variation from the reverberation and noise.

A GMM-HMM acoustic model was used to generate alignments for training neural networks. This acoustic model was trained on clean Fisher speech data using features that were obtained by splicing together 7 frames (3 on each side of the current frame) of 13-dimensional MFCCs (C0-C12) and projecting down to 40 dimensions using linear discriminant analysis (LDA). The MFCCs were normalized to have zero mean per speaker. We also used a single semi-tied covariance (STC) transform [27] on the features obtained using LDA. These combined features are referred to as LDA+STC. Moreover, speaker adaptive training (SAT) was done using a single feature-space maximum likelihood linear regression (FMLLR) transform estimated per speaker. Alignments for clean speech were generated using the GMM-HMM system, using clean data as recommended in [28, 29].

[1]We would like to thank Mitsubishi Electric Research Laboratories (MERL), for providing the RWCP database.

5.2. Language Model

A trigram language model (LM) is first trained on the 3M words of the training transcripts, which is later interpolated with another trigram LM trained on 22M words of the Fisher English transcripts (LDC2004T19 and LDC2005T19). The same process is repeated for building a 4gram LM. We use SRI's language modeling toolkit SRILM [30] for building our LMs, with Kneser-Ney smoothing. The final trigram LM has 1.6M trigrams and the 4gram LM has 1.7M 4grams. We directly use the trigram LM for decoding. The 4gram LM is only used for rescoring the lattices generated by the trigram LM; we do not use it for lattice generation for reasons of memory efficiency.

Estimating pronunciation probabilities from training data has been found helpful when multiple pronunciations are available for certain words in the training lexicon [31, 32]. In [32] it is shown that modeling word-dependent silence probabilities as well as pronunciation probabilities, imroves recognition performance consistently over multiple datasets. We follow that approach in our ASpIRE system.

5.3. Decoding

Two data sets *dev* of 5 hrs and *dev-test* of 10 hrs were provided as part of ASpIRE challenge. Each set is composed of 10 minute recordings. The end points for the speech portions of the recording were also provided for the *dev* set. However in order to emulate the decoding scenario of *dev-test*, we report performance on *dev* set without the knowledge of segment information.

Decoding the entire 10 minute recording as one segment is not possible due to round-off in the decoder. We segmented the recordings into chunks of 10 seconds long each, shifted by 5 seconds each time. There was no attempt to make the chunk boundaries coincide with silence. We reasoned that if a recording is cut in the middle, only the part of the transcript near the cut point will be affected, so we filtered the transcripts by removing words whose midpoints were within 2.5 seconds of the edge of its chunk of origin, before combining them into a single long transcript.

6. Results

The TDNN had 6 layers, of which 3 layers (not counting the input layer) were subject to frame splicing across multiple time offsets. Three different systems TDNN-A, TDNN-B and TDNN-C corresponding to the three different input contexts of $[t-13, t+9]$ frames, $[t-16, t+12]$ frames and $[t-22, t+12]$ were used in the comparison. The splicing configuration of the TDNN-A system was $[-2, 2]$, $\{-1, 2\}$, $\{0\}$, $\{-3, 3\}$, $\{-7, 2\}$, $\{0\}$ (where the $\{0\}$ layers are conventional, non-splicing hidden layers). The TDNN-B and TDNN-C systems were as TDNN-A except replacing $\{-7, 2\}$ with $\{-10, -7, 2, 5\}$ and $\{-16, -7, 2, 5\}$ respectively. In all hidden layers the p-norm input and output dimensions were 4000 and 400 respectively.

Results corresponding to this comparison are presented in Table 1. It was observed that input contexts of $[t-16, t+12]$, were optimal for training on reveberant speech. This result can be compared with the input temporal context of $[t-13, t+9]$ found optimal for recognition of non-reverberant speech in [5]. This additional context could be necessary when training on reverberant data to compensate for the late reverberations. The use of even larger temporal contexts (TDNN-C) did not lead to better results. However it was interesting to note that use of larger contexts were not detrimental to the same extent as seen in other speech recognition tasks with non-reverberant data [5].

Table 1: Comparison of input contexts and training data augmentation, used for training the TDNNs

Acoustic Model	training data	*dev* WER
TDNN A	rvb	31.7
TDNN B	rvb	30.8
TDNN B	rvb + sp	31.0
TDNN C	rvb + sp	31.1

rvb :	reverberation of training data using real world RIRs
sp :	speed perturbation of data prior to reverberation

Table 2: Comparison of systems with and without iVectors

Acoustic Model	*dev* WER
TDNN B w/o iVectors	34.8
TDNN B + iVectors[1]	33.8
TDNN B + iVectors[2]	30.8

[1] estimated on speech and non-speech
[2] estimated on reliable speech segments

Table 3: Results with sequence training of TDNN models

Acoustic Model	*dev* WER	*dev-test* WER
TDNN A	31.7	- *
TDNN A + sequence training[1]	34.0	- *
TDNN A + sequence training[2]	30.6	30.1
TDNN B	30.8	27.7
TDNN B + sequence training[2]	29.5	28.9

[1] with sMBR criterion
[2] with modified sMBR criterion
* not available due to shutdown of scoring servers

These systems were trained on data generated from two different types of data augmentation techniques which are reverberation (rvb) and speed perturbation (sp). Speed perturbation which was shown to be advantageous across several LVCSR tasks [26], was not helpful in the current task. Training for more epochs improved the performance of the TDNN-B (rvb+sp) system; however, it just matched the TDNN-B (rvb) system.

Table 2 compares systems trained with and without iVectors, and different types of iVector extraction. The use of iVectors from reliable speech segments is critical.

Table 3 shows results of TDNNs using sequence training. The standard sMBR criterion was detrimental to the performance; but using the modified sMBR criterion, gains were observed on *dev* set. However these did not translate to *dev-test* set. With sequence training there was 4.2% relative improvement on *dev* set and 4.3% relative decrease on *dev-test* set. TDNN-B without sequence training was chosen as our best system with *dev-test* performance of 27.7%.

7. Conclusions

In this paper, we used a TDNN on input context on $[-16, 12]$ frames for reverberation robust acoustic modeling. We discussed issues in extracting iVectors from unsegmented reverberant audio recordings and suggested steps to tackle these issues. We proposed a modified sMBR criterion, which penalizes insertions. Using a combination of these techniques, we showed improvements in a far-field recognition task.

8. References

[1] T. Yoshioka, A. Sehr, M. Delcroix, K. Kinoshita, R. Maas, T. Nakatani, and W. Kellermann, "Making machines understand us in reverberant rooms: Robustness against reverberation for automatic speech recognition," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 114–126, Nov 2012.

[2] H. Sak, A. Senior, and F. Beaufays, "Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition," Feb. 2014. [Online]. Available: http://arxiv.org/abs/1402.1128

[3] F. Weninger, S. Watanabe, J. Le Roux, J. Hershey, Y. Tachioka, J. Geiger, B. Schuller, and G. Rigoll, "The merl/melco/tum system for the reverb challenge using deep recurrent neural network feature enhancement," in *Proc. REVERB Workshop*, 2014.

[4] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 37, no. 3, pp. 328–339, Mar. 1989.

[5] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," *submitted to Interspeech 2015*, 2015. [Online]. Available: http://speak.clsp.jhu.edu/uploads/publications/papers/1048_pdf.pdf

[6] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q.-F. Liu, "Fast Adaptation of Deep Neural Network based on Discriminant Codes for Speech Recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. PP, no. 99, pp. 1–1, 2014.

[7] M. Karafiat, L. Burget, P. Matejka, O. Glembek, and J. Cernocky, "iVector-based discriminative adaptation for automatic speech recognition," in *2011 IEEE Workshop on Automatic Speech Recognition & Understanding*. IEEE, Dec. 2011, pp. 152–157.

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

[9] M. J. Alam, V. Gupta, P. Kenny, and P. Dumouchel, "Use of multiple front-ends and i-vector-based speaker adaptation for robust speech recognition," *Proc. of IEEE REVERB Workshop*, 2014.

[10] *Automatic Speech recognition In Reverberant Environments (ASpIRE) Challenge*, 2015 (accessed March 23 , 2015), http://www.iarpa.gov/index.php/working-with-iarpa/prize-challenges/306-automatic-speech-in-reverberant-environments-aspire-challenge.

[11] C. Cieri, D. Miller, and K. Walker, "The fisher corpus: a resource for the next generations of speech-to-text." in *LREC*, vol. 4, 2004, pp. 69–71.

[12] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *CoRR*, vol. abs/1410.7455, 2014. [Online]. Available: http://arxiv.org/abs/1410.7455

[13] M. Miyoshi and Y. Kaneda, "Inverse filtering of room acoustics," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 36, no. 2, pp. 145–152, 1988.

[14] A. Krueger and R. Haeb-Umbach, "A model-based approach to joint compensation of noise and reverberation for speech recognition," in *Robust speech recognition of uncertain or missing data*. Springer, 2011, pp. 257–290.

[15] T. Nakatani, T. Yoshioka, K. Kinoshita, M. Miyoshi, and B.-H. Juang, "Speech dereverberation based on variance-normalized delayed linear prediction," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 18, no. 7, pp. 1717–1731, 2010.

[16] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7398–7402.

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

[18] S. B. Davis and P. Mermelstein, "Comparison of parametric representation for monosyllabic word recognition in continuously spoken sentences," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 28, no. 4, pp. 357–366, 1980.

[19] M. Gibson, "Minimum bayes risk acoustic model estimation and adaptation," Ph.D. dissertation, University of Sheffield, 2008.

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

[21] D. Povey and B. Kingsbury, "Evaluation of proposed modifications to mpe for large scale discriminative training," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 4. IEEE, 2007, pp. IV–321.

[22] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788 –798, May 2011.

[23] S. Nakamura, K. Hiyane, F. Asano, T. Nishiura, and T. Yamada, "Acoustical sound database in real environments for sound scene understanding and hands-free speech recognition." in *LREC*, 2000.

[24] K. Kinoshita, M. Delcroix, T. Yoshioka, T. Nakatani, A. Sehr, W. Kellermann, and R. Maas, "The reverb challenge: Acommon evaluation framework for dereverberation and recognition of reverberant speech," in *Applications of Signal Processing to Audio and Acoustics (WASPAA), 2013 IEEE Workshop on*. IEEE, 2013, pp. 1–4.

[25] M. Jeub, M. Schafer, and P. Vary, "A binaural room impulse response database for the evaluation of dereverberation algorithms," in *Digital Signal Processing, 2009 16th International Conference on*. IEEE, 2009, pp. 1–5.

[26] T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," *Submitted to Interspeech 2015*, 2015. [Online]. Available: http://speak.clsp.jhu.edu/uploads/publications/papers/1050_pdf.pdf

[27] M. J. Gales, "Semi-tied covariance matrices for hidden markov models," *Speech and Audio Processing, IEEE Transactions on*, vol. 7, no. 3, pp. 272–281, 1999.

[28] M. Delcroix, T. Yoshioka, A. Ogawa, Y. Kubo, M. Fujimoto, N. Ito, K. Kinoshita, M. Espi, T. Hori, T. Nakatani *et al.*, "Linear prediction-based dereverberation with advanced speech enhancement and recognition technologies for the reverb challenge," in *Proc. REVERB Workshop*, 2014.

[29] Y. Tachioka, T. Narita, F. Weninger, and S. Watanabe, "Dual system combination approach for various reverberant environments with dereverberation techniques," in *Proc. of IEEE REVERB Workshop*, 2014.

[30] A. Stolcke *et al.*, "Srilm-an extensible language modeling toolkit." in *Proceedings of INTERSPEECH*, 2002.

[31] T. Hain, P. Woodland, G. Evermann, and D. Povey, "The CU-HTK march 2000 Hub5e transcription system," in *Proceedings of Speech Transcription Workshop*, vol. 1, 2000.

[32] G. Chen, H. Xu, M. Wu, D. Povey, and S. Khudanpur, "Pronunciation and silence probability modeling for ASR," *Submitted to INTERSPEECH 2015*, 2015. [Online]. Available: http://www.clsp.jhu.edu/~guoguo/papers/interspeech2015_silprob.pdf