Speech Dereverberation Using Long Short-Term Memory

Masato Mimura *Shinsuke Sakai* *Tatsuya Kawahara*

Kyoto University, Academic Center for Computing and Media Studies,
Sakyo-ku, Kyoto 606-8501, Japan

Abstract

Recently, neural networks have been used for not only phone recognition but also denoising and dereverberation. However, the conventional denoising deep autoencoder (DAE) based on the feed-forward structure is not capable of handling very long speech frames of reverberation. LSTM can be effectively trained to reduce the average error between the enhanced signal and the original clean signal by considering the effect of the long past time frames. In this paper, we demonstrate that considering as long as the maximum reverberation time of the database is effective. Since the effect of reverberation varies depending on the phone-class of the whole speech context, we augment the input of the autoencoder with the phone-class information of the past frames as well as the current frame and call this version of the LSTM autoencoder pLSTM. In the speech recognition experiment using the data set of Reverb Challenge 2014, the LSTM front-end reduced the WER of the multi-condition DNN-HMM by 14.5%, and the use of the phone class feature yielded in pLSTM further improvement of 7.5%. The performance with the pLSTM is comparable to that of pDAE, while the number of parameters is only 1/25-1/8.

Index Terms: Speech Dereverberation, Long Short-Term Memory (LSTM), Deep Autoencoder (DAE)

1. Introduction

In recent years, the speech recognition technology based on statistical techniques achieved a remarkable progress supported by the ever increasing training data and the improvements in the computing resources. Applications such as voice search are now being used in our daily life. However, speech recognition accuracy in adverse environments such as those with reverberation and background noise is still at low levels. A key breakthrough for the speech recognition technology to be accepted widely in the society will be the methodology for hands-free input. This is critical for realizing conversational robots, for example. Speech reverberation adversely influences the recognition accuracy when the microphone is distant and various efforts have been made to solve this problem.

This paper focuses on the front-end feature enhancement for reverberant speech recognition. One of the simplest approaches to feature enhancement is the cepstral mean normalization (CMN) [1]. However, since reverberation time is usually longer than the frame window length for feature extraction, its effectiveness is limited. More sophisticated enhancement techniques include deconvolution approaches that reconstruct clean speech by inverse-filtering reverberant speech [2][3][4] and spectral enhancement approaches that estimate and remove the influences of the late reverberation [5][6].

Recently, following the great success of deep neural networks (DNN), dereverberation by deep autoencoders (DAE) has been investigated [7][8][9]. In these works, DAEs are trained using reverberant speech features as input and the clean speech features as target so that they recover the clean speech from corrupted speech in the recognition stage. In [10], we proposed to use phone-class features as well as acoustic features for the DAE-based dereverberation, and showed that it improves speech recognition performance.

While DAEs have a vertically deep structure and effectively learn the complicated mapping, they can exploit only local context information with limited time windows. The reverberant time is often long. Using very long context information, however, makes the number of parameters of the DAE so large that it cannot be reliably trained. Moreover, it is difficult to utilize the phone-class information of the past frames in the DAE framework, while they may be useful for recovering the clean data. One possible solution is to use Recurrent Neural Networks (RNNs), which can perform sequential information processing in a compact representation. But conventional RNNs have fundamental shortcomings that they cannot learn to find very long-term dependencies because of the *vanishing gradient problem* [11]. For example, the reverberation time of the large room in the Reverb Challenge 2014 [12] is 0.7 seconds (= 70 frames in the usual setting), and it may be too long for RNNs to adequately handle it.

The Long Short-Term Memory (LSTM) architecture [13] was introduced to overcome the vanishing gradient problem by enforcing a constant error flow through the special units called *Constant Error Carousels*. Recently, LSTMs have been applied to several tasks in the speech processing area ([14][15][16][17][18][19][20][21][22][23][24][25][26]), and yielded comparable to or even better results than DNNs without any pre-training and high dimensional spliced feature vectors as input. In this paper, we explore the speech dereverberation using the LSTM [16] and propose to use the phone-class information for this LSTM-based dereverberation.

After a brief review on the DAE-based front-end dereverberation and the phone-class feature in Section 2, the detail of the proposed LSTM-based method is explained in Section 3. Experimental evaluations of the method are presented in Section 4 before the conclusion in Section 5.

2. DAE-based dereverberation

2.1. Deep Autoencoders (DAE)

The combination of DNNs for phone state classification and traditional HMM acoustic models has yielded a dramatic improvement in speech recognition accuracies [27][28][29][30]. DNNs are also applied to front-end feature enhancement in the robust speech recognition area [31][7][8][16][32][33]. DNNs used for regression tasks such as speech enhancement are often called deep autoencoders (DAEs) [34]. Unlike DNNs for classification, DAEs are typically trained to reconstruct signals by using the mean squared error (MSE) as the loss function [35]. A DAE

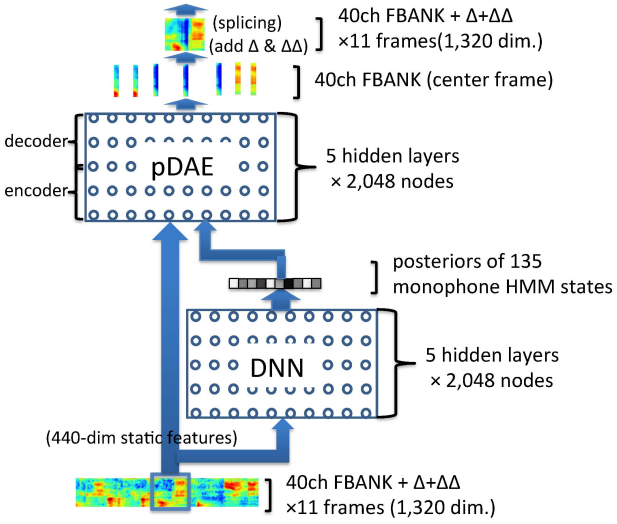

Figure 1: Feature enhancement by pDAE using soft phone class features

has a vertically symmetric network structure and each layer is initialized by an RBM [34].

DAEs for dereverberation are trained using the clean speech features as target and the reverberant speech features as input. In [9], we investigated reverberant speech recognition based on DAE front-end coupled with DNN acoustic model. Since the dereverberation using the DAE is performed not at the STFT level ([7]) but at the feature level ([8][16]) in our system, we can directly feed the DAE output to the DNN-HMM acoustic model. The input feature vector of DAE at frame t, $\boldsymbol{x}_t^{DAE}$, consists of multiple frames of filterbank output,

$$\boldsymbol{x}_t^{DAE} = [\boldsymbol{a}_{t-5}, \boldsymbol{a}_{t-4}, ..., \boldsymbol{a}_t, ..., \boldsymbol{a}_{t+4}, \boldsymbol{a}_{t+5}], \quad (1)$$

where $\boldsymbol{a}_t$ is the acoustic feature (filterbank output) at frame t.

2.2. Augmentation with phone-class feature

The mapping from corrupted data to the clean data is conventionally conducted only with the acoustic information. Since the acoustic features in clean speech vary depending on phones, the phone-class information is helpful for the DAE to recover the clean speech from corrupted speech, as we showed in [10]. While we compared four different types of phone-class features in [10], we use only the soft feature PC_{soft} in this paper, which brought significant improvement without an additional recognition pass. PC_{soft} is derived from the posterior outputs of monophone DNN. DAE using the PC_{soft} feature is illustrated in Figure 1. We call the DAE augmented with the PC_{soft} feature *pDAE*, hereafter. The input feature of the pDAE is defined as

$$\boldsymbol{x}_t^{pDAE} = [\boldsymbol{x}_t^{DAE}, \boldsymbol{p}_t] \quad (2)$$

Here, $\boldsymbol{p}_t$ is the phone-class feature for frame t.

3. LSTM for dereverberation

3.1. Long Short-Term Memory (LSTM)

An LSTM network (Figure 2) computes a mapping from an input sequence $\boldsymbol{x} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_T)$ to the cell output sequence

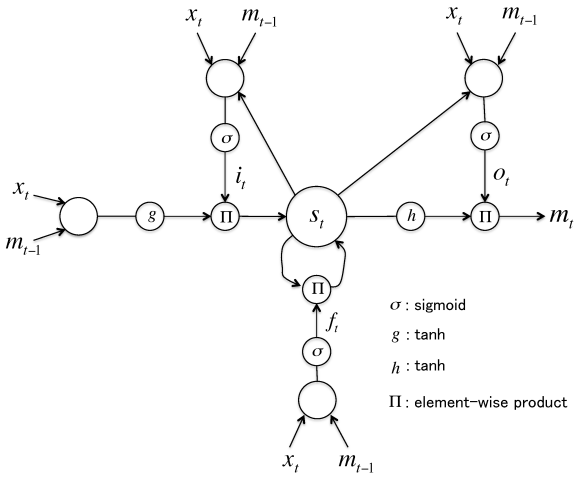

Figure 2: Long Short-Term Memory (LSTM)

$\boldsymbol{m} = (\boldsymbol{m}_1, ..., \boldsymbol{m}_T)$ iteratively, unlike feed-forward networks which compute the output of each frame independently. The mapping is obtained by calculating the network unit activations using the following equations iteratively from $t = 1$ to T:

$$\boldsymbol{i}_t = \sigma(W_{ix}\boldsymbol{x}_t + W_{im}\boldsymbol{m}_{t-1} + W_{is}\boldsymbol{s}_{t-1} + \boldsymbol{b}_i) \quad (3)$$
$$\boldsymbol{f}_t = \sigma(W_{fx}\boldsymbol{x}_t + W_{fm}\boldsymbol{m}_{t-1} + W_{fs}\boldsymbol{s}_{t-1} + \boldsymbol{b}_f) \quad (4)$$
$$\boldsymbol{s}_t = \boldsymbol{f}_t \odot \boldsymbol{s}_{t-1} + \boldsymbol{i}_t \odot \tanh(W_{sx}\boldsymbol{x}_t + W_{sm}\boldsymbol{m}_{t-1} + \boldsymbol{b}_c) \quad (5)$$
$$\boldsymbol{o}_t = \sigma(W_{ot}\boldsymbol{x}_t + W_{om}\boldsymbol{m}_{t-1} + W_{os}\boldsymbol{s}_t + \boldsymbol{b}_o) \quad (6)$$
$$\boldsymbol{m}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{s}_t) \quad (7)$$

where W_{**} denotes a weight matrix (e.g. W_{ix} is the matrix of weights from the input gate to the input) and $\boldsymbol{b}_*$ denotes a bias vector (e.g. $\boldsymbol{b}_i$ is the input gate bias vector). σ is the logistic sigmoid function. i, f, o and s are the input gate, forget gate, output gate and cell activation vectors, respectively. m is the cell output activation vector. $\odot$ is the element-wise product of the vectors.

Since the LSTM is free from the vanishing gradient problem, it can work even when there are very long delays. Another important advantage of the LSTM is that it can handle signals that have a mix of low and high frequency components due to the existence of three kinds of gate units. Therefore, the LSTM trained using multi-condition data is expected to be able to perform dereverberation adaptively whether the reverberation time of the test utterance is short or long.

3.2. LSTM-based dereverberation

The network for regression tasks such as dereverberation can be built by stacking the output layer with identity activation function on the top of the LSTM layer. Figure 3 illustrates the feature enhancement by using a 2-layer LSTM and an output layer. The enhanced feature $\boldsymbol{a}_t^{enh}$ is calculated using the cell output $\boldsymbol{m}_t$ by the following equation.

$$\boldsymbol{a}_t^{enh} = W_{output}\boldsymbol{m}_t + \boldsymbol{b}_{output} \quad (8)$$

where W_{output} and $\boldsymbol{b}_{output}$ are the weight matrix and the bias vector of the output layer, respectively. The input vector of the LSTM for frame t is identical to the acoustic feature of the current frame.

$$\boldsymbol{x}_t^{LSTM} = [\boldsymbol{a}_t] \quad (9)$$

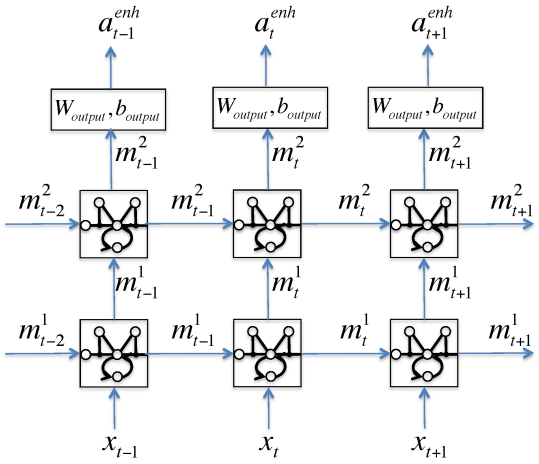

Figure 3: Feature enhancement using 2-layer LSTM

The training of the network is performed by backpropagation through time (BPTT) using the mean squared error between the enhanced feature $\boldsymbol{a}_t^{enh}$ and the original clean feature $\boldsymbol{a}_t^{cln}$ as the loss function. Ideally the gradients used for the stochastic gradient descent algorithm are calculated using inputs of all speech frames from $\tau = 1$ to t, but it is actually infeasible. Therefore, we exclude frames which are more distant from t than the fixed number T_{bptt} from the calculation (*truncated BPTT*). The loss function is defined by the following equation.

$$L = \sum_{\tau=t-T_{bptt}+1}^{t} |\boldsymbol{a}_t^{enh} - \boldsymbol{a}_t^{cln}|^2 \tag{10}$$

The speech frames more distant than T_{bptt} from t does not affect the error of the current frame. Therefore, T_{bptt} should be adequately determined in accordance with the reverberation time of the speech database.

The LSTM-based dereverberation network can also use the phone-class feature as input, and the input vector of this augmented model (pLSTM) is defined as

$$\boldsymbol{x}_t^{pLSTM} = [\boldsymbol{x}_t^{LSTM}, \boldsymbol{p}_t] \tag{11}$$

With the recurrent architecture, phone-class information of the past speech frames is considered as well as the current frame, which was difficult in the feed-forward DAE framework.

4. Experimental evaluation

4.1. Task and data set

The proposed system was evaluated following the instructions for the task of the Reverb Challenge 2014 [12]. For training, we used the standard multi-condition data that is generated by convolving clean WSJCAM0 data with room impulse responses (RIRs) and subsequently adding noise signals. The amount of the training data is 15.5 hours (7,861 utterances). Evaluation data consists of "SimData" and "RealData". SimData is a set of reverberant speech generated by convolving clean speech with various RIRs and adding measured noise signals to make the resulting SNR to be 20dB. RIRs were recorded in three different-sized rooms (small, medium, and large) and with two microphone distances (near=50cm and far=200cm). The reverberation time (T60) of the small, medium, and large rooms are about 0.25s, 0.5s, and 0.7s, respectively. RealData was recorded in a different room from those used for measuring RIRs for SimData. It has a reverberation time of 0.7s. There are two microphone distances in RealData, which are near (≈100cm) and far (≈250cm). In the experiments in this paper, we only use a single channel data both for training and testing. For decoding, we used the HDecode command from HTK-3.4.1 with a small modification to handle DNN output. The language model we used is the standard WSJ 5K trigram model. The triphone DNN-HMM acoustic model with 3,117 shared states was trained using the multi-condition data. The monophone DNN with 135 states was also trained for the calculation of the phone-class feature. The acoustic feature used in all models is 40-channel log Mel-scale filterbank outputs.

The baseline DAE has six layers in total including five sigmoidal hidden layers. The number of nodes in each layer is 2,048 except for input and output layers. The detail of the DAE training is described in [9].

4.2. Training of LSTM model

We trained the LSTM model using the multi-condition data by the truncated BPTT algorithm described in Section 3.1. The number of the memory cells was 400. The initial learning rate was set to be 0.1, and it was halved if the improvement in the frame accuracies on the heldout set calculated with the triphone DNN between two consecutive epochs fell below 0.2%. The training was stopped after 20 epochs. The momentum was set to be 0.4. The weights in all the networks are initialized to the range (-0.08, 0.08) with a uniform distribution. For increasing throughput, we introduced the mini-batch based parallelization. Each mini-batch consists of 128 utterances. The unit activations for each utterance can be independently calculated as matrix operations using GPGPUs. We managed so that the length of the utterances in the same mini-batch is similar in order to maximize the efficiency of the parallelization. The gradient calculated by the truncated BPTT was divided by the product of mini-batch size and T_{bptt}. Since the gradient can sometimes explode in the training of LSTMs, we used hard constraint over the norm of the gradient so that it never exceed 15.0. We performed the gradient calculation and weight update once per five frames.

We compared three different values for T_{bptt}, 25, 50 and 70, which correspond to the reverberation time of the three types of rooms used in the training corpus. Figure 4 plots the change of the training error over epochs for the LSTMs with different T_{bptt}. The LSTM with the largest T_{bptt} achieved the lowest mean squared error. From these results, we understand that the speech frames as distant as the largest reverberation time in the corpus can affect the dereverberation of the current frame.

Figure 5 shows the change of the mean squared error on the training data for the standard LSTM, the LSTM augmented with the phone-class feature (pLSTM) and the pLSTM with 2-layers. As shown in the figure, the use of the phone-class information contributed to the drastic reduction in the error. The increase of the number of LSTM layers yielded a further improvement.

4.3. Evaluation on simulated reverberant data

We evaluated the dereverberation front-end models using the simulated reverberant test data (SimData) in the Reverb Challenge 2014.

First, we evaluated the mean squared error (MSE) between the original clean feature and the enhanced feature obtained by LSTMs with different T_{bptt}. Figure 6 shows the MSE on each condition in SimData. As expected from the errors in the train-

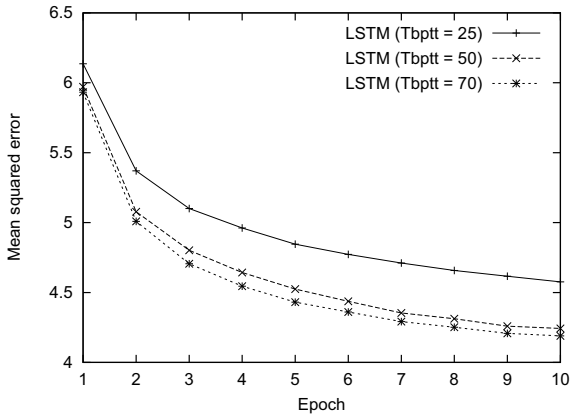

Figure 4: Change of training error over epochs for LSTMs with different T_{bptt}

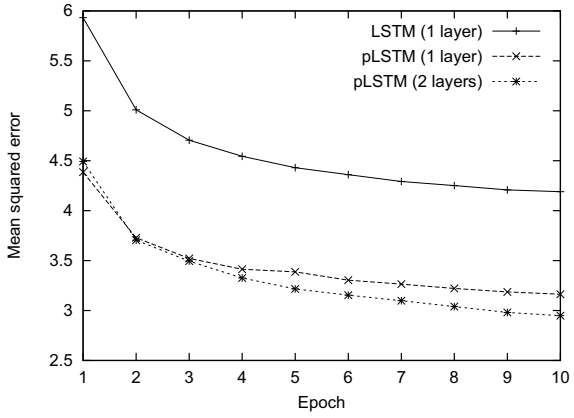

Figure 5: Training data errors by LSTM and pLSTM

ing stage, the LSTM with the T_{bptt} which corresponds to the longest reverberation time in the corpus achieved the smallest MSE in all conditions, except for Middle-Near. From the figure, we see that the LSTM trained using the multi-condition data with a large T_{bptt} can perform dereverberation adaptive to the degree of the reverberation in the test utterance.

The average MSE and word error rate (WER) on SimData by all models including the baseline DAE and pDAE are shown in Table 1. When using only standard acoustic features (filterbank outputs) as input, the performance of the LSTM was not as good as the baseline DAE both on the MSE and WER. However, when the input was augmented with the phone-class feature, the LSTM (pLSTM) yielded comparable results to the DAE (pDAE). These results suggest that the past phone-class features as well as the current one are working effectively in the LSTM framework.

4.4. Evaluation on real reverberant data

We conducted speech recognition experiments using the real reverberant test data (RealData) of the Reverb Challenge 2014. Note that the microphone distances in RealData are much different from those in the training data and the recognition of RealData is much harder than SimData. The WERs obtained by all models are shown in Table 2.

The LSTM front-end improved the WER of the multi-condition DNN-HMM by relative 14.5%, and the use of the

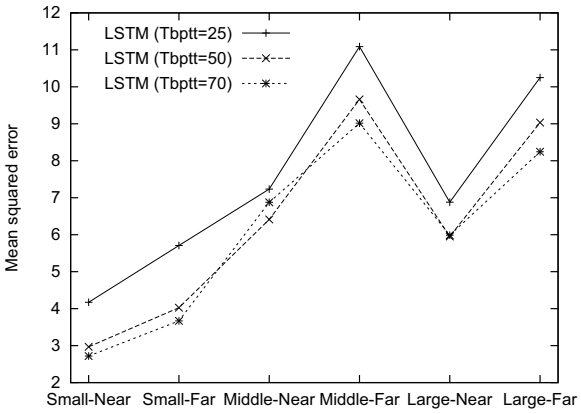

Figure 6: Mean squared error on each condition in SimData by LSTMs with different T_{bptt}

Table 1: Mean squared error (MSE) and word error rate (WER) on SimData

Front-end	# of params	SimData	
		MSE	WER (%)
(no enhance)	-	13.8	8.74
DAE (6 layers)	17.8M	5.50	10.62
pDAE (6 layers)	18.0M	5.17	9.87
LSTM (1 layer)	0.72M	6.08	12.07
pLSTM (1 layer)	0.94M	5.25	9.74
pLSTM (2 layers)	2.22M	5.08	9.89

Table 2: Word error rate on RealData (WER (%))

Front-end	# of params	RealData		
		Near	Far	Ave.
(no enhance)	-	28.59	30.87	29.67
DAE (6 layers)	17.8M	24.37	25.52	24.93
pDAE (6 layers)	18.0M	23.47	23.09	23.29
LSTM (1 layer)	0.72M	25.68	25.02	25.36
pLSTM (1 layer)	0.94M	24.62	24.44	24.53
pLSTM (2 layers)	2.22M	23.19	23.70	23.45

phone-class feature yielded further 7.5% relative improvement. Both of these improvements are statistically significant at the 1% level. We understand that the performance of the LSTMs is as good as the baseline DAEs, although the number of parameters is much smaller (1/25-1/8). It is noteworthy that the improvement obtained by the augmentation with the phone-class feature was larger for the LSTM (7.5%) than for the DAE (6.5%).

5. Conclusion

This paper presented our initial results for the LSTM-based dereverberation with the phone-class feature. The LSTM architecture achieved comparable dereverberation performance to the strong baseline DAE with an order of magnitude smaller number of parameters. We also show that the phone-class feature in the long context effectively improved the performance.

6. References

[1] A.E.Rosenberg, C.H.Lee, and F.K.Soong, "Cepstral channel normalization techniques for HMM-based speaker verification," in *ICSLP*, 1994, pp. 1835–1838.

[2] M.Gurelli and C.Nikias, "Evam: An eigenvector-based algorithm for multichannel blind deconvolution of input colored signals," *IEEE Trans. Audio, Speech & Language Process.*, vol. 43, no. 1, pp. 134–149, 1995.

[3] M.Delcroix, T.Hikichi, and M.Miyoshi, "On the use of lime dereverberation algorithm in an acoustic environment with a noise source," in *ICASSP*, vol. 1, 2006.

[4] S.Gannot and M.Moonen, "Subspace methods for multimicrophone speech dereverberation," in *EURASIP J .Appl. Signal Process.*, vol. 11, 2003, pp. 1074–1090.

[5] M.Wu and D.Wang, "A two-stage algorithm for one-microphone reverberant speech enhancement," *IEEE Trans. Audio, Speech & Language Process.*, vol. 14, no. 3, pp. 774–784, 2006.

[6] K.Kinoshita, M.Delcroix, T.Nakatani, and M.Miyoshi, "Suppression of late reverberation effect on speech signal using long-term multiplestep linear prediction," *IEEE Trans. Audio, Speech & Language Process.*, vol. 17, no. 4, pp. 534–545, 2009.

[7] T.Ishii, H.Komiyama, T.Shinozaki, Y.Horiuchi, and S.Kuroiwa, "Reverberant speech recognition based on denoising autoencoder," in *INTERSPEECH*, 2013, pp. 3512–3516.

[8] X. Feng, Y. Zhang, and J. Glass, "Speech Feature Denoising and Dereverberation via Deep Autoencoders for Noisy Reverberant Speech Recognition," in *Proc. ICASSP*, 2014, pp. 1778–1782.

[9] M.Mimura, S.Sakai, and T.Kawahara, "Exploiting Deep Neural Networks and Deep Autoencoders in Reverberant Speech Recognition," in *HSCMA*, 2014.

[10] ——, "Deep autoencoders augmented with phone-class feature for reverberant speech recognition," in *Proc. ICASSP*, 2015, pp. 4365–4369.

[11] S.Hochreiter, Y.Bengio, P.Frasconi, and J.Schmidhuber, *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*, S. C. Kremer and J. F. Kolen, Eds. IEEE Press, 2001.

[12] K.Kinoshita, M.Delcroix, T.Yoshioka, T.Nakatani, E.Habets, R.Haeb-Umbach, V.Leutnant, A.Sehr, W.Kellermann, R.Maas, S.Gannot, and B.Raj, "The reverb challenge: A common evaluation framework for dereverberation and recognition of reverberant speech," in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA-13)*, 2013.

[13] S.Hochreiter and J.Schmidhuber, "Long Short-Term Memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] Y.Fan, Y.Qian, F.Xie, and F.K.Soong, "Tts synthesis with bidirectional lstm based recurrent neural networks," in *INTERSPEECH*, 2014, pp. 1964–1968.

[15] R.Fernandez, A.Rendel, B.Ramabhadran, and R.Hoory, "Prosody contour prediction with long short-term memory, bi-directional, deep recurrent neural networks," in *INTERSPEECH*, 2014, pp. 2268–2272.

[16] F. Weninger, S. Watanabe, Y. Tachioka, and B. Schuller, "Deep Reccurent De-noising Auto-encoder and Blind De-reverberation for Reverberated Speech Recognition," in *Proc. ICASSP*, 2014, pp. 4656–4660.

[17] A.Graves, A.Mohamed, and G.Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. ICASSP*, 2013, pp. 6645–6649.

[18] A.Graves, N.Jaitly, and A.Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *Proc. ASRU*, 2013, pp. 273–278.

[19] H.Sak, A.Senior, and F.Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *INTERSPEECH*, 2014, pp. 338–342.

[20] H.Sutskever, O.Vinyals, and Q.V.Le, "Sequence to sequence learning with neural networks," in *NIPS*, 2014, pp. 3104–3112.

[21] J.Gonzalez-Dominguez, I.Lopez-Moreno, H.Sak, J.Gonzalez-Rodriguez, and P.J.Moreno, "Automatic language identification using long short-term memory recurrent neural networks," in *INTERSPEECH*, 2014, pp. 2155–2159.

[22] J.T.Geiger, Z.Zhang, F.Weninger, B.Schuller, and G.Rigoll, "Robust speech recognition using long short-term memory reccurent neural networks for hyblid acoustic modelling," in *INTERSPEECH*, 2014, pp. 631–635.

[23] E.Marchi, G.Ferroni, F.Eyben, L.Gabrielli, S.Squartini, and B.Schuller, "Multi-resolution linear prediction based features for audio onset detection with bidirectional lstm neural networks," in *Proc. ICASSP*, 2014, pp. 2183–2187.

[24] R.Brueckner and B.Schuller, "Social signal classification using deep blstm recurrent neural networks," in *Proc. ICASSP*, 2014, pp. 4856–4860.

[25] M.Sundermeyer, R.Schulüter, and H.Ney, "rwthlm - the rwth aachen university neural network language modeling toolkit," in *INTERSPEECH*, 2014, pp. 2093–2097.

[26] M.Sundermeyer, Z.Tüske, R.Schulüter, and H.Ney, "Lattice decoding and rescoring with long-span neural network language models," in *INTERSPEECH*, 2014, pp. 661–665.

[27] G.E.Hinton, L.Deng, D.Yu, G.Dahl, A.Mohamed, N.Jaitly, A.Senior, V.Vanhoucke, P.Nguyen, T.Sainath, and B.Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[28] A.Mohamed, G.Dahl, and G.Hinton, "Acoustic modelling using deep belief networks," *IEEE Trans. Audio, Speech, & Language Proc.*, vol. 20, no. 1, pp. 14–22, 2012.

[29] G.E.Dahl, D.Yu, L.Deng, and A.Acero, "Context-dependent pre-trained deep neural networks for large vocabulary speech recognition," *IEEE Trans. Audio, Speech, & Language Proc.*, vol. 20, no. 1, pp. 30–42, 2012.

[30] F.Seide, G.Li, and D.Yu, "Conversational speech transcription using context-dependent deep neural networks," in *INTERSPEECH*, 2011, pp. 437–440.

[31] L. Deng, M. Seltzer, D. Yu, A. Acero, A. rahman Mohamed, and G. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *INTERSPEECH*, 2010, pp. 1692–1695.

[32] X.Lu, Y.Tsao, S.Matsuda, and C.Hori, "Speech enhancement based on deep denoising autoencoder," in *INTERSPEECH*, 2013, pp. 436–440.

[33] J. Du, Q. Wang, T. Gao, Y. Xu, L. Dai, and C.-H. Lee, "Robust speech recognition with speech enhanced deep neural networks," in *INTERSPEECH*, 2014, pp. 616–620.

[34] G.E.Hinton and R.R.Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, pp. 504–507, 2006.

[35] Y.Bengio, P.Lamblin, D.Popovici, and H.Larochelle, "Greedy layer-wise training of deep networks," in *in Advances in Neural Information Processing Systems 19 (NIPS06)*, 2007, pp. 153–160.