Mimura *et al. EURASIP Journal on Advances in Signal Processing* (2015) 2015:62
DOI 10.1186/s13634-015-0246-6

EURASIP Journal on Advances in Signal Processing
a SpringerOpen Journal

RESEARCH Open Access

Reverberant speech recognition combining deep neural networks and deep autoencoders augmented with a phone-class feature

Masato Mimura*, Shinsuke Sakai and Tatsuya Kawahara

Abstract
We propose an approach to reverberant speech recognition adopting deep learning in the front-end as well as back-end of a reverberant speech recognition system, and a novel method to improve the dereverberation performance of the front-end network using phone-class information. At the front-end, we adopt a deep autoencoder (DAE) for enhancing the speech feature parameters, and speech recognition is performed in the back-end using DNN-HMM acoustic models trained on multi-condition data. The system was evaluated through the ASR task in the Reverb Challenge 2014. The DNN-HMM system trained on the multi-condition training set achieved a conspicuously higher word accuracy compared to the MLLR-adapted GMM-HMM system trained on the same data. Furthermore, feature enhancement with the deep autoencoder contributed to the improvement of recognition accuracy especially in the more adverse conditions. While the mapping between reverberant and clean speech in DAE-based dereverberation is conventionally conducted only with the acoustic information, we presume the mapping is also dependent on the phone information. Therefore, we propose a new scheme (pDAE), which augments a phone-class feature to the standard acoustic features as input. Two types of the phone-class feature are investigated. One is the hard recognition result of monophones, and the other is a soft representation derived from the posterior outputs of monophone DNN. The augmented feature in either type results in a significant improvement (7–8 % relative) from the standard DAE.

Keywords: Reverberant speech recognition; Deep Neural Networks (DNN); Deep Autoencoder (DAE)

1 Introduction

In recent years, the automatic speech recognition (ASR) 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 with reverberation and background noise, which are commonly observed in home and public space, is still at low levels. A key breakthrough for the ASR technology to be accepted widely in the society will be the methodology for hands-free input. This is also critical for realizing conversational robots. Speech reverberation adversely influences the ASR accuracy in such conditions and various efforts have been made to solve this problem.

Reverberant speech recognition has been tackled by feature enhancement at the front-end and model adaptation at the back-end. 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. A major back-end approach is the use of maximum-likelihood linear regression (MLLR) [2] that adapts the acoustic model parameters to the corrupted speech.

More sophisticated enhancement techniques for ASR have been investigated. Speech enhancement techniques include deconvolution approaches that reconstruct clean speech by inverse-filtering reverberant speech [3–5] and spectral enhancement approaches that estimate and remove the influences of the late reverberation [6, 7]. Since an improvement measured by SNR may not be

*Correspondence: mimura@ar.media.kyoto-u.ac.jp
Academic Center for Computing and Media Studies, Kyoto University, Sakyo-ku, 606-8501 Kyoto, Japan

directly related to the ASR accuracy, there also are approaches to speech enhancement based on speech recognition likelihoods in the back-end [8].

Recently, following the great success of deep neural networks (DNN), speech dereverberation by deep autoencoders (DAE) has been investigated [9–13]. 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. DAE can effectively and flexibly learn mapping from corrupted speech to the original clean speech based on the deep learning scheme.

In this paper, we propose to use deep learning both in the front-end (DAE-based dereverberation) and back-end (DNN-HMM acoustic model) in a reverberant speech-recognition system. Recognition of reverberant speech is performed combining "standard" DNN-HMM [14] decoding and a feature enhancement through deep autoencoder (DAE) [9, 10, 15]. The combination of the DNN classifier and the DAE can be regarded as a single DNN classifier with a very deep structure. However, we can expect a mutually complementary effect from the combination of the two networks that are optimized toward different targets.

We also propose a new scheme (pDAE) for DAE-based front-end dereverberation, in which the input vector consisting of acoustic features is augmented with phone-class features. While DAE-based dereverberation is conventionally conducted only with the acoustic information, we presume that the mapping from reverberant speech to clean speech is also dependent on the phone information. Since each dimension of the acoustic feature such as filter-bank output has a different range of values depending on phones, the information on "which phone-class the current speech frame belongs to" should be helpful for DAE to recover the clean speech from reverberant speech. We investigate the following two types of the phone-class features: soft and hard features. We evaluate the effect of these features in the training and recognition stage.

After a brief review on DNNs for reverberant speech recognition (DAE front-end and DNN-HMM back-end) and our baseline system combining the two networks, the detail of the proposed DAE augmented with the phone class feature is explained in Section 3. Experimental evaluations of the method are presented in Section 4 before the conclusion in Section 5.

2 DNN for reverberant speech recognition

2.1 DNN-HMM

Pattern recognition by neural networks has a long history [16]. In recent years, deep neural networks (DNN) have been drawing much attention again in the pattern-recognition field thanks to the establishment of an effective pre-training method [17] and the dramatic increase of computing power and also the available training data. It has been applied to ASR combined with hidden Markov models (HMM) and reported to achieve significantly higher accuracy than the conventional GMM (Gaussian Mixture Model)-HMM scheme in various task domains [14, 18–20].

In the first place, each layer of the network is trained as a restricted Boltzmann machine (RBM) independently. Next, these RBMs are stacked together to constitute a deep belief network (DBN). An initial DNN is then established by adding a randomly initialized softmax layer. This DNN is trained in a supervised way through error backpropagation using HMM state IDs as labels.

The activation vector of the l-th hidden layer $\boldsymbol{x}^l$ and the i-th output x_i^{output} are calculated by

$$\boldsymbol{x}^l = \frac{1}{1+\exp\left(-\left(W^l\boldsymbol{x}^{l-1}+\boldsymbol{b}^l\right)\right)}, \tag{1}$$

and

$$x_i^{output} = \frac{\exp\left(W_{i,*}^{output}\boldsymbol{x}^L+\boldsymbol{b}_i^L\right)}{\sum_j \exp\left(W_{j,*}^{output}\boldsymbol{x}^L+\boldsymbol{b}_j^L\right)}, \tag{2}$$

respectively. Here, W^l and $\boldsymbol{b}^l$ are the weight matrix and the bias vector of the l-th hidden layer, respectively. W^{output} and $\boldsymbol{b}^{output}$ are the weight matrix and the bias vector of the softmax output layer, respectively. x^0 corresponds to the input acoustic feature vector of the DNN.

There have so far been two typical ways to combine DNNs and HMMs. In one approach, the state emission probabilities are computed using DNNs instead of the conventional GMMs. In the other approach, the output from DNNs are used as input to conventional GMM-HMMs. The former is called the hybrid approach [14, 19, 20], and the latter is called the tandem approach [21–23]. In this paper, we adopt the hybrid approach, which has a simple structure and therefore is easy to build. We call these acoustic models built with the hybrid approach as *DNN-HMM* hereafter in this paper. Our DNN-HMM models are built using the standard recipe described in [14].

One of the advantages of the DNN-HMM is that they are suited for handling multiple frames, which is vital especially for reverberant speech recognition where we need to handle long-term artifacts.

2.2 Speech feature enhancement by deep autoencoders (DAE)

The DNN structure described in the previous section can be utilized as a deep autoencoder (DAE) when trained for a different target [24]. In this case, the lower layers are regarded as an encoder to obtain an efficient code and the upper layers are regarded as a decoder that "reverses" the encoder. As a whole, a DAE has a vertically symmetric network structure (Fig. 1). Unlike DNNs for classification, DAEs are typically trained to reconstruct signals using error backpropagation with the mean-squared error as the loss function [25].

Initialization by RBM training is very important in DAEs as well. However, each of the networks in the decoder layers are initialized with the same RBM of the counterpart in the encoder layer. In the decoder layers, network weights are initialized as the transpose of those used for the corresponding encoder layer network, and biases are initialized using visible biases from RBMs instead of hidden biases that are used for the encoder layers. We use the identity function as the output function of the DAE, and the output activation of the DAE is calculated as

$$\boldsymbol{x}^{output} = W^L \boldsymbol{x}^{L-1} + \boldsymbol{b}^L \tag{3}$$

The DAE-based speech processing was pioneered by [9], and DAEs have been applied to front-end feature enhancements in robust speech recognition [10–12, 15]. DAEs for speech enhancement are trained using the clean speech features as target and the corrupted speech features as input (denoising autoencoders [26]). DAE-based dereverberation has also been investigated recently [10–12]. It is shown that networks with a deep structure can effectively and flexibly learn mapping from corrupted speech to the original clean speech based on the deep learning scheme. However, in these pioneering works, the speech recognition experiments were conducted using traditional GMM-HMM systems. Most recently, the system combining the front-end dereverberation based on deep learning and the DNN-HMM back-end is being investigated [13, 27].

2.3 Combination of DAE front-end and DNN-HMM back-end

We first investigate using deep learning both in front-end (DAE-based dereverberation) and back-end (DNN-HMM acoustic model) in reverberant speech recognition [13]. The combination of the DNN classifier and the DAE can be regarded as a single DNN classifier with a very deep structure (Fig. 2). We can expect a mutually complementary effect from the combination of the two networks that are optimized toward different targets. The DNN part of the DNN-HMM and the DAE are trained separately using the same multi-condition data.

Since the dereverberation using the DAE is performed not on the STFT level ([10]) but the feature level ([11, 12]) in our system, we can feed the DAE output to the DNN-HMM acoustic model without any feature extraction process. However, we have options with regard to adding delta parameters and splicing the context frames.

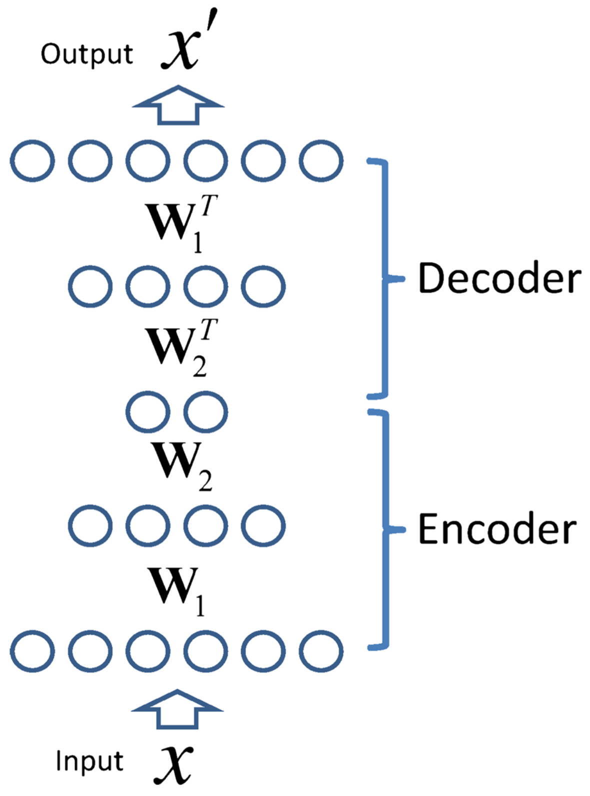

Fig. 1 Deep auto encoder

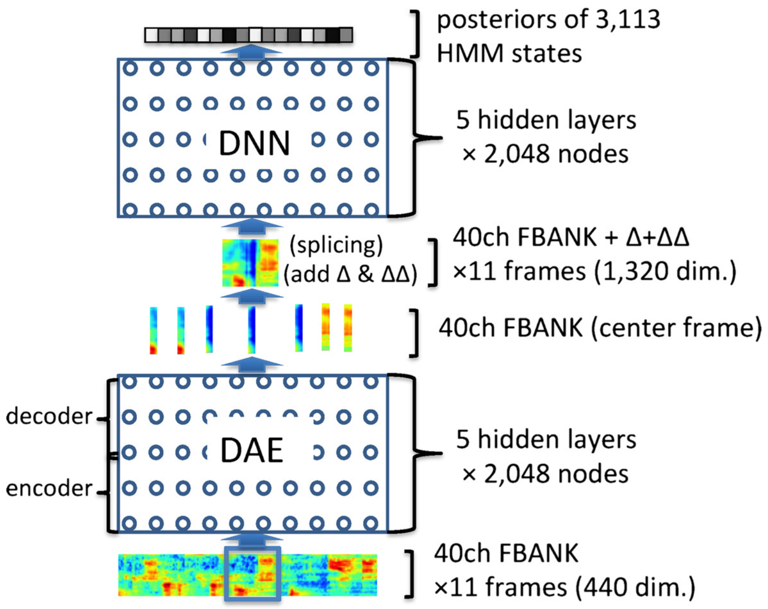
posteriors of 3,113 HMM states
DNN
5 hidden layers × 2,048 nodes
(splicing) (add Δ & ΔΔ)
40ch FBANK + Δ+ΔΔ ×11 frames (1,320 dim.)
40ch FBANK (center frame)
decoder
encoder
DAE
5 hidden layers × 2,048 nodes
40ch FBANK ×11 frames (440 dim.)

Fig. 2 Baseline system combining the DAE front-end and DNN-HMM back-end

The input vector of DNN consists of multiple frames of filterbank outputs to handle long-term information. It is also important to add delta and acceleration parameters to the feature vector of each frame for obtaining better ASR performance as shown in the later section. On the other hand, it may be more difficult to learn network parameters when the target vector of DAE has a larger dimension. To manage this tradeoff, we investigate three options for DAE target.

In the first option, we use only the static part in the filterbank feature of the center frame as a DAE target. In this setting, while the number of dimensions of the DAE output is very small, we need to calculate delta parameters of DAE outputs and splice multiple frames of the resulting vectors before feeding to the DNN-HMM acoustic model. In the second option, the DAE outputs delta and acceleration parameters of filterbank features of the center frame along with static parameters. Here, we still need to splice the DAE outputs. In the third one, the DAE outputs multiple frames of features including delta and acceleration parameters simultaneously. When using this option, we can directly input the DAE outputs to the DNN-HMM acoustic model.

3 DAE augmented with a phone-class feature

3.1 pDAE using phone-class information

Recently, extension of DNN-HMM is investigated by augmenting input with additional information, for example speaker adaptation using I-Vectors [28, 29] and noise-aware training using noise information [30]. Saon et al. [28] proposed a method to train a single network that conducts speaker adaptation and phone classification simultaneously by feeding I-Vectors (speaker identity features) to the network. Considering that there are variations in the acoustic features from different speakers caused by the factors such as variations in vocal tract lengths, the speaker-identity features are helpful for the normalization of the variations.

In this work, we propose to use a *phone-class feature* as an additional input of the DAE to enhance the dereverberation performance. Since the acoustic features in clean speech vary depending on phones, the phone-class information is expected to be helpful for the DAE to recover the clean speech from corrupted speech. We refer to this proposed DAE as *pDAE* (Fig. 3). The training procedure of pDAE is the same as the standard DAE, except that the input is augmented with an additional phone-class feature of the center frame of the input. The activation vector of the first hidden layer for frame t is calculated as

$$\boldsymbol{x}^1(t) = \frac{1}{1 + \exp\left(-\left(W_a^1 \boldsymbol{x}^0(t) + W_p^1 \boldsymbol{p}(t) + \boldsymbol{b}^1\right)\right)} \quad (4)$$

Here, $\boldsymbol{p}(t)$ is the phone class feature for frame t. The weight of the first hidden layer consists of two different matrices; W_a^1 is applied to the standard acoustic feature $\boldsymbol{x}^0(t)$, and W_p^1 is to the phone class feature $\boldsymbol{p}(t)$.

The concept of the proposed method is similar to the stochastic matching proposed by Sankar et al. [31], where the feature normalization is conducted by a function that depends on the phone information. In this work, the mapping is done by the more general deep-learning scheme (DAE).

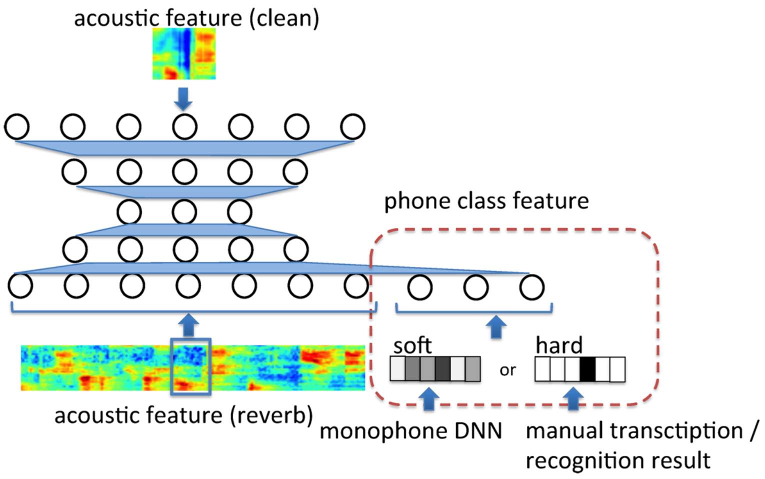
acoustic feature (clean)
phone class feature
soft
or
hard
acoustic feature (reverb)
monophone DNN
manual transctiption / recognition result

Fig. 3 Feature enhancement by pDAE using a phone-class feature

3.2 Phone-class features

The following two types of phone-class features are investigated in this study: soft and hard features.

The soft phone-class feature PC_{soft} is a soft representation of phone classification results. It is defined as a vector containing all the phone state posteriors for the frame calculated with a DNN trained for phone state classification. We use monophone state posteriors (135-dimensional) instead of triphone state posteriors (3113-dimensional) which are used in the acoustic model in order to keep the dimension of the input vector not much larger than the original vector (440-dimensional). The monophone DNN is trained using the same multi-condition data used for the triphone DNN training. pDAE using the PC_{soft} feature is illustrated in Fig. 4. This is similar to the MLP-derived features used in the tandem approach [21–23].

A hard phone-class feature PC_{hard} is defined as the most probable phone state for the frame and is encoded using the 1-of-K scheme. The element corresponding to the phone state which has the largest posterior probability is 1, and all other elements are 0. We can derive the PC_{hard} from PC_{soft} by identifying the phone state with the highest posterior probability.

Another hard feature is PC^{oracle}_{hard}, and it is defined as the "ground truth" phone state for the frame. It is also encoded using the 1-of-K scheme, where the element corresponding to the correct state is 1 and others are set to be 0. With the training data for which manual transcription is available, we can compute the PC^{oracle}_{hard} features from the forced-state alignments using the manual transcripts.

In the recognition time, the "ground truth" phone sequence for the input utterance is not available and, therefore, PC^{oracle}_{hard} features cannot be obtained. Instead, we can use an approximate version of it that we call PC^{decode}_{hard}. We first recognize the input without using a DAE and obtain a best word sequence hypothesis. Then, the phone HMM state labels for all the input frames are generated by performing forced alignment using the word sequence hypothesis. These state labels are converted into PC^{decode}_{hard} features in the same manner as PC^{oracle}_{hard}. The PC^{decode}_{hard} feature is expected to be more reliable than the PC_{hard} feature which is computed frame by frame because it take advantage of the initial recognition results generated using triphone models as well as the language model. However, computation of the PC^{decode}_{hard} features requires an extra decoding pass and may not be suitable for online real-time processing. Note that it is not straightforward to define a soft feature that has to do with all the monophone states from the one-best or even N-best recognition results.

From the definitions above, the soft feature PC_{soft} and the hard features, PC_{hard}, PC^{oracle}_{hard}, and PC^{decode}_{hard} all have the same number of dimensions, which is the number of different states in the monophone GMM-HMM. Therefore, we can try various combinations of phone class features for training and recognition stages. In Section 4.5, we investigate the best combination of them.

4 Experimental evaluations

4.1 Task and data set

The proposed system was evaluated following the instructions for the ASR task of the Reverb Challenge 2014 [32]. 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 data. The amount of the training data is 15.5 h (7861 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 data to make the resulting SNR

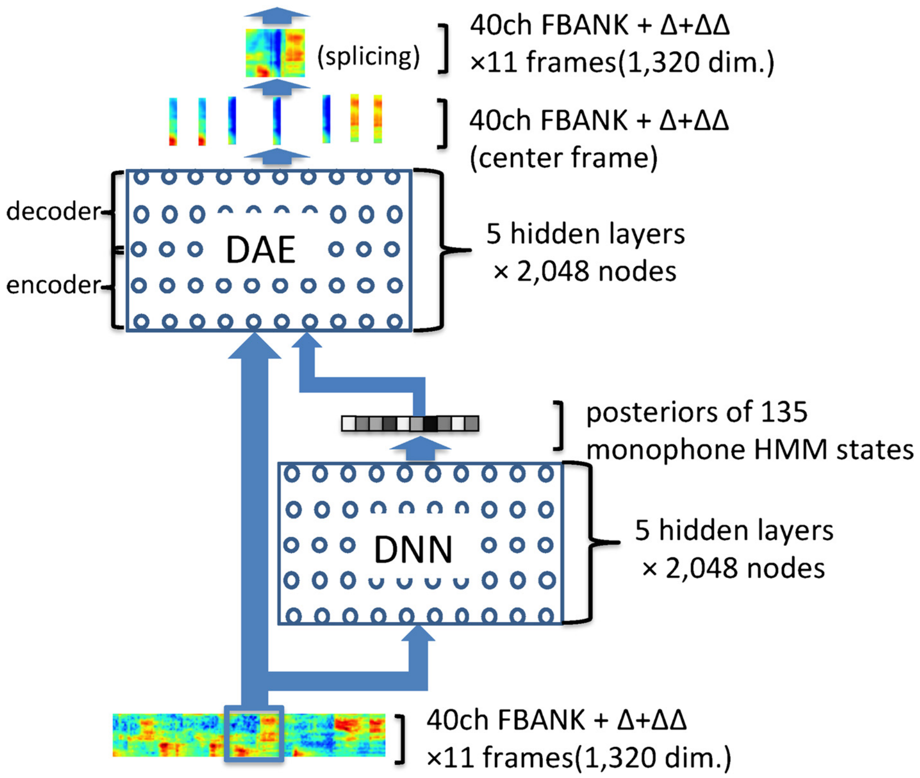
(splicing)
40ch FBANK + Δ+ΔΔ ×11 frames(1,320 dim.)
40ch FBANK + Δ+ΔΔ (center frame)
decoder
encoder
DAE
5 hidden layers × 2,048 nodes
posteriors of 135 monophone HMM states
DNN
5 hidden layers × 2,048 nodes
40ch FBANK + Δ+ΔΔ ×11 frames(1,320 dim.)

Fig. 4 Feature enhancement by pDAE using PC_{soft} feature

to be 20 dB. RIRs were recorded in three different-sized rooms (small, medium, and large) and with two microphone distances (near = 50 cm and far = 200 cm). The reverberation time (T60) of the small, medium, and large rooms are about 0.25, 0.5, and 0.7 s, respectively. These rooms are different from those for measuring RIRs used in generating multi-condition training data. RealData was recorded in a different room from those used for measuring RIRs for SimData. It has a reverberation time of 0.7 s. There are two microphone distances in RealData, which are near (≈100 cm) and far (≈250 cm). Utterance texts for both SimData and RealData were chosen from WSJ-CAM0 prompts. All the reverberant speech recordings were made with eight microphones.

In the experiments in this paper, we only use a single channel both for training and testing. The speech recognition performance is measured by word error rate (WER) in a 5K vocabulary speech-recognition task. For training, we used the 7861 utterances of multi-condition data, which was also the training data for multi-condition baseline GMM-HMM. The training tools for the DNN-HMM and DAE were implemented using Python. We used the CUDAMat library [33] to perform matrix operations on a GPU for speeding up the training procedures. 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[1].

The baseline triphone GMM-HMMs have 3113 shared states, and each state has 10 Gaussian-mixture components. The acoustic features are MFCCs. The results obtained with the GMM-HMM systems trained using the clean data and the multi-condition data (GMM-HMM (cln) and GMM-HMM (mc)) are shown in row 1 and 2 of Table 1, respectively. While the performance with the clean GMM-HMM is very low in the adverse conditions, it is effectively enhanced by the multi-condition training. The results with the MLLR-adapted multi-condition GMM-HMM system are shown in Table 1, row 3. MLLR adaptation was conducted using all utterances of each test condition, which is defined by the combination of the room size and the microphone distance. The model adaptation also improved the performance in all reverberant conditions.

4.2 DNN-HMM

Here, we describe the details of the DNN-HMM system we used for the evaluation experiments.

A 1320-dimensional feature vector consisting of eleven frames of 40-channel log Mel-scale filter bank outputs (LMFB) and their delta and acceleration coefficients is used as the input to the network. The targets are chosen

Table 1 Speech recognition performance on reverb challenge 2014 test set (WER (%))

		SimData							RealData		
		Room 1		Room 2		Room 3		Ave.	Room 1		Ave.
		Near	Far	Near	Far	Near	Far		Near	Far	
(1)	GMM-HMM (cln)	10.79	16.11	33.62	81.72	43.97	88.35	45.92	87.93	85.92	86.95
(2)	GMM-HMM (mc)	14.38	14.06	15.03	28.85	19.06	35.57	21.16	46.79	45.44	46.13
(3)	GMM-HMM (mc, w MLLR)	12.39	12.71	14.23	26.23	17.11	33.92	19.43	42.89	42.27	42.59
(4)	DNN-HMM (cln)	6.85	10.22	16.18	45.52	23.12	60.25	27.05	65.25	66.78	65.99
(5)	DNN-HMM (cln) + DAE	6.25	6.78	7.65	13.67	9.04	16.75	10.03	30.66	31.87	31.25
(6)	DNN-HMM (cln) + pDAE (PC_{soft})	5.51	6.44	7.06	12.74	8.17	14.26	9.04	27.37	26.60	27.00
(7)	DNN-HMM (cln) + pDAE (PC_{hard}^{decode})	5.18	6.12	7.14	12.57	7.66	12.42	8.54	27.75	26.60	27.20
(8)	DNN-HMM (mc)	5.42	6.37	7.27	12.56	7.85	12.90	8.74	28.59	30.87	29.67
(9)	DNN-HMM (mc) + DAE	9.30	9.69	8.36	11.92	9.30	15.25	10.62	24.37	25.52	24.93
(10)	DNN-HMM (mc) + pDAE (PC_{soft})	8.59	9.13	7.77	11.53	8.74	13.53	9.87	23.47	23.09	23.29
(11)	DNN-HMM (mc) + pDAE (PC_{hard}^{decode})	7.29	7.86	7.48	10.87	8.09	11.06	8.78	22.74	22.96	22.85
(12)	DNN-HMM (retrain) + DAE	6.10	6.32	7.04	13.04	6.89	13.50	8.83	31.30	32.14	31.71

to be the 3113 shared states of the baseline GMM-HMMs. The six-layer network consists of five hidden layers and a softmax output layer. Each of the hidden layers consists of 2048 nodes. The network is initialized using the RBMs trained with reverberant speech.

The fine-tuning of the DNN is performed using cross entropy as the loss function through error backpropagation supervised by state IDs for frames. The mini-batch size for the stochastic gradient descent algorithm was set to be 256. The learning rate was set to be 0.08 initially, and it was halved if the improvement in the frame accuracies on the held-out set between two consecutive epochs fell below 0.2 %. The momentum was set to be 0.9. The training was stopped after 20 epochs. The state labels for the frames were generated by the forced alignment of clean data with HVite command of HTK3.4 using the baseline GMM-HMM acoustic model. The HMM model parameters other than emission probabilities such as transition probabilities were copied from the baseline GMM-HMM.

The WER for the evaluation data set obtained with the DNN-HMM system trained using multi-condition data ("DNN-HMM(mc)") are shown in the row 8 of Table 1. For all subsets of the "SimData" of the evaluation set, the DNN-HMM system achieved drastically higher accuracies than the adapted GMM-HMM system. In the most adverse condition (Room 3, Far), the WER was reduced by 21.0 points (from 33.92 to 12.90 %). For the "RealData", the DNN-HMM system achieved a higher accuracy than the the adapted GMM-HMMs by 12.9 points.

The DNN-HMM system was trained on the clean training set as well as the multi-condition training set. The WER obtained with this clean DNN-HMM ("DNN-HMM(cln)") system are shown in row 4 of Table 1. The accuracies by the clean DNN-HMMs are drastically lower than the multi-condition DNN-HMMs. We see that the multi-condition training is effective for DNN-HMMs as well as GMM-HMMs from these results.

4.2.1 Evaluation for clean test data

We also performed evaluation experiments on clean speech ("ClnData"). The WER obtained with the baseline GMM-HMM systems are shown in rows 1 through 3 of Table 2. The results with the multi-condition DNN-HMM system is shown in row 5. We see that the accuracies for clean speech deteriorate significantly with the GMM-HMMs trained using multi-condition data. Meanwhile, the results obtained by DNN-HMMs trained using multi-condition data were much better than those with the GMM-HMMs trained using clean data.

The accuracies by the clean DNN-HMMs (Table 2, row 4) were better than the multi-condition DNN-HMMs, although the difference between them were not as large as the difference between the clean and multi-condition GMM-HMM systems. These results show that DNN-HMM trained with the multi-condition data is robust against mismatch of the input condition.

Table 2 System performance on clean data (WER (%))

	ClnData			
	Room 1	Room 2	Room 3	Ave.
Baseline (cln, w/o MLLR)	7.79	8.26	7.80	7.96
Baseline (mc, w/o MLLR)	18.60	18.57	17.80	18.33
Baseline (mc, w MLLR)	13.88	14.01	13.44	13.78
DNN-HMM (cln)	4.02	4.47	4.38	4.29
DNN-HMM (mc)	6.25	6.40	6.38	6.35

4.2.2 Importance of delta feature

To confirm the importance of delta and acceleration parameters in DNN-based acoustic modeling for reverberant speech recognition, we evaluated the DNN-HMM system trained with only the static part of the acoustic feature of the multi-condition data.

As shown in the Table 3, removing the delta and acceleration parameters caused significant performance degradation in both of the real and simulated conditions, although the input vector contains LMFB features of 11 context frames. These results clearly show that delta and acceleration parameters are important. Therefore, we decided to add delta and acceleration coefficients to the outputs of the DAE front-end in some way before feeding to the DNN-HMM acoustic model.

4.2.3 Matched condition training

In general, multi-condition training is an effective strategy, since the run-time reverberation condition is unknown in the system development stage. However, the part of the training data with mismatched reverberation conditions to the run time may cause an adverse effect. We could expect better performance if we prepare models trained with single reverberation conditions and choose a best-matched model at the run time in some way, although the choice of the best-matched model at run time itself is non-trivial.

To see the possible effectiveness of this approach, we trained DNNs with simulated training data that would match the "RealData" part of the evaluation data. We generated two simulated training-data sets using the RIR for "Near" and "Far" microphone distances in a "Large" room.

Table 3 Importance of delta and acceleration parameters (WER (%))

	SimData	RealData
DNN-HMM (mc, w delta)	8.74	29.67
DNN-HMM (mc, w/o delta)	10.73	33.05

Each of the resulting training sets has the same size as the whole multi-condition training set. The experimental results are shown in Table 4.

The accuracy for "RealData"-"Far" is improved with the "Large"-"Far" model. However, the accuracies for both conditions in "RealData" are drastically degraded with the "Large"-"Near" model. Although both labeled "Near", the microphone distances of 50 cm in "Large"- "Near" (training) and 100 cm in "RealData"-"Near" (test) seem to have made a big difference in the reverberated speech.

From these preliminary experimental results, we see that the recognition performance can be improved with the collection of single condition models, when a single-condition model matches the run-time condition very well. However, when the selected single-condition model is not matched well, it does not perform as well as the multi-condition model.

4.3 Denoising deep autoencoder (DAE)

Here, we describe the detail of the DAE front-end. The input of the DAE was set to be the eleven-frame sequence of 40-channel LMFB features (440-dimensional). The target for the DAE was one frame (40-dimensional) of the clean speech which corresponds to the center frame of the input. Other target options will be investigated in Section 4.3.3.

The DAE is fine-tuned using reverberant speech as the input and clean speech as the target. The input frames and the output frames for the training were adjusted to be time-aligned in the multi-condition training-data generation process. The last portions of reverberant speech utterance files exceeding the length of the clean speech were trimmed to equalize the lengths of input and output.

The DAE has six layers in total. The number of the layers was determined using a development set. The number of nodes in each layer is set to be 2048 except for input and output layers[2]. The network is initialized using the same RBMs as used for initializing the DNNs described in the last subsection, which were trained using reverberant speech. The lower three layers were initialized using the weights of the first three RBMs and the hidden unit biases. The upper three layers were initialized using the transpose of the weights mentioned above and the visible unit biases.

Table 4 Performances of single-condition DNN-HMMs on RealData (WER (%))

	RealData	
	Room 1	
	Near (= 100 cm)	Far (= 250 cm)
DNN-HMM (mc)	28.59	30.87
DNN-HMM (Large Near (= 50 cm))	37.18	39.13
DNN-HMM (Large Far (= 200 cm))	28.71	27.95

While the RBM we used for the initialization of the output layer originally has 40 * 11 nodes in the upper layer, we used only the 40 nodes corresponding to the center frame.

The fine-tuning of the DAE was performed by error backpropagation with squared error as the loss function. The parameters such as the mini-batch size and the momentum are set to be the same as those for DNN training. However, the initial learning rate was set to be 0.001, which is smaller than the one for DNN training. The learning rate was halved if the improvement in the frame accuracies on the held-out set between consecutive two epochs fell below 0.2 %. The frame accuracies were calculated using the "DNN-HMM (cln)" and the features enhanced with the network obtained at each epoch end.

The evaluation results with the combination of the DAE and the clean DNN-HMM are shown in Table 1, row 5. The accuracies are drastically improved in all conditions from the clean DNN-HMM without DAE (row 4 of the same table). The DAE has done an effective feature enhancement as expected. Interestingly, these results are comparable to those from the multi-condition DNN-HMM without DAE (Table 1, row 8).

The results with the combination of the DAE and the multi-condition DNN-HMM are shown in Table 1, row 9. The combination of the DAE front-end and the multi-condition DNN-HMM significantly improved the WER for very adverse "RealData" conditions, while it was not effective for "SimData" conditions, which have similar RIRs to the training data. Since SimData is very similar to the training data, either of the DAE front-end or the multi-condition DNN-HMM back-end is sufficient for dereverberation, and the simultaneous use of the both results in significant degradation in the accuracies. On the other hand, RealData has different characteristics from the training data, and the combination of the DAE and the multi-condition DNN-HMM has a complementary effect.

Figure 5 shows an example of DAE-enhanced utterances. The smearing in the spectral pattern along with the time axis is ameliorated in the DAE-enhanced features.

4.3.1 MFCC vs. LMFB

It is widely accepted that the LMFB feature is more effective in DNN-based acoustic modeling than the cepstrum feature such as MFCC. But it is not clear if LMFB is the best choice for the DAE-based front-end dereverberation. Therefore, here we compared the MFCC and LMFB features. The MFCC feature consists of 12-dimensional MFCC, ΔMFCC, $\Delta\Delta$MFCC, power, Δpower, and $\Delta\Delta$power. The recognition results obtained by MFCC-based and LMFB-based systems are shown in Table 5.

When used without the DAE-based front-end, the accuracy with the DNN-HMM (mc) was degraded significantly by using MFCC as the acoustic feature (from row 3 to

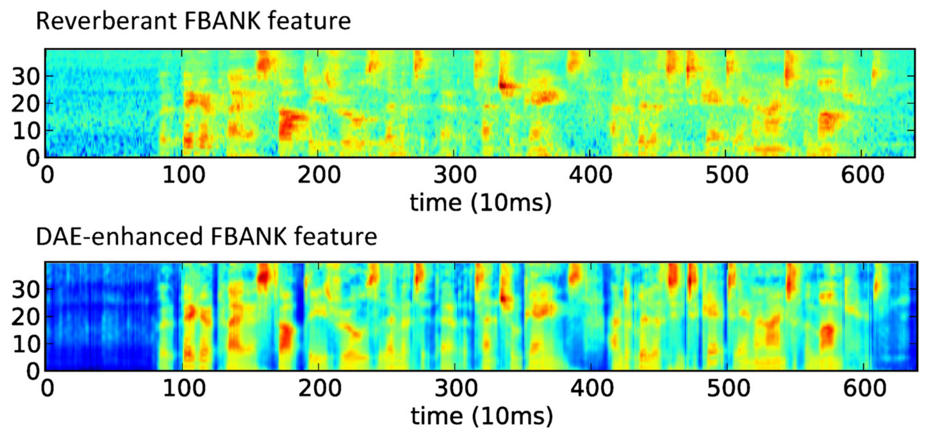

Fig. 5 Example of DAE-enhanced utterance

row 1). More importantly, when using MFCC, the DAE front-end yielded little improvement (from row 1 to row 2), while the improvement by the DAE was significant in the LMFB-based system (from row 3 to row 4). From these results, we understand that it is essential to use the LMFB feature as the acoustic feature in the combined system.

4.3.2 DNN-HMM retrained with DAE output

The speech-feature parameters enhanced by the DAE may have different characteristics from the original reverberant speech. Therefore, we retrained the DNN using the DAE output and performed speech recognition experiments. This time, the RBMs for initializing the network were also trained using the DAE output as training data.

The WER obtained using this retrained network is shown in Table 1, row 12. We see that improvement of the accuracy is observed for "SimData", but not for "RealData". The DAE is trained using the simulated reverberant speech data generated in the same manner as SimData. Therefore, DNN-HMM retrained using the multi-condition speech data processed through the DAE is best matched to the DAE-processed SimData test set. That is the reason why it outperforms 5 and 9 where DNN-HMMs were trained using the unmatched training data. However, it does not work well with the RealData, because the RealData has much different characteristics (room size and microphone distance) from the training data and the SimData, and the retrained model may have over-fitted to the training data.

Table 5 Comparison of acoustic feature (WER (%))

	Feature	SimData	RealData
DNN-HMM (mc)	MFCC	10.97	32.97
DNN-HMM (mc) + DAE	MFCC	11.80	32.51
DNN-HMM (mc)	LMFB	8.74	29.67
DNN-HMM (mc) + DAE	LMFB	10.62	24.93

4.3.3 Autoencoder target options

In the experiment above, the DAE was trained with only the center frame of static LMFB feature parameters as target. However, as described in Section 3.1, we have some target options. Here, we compare the following three types of DAE: DAE trained with the 11 frames of LMFB feature with delta and acceleration parameters ("DAE(1320)"), DAE trained with the center frame of the LMFB feature with delta and acceleration parameters ("DAE(120)"), and DAE trained with the center frame of LMFB feature without delta and acceleration parameters ("DAE(40)"). The recognition results obtained with these three DAEs are shown in Table 6.

Clearly different tendencies were observed in "SimData" and "RealData". In "SimData", which has similar RIR to the training data, as the number of dimensions of the target (therefore output) increased, the accuracy is slightly improved. On the other hand, in "RealData", increasing the target dimensions caused significant performance degradation. From these results, we understand that the DAE with the smallest number of target dimensions is the least susceptible to over-fitting and the most robust against unknown reverberation conditions, while it requires additional processes for adding delta parameters and splicing context frames, which are computationally inexpensive.

Table 6 Comparison of autoencoder target options (WER (%))

	SimData	RealData
DNN-HMM (cln) + DAE (40)	10.03	31.25
DNN-HMM (cln) + DAE (120)	9.29	33.61
DNN-HMM (cln) + DAE (1320)	9.28	34.68
DNN-HMM (mc) + DAE (40)	10.62	24.93
DNN-HMM (mc) + DAE (120)	9.70	26.32
DNN-HMM (mc) + DAE (1320)	9.65	28.21

4.4 DAE augmented with phone class feature (pDAE)

Then, we evaluated the dereverberation performance of the proposed DAE augmented with the phone-class feature. The training procedure for pDAE is the same as that for the baseline DAE, except that the input vector is augmented with the 135-dimensional phone-class feature. The input layer of pDAE has 575 (440 + 135) nodes. The mean and variance of the phone-class feature vector are normalized in the same manner as the filterbank feature. The frame accuracy obtained on the held-out set during the fine-tuning of the pDAE and the baseline DAE is shown in Fig. 6. Here, we used PC_{soft} as the phone-class feature. The frame accuracy was calculated using the clean DNN-HMM back-end and outputs of the DAE and the pDAE at the end of each epoch. We observe that pDAE is consistently better than the baseline DAE, suggesting that feeding the phone-class information to the DAE is effective.

The evaluation results with the combination of the pDAE (PC_{soft}) and the clean DNN-HMM back-end are shown in Table 1, row 6. In all "SimData" conditions, the WER was reduced from the baseline DAE (Table 1, row 5). In more adverse "Far" conditions in "Room 2" and "Room 3", the improvements are larger. In "RealData" conditions, the WER was reduced by 4.25 points from the baseline DAE, and the improvement is significantly higher than in "SimData". The phone-class information is more effective when the mismatch between the training data and the test data is larger.

The WER with the combination of the pDAE (PC_{soft}) and the multi-condition DNN-HMM back-end is shown in Table 1, row 10. In all "SimData" conditions, the performance degradation observed in the combination of the multi-condition DNN-HMM and the standard DAE front-end (Table 1, row 9) was mitigated by using the pDAE. In "RealData", the WER was reduced by 1.64 points from the baseline DAE, confirming that the phone information is effective even when using the multi-condition DNN-HMM back-end, which is more robust for reverberant speech. The improvement from the standard DAE in both "SimData" and "RealData" conditions is statistically significant at the 1 % level.

4.5 Comparison of soft and hard phone-class features

Next, we compared the two types of the phone-class features described in Section 3.2. We evaluated six different combinations of the features in the training and recognition stage through speech-recognition experiments on "RealData" using the multi-condition DNN-HMM back-end.

Comparison of PC_{soft} and PC_{hard}^{oracle} in the training stage is shown in the two columns in Table 7. The dereverberation performance of the pDAE is degraded by using the PC_{hard}^{oracle} feature in the training stage, whichever type of the phone-class feature is used in the recognition stage, although the PC_{hard}^{oracle} feature is more accurate than the PC_{soft} feature. One of the reasons for this may be that the PC_{soft} feature has richer information.

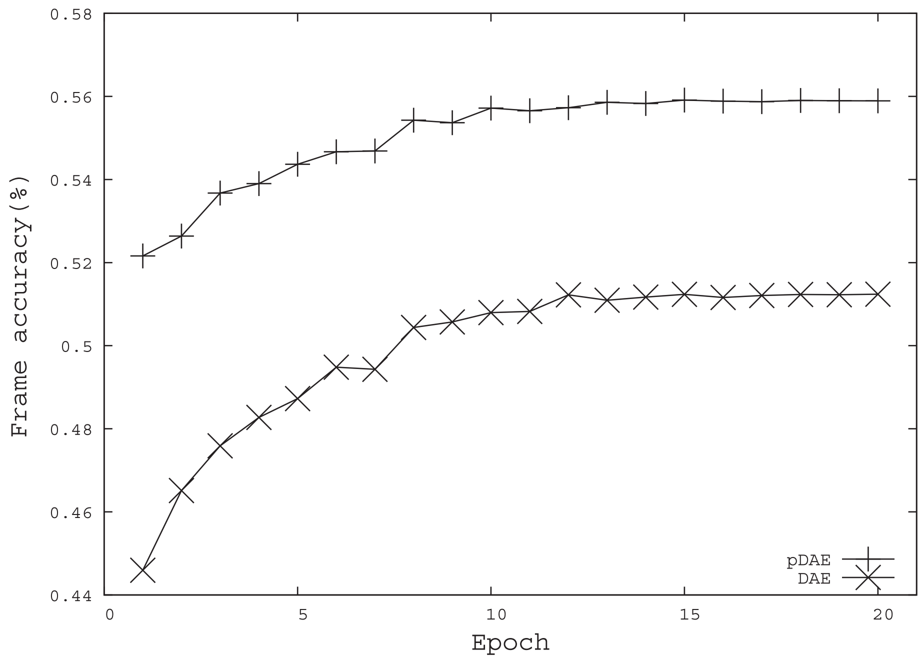

Fig. 6 Frame accuracy by DAE and pDAE on held-out set

Table 7 Comparison of phone-class features (WER (%))

Recognition\Training	PC_{soft}	PC_{hard}^{oracle}
PC_{soft}	23.29	24.10
PC_{hard}	23.27	24.34
PC_{hard}^{decode}	22.85	23.29
(cf.) PC_{hard}^{oracle}	13.74	14.25

We also conducted an oracle experiment where we used the PC_{hard}^{oracle} feature derived from the manual transcription of the test data. As shown in the last row of Table 7, the WER was surprisingly reduced, which clearly confirms our hypothesis that phone-class information is useful for the DAE-based dereverberation. However, the hard version of the PC_{soft} feature (PC_{hard}) did not yield any improvement.

On the other hand, the results with the PC_{hard}^{decode} feature derived from the initial recognition result is better than those with the PC_{soft} feature as expected, though it requires another recognition pass. The WER in all reverberant conditions including "SimData" obtained with the PC_{hard}^{decode} feature is shown in row 7 and 11 in Table 1. Compared with the results of the PC_{soft} feature (row 6 and 10), the combination with the multi-condition DNN results in further WER reduction in all conditions.

5 Conclusions

In this paper, we investigated an approach to reverberant speech recognition adopting deep learning in the front-end as well as back-end of the system and evaluated it through the ASR task (one channel) of the Reverb Challenge 2014.

The DNN-HMM system trained with the multi-condition training set achieved a conspicuously higher word accuracy compared with the MLLR-adapted GMM-HMM system trained with the same data. Furthermore, feature enhancement with the DAE contributed to the improvement of recognition accuracy especially in the more adverse conditions.

We have also proposed a novel approach to reverberant speech recognition based on the DAE augmented with the phone-class information. The proposed method significantly and consistently improved the recognition accuracy in all reverberant conditions. We compared two types of the phone-class feature and concluded that the PC_{soft} feature, which does not require an extra decoding step, is enough for significant improvement while using the PC_{hard}^{decode} feature in the recognition stage can yield further improvement. It is also shown that using the PC_{soft} feature is more effective than the PC_{hard}^{oracle} feature in the training phase.

The average WER on "RealData" obtained with the proposed pDAE using the PC_{hard}^{decode} feature (Table 1, row 11) was 1.0 points better than the best result in the same condition ("1ch", "no own data", "no full batch") of the Reverb Challenge 2014.

Endnotes

[1]The WERs in this paper are much lower than those in the results we submitted to the Reverb Challenge 2014 (http://reverb2014.dereverberation.com/result_asr.html) mainly because the trigram language model was used in this paper and the bigram model was used in the Challenge.

[2]We conducted small preliminary experiments to introduce a bottleneck layer with a fewer number of units, but the recognition performance was degraded.

Competing interests
The authors declare that they have no competing interests.

Acknowledgements
This work was supported by JST CREST and ERATO programs.

Received: 17 February 2015 Accepted: 8 July 2015
Published online: 23 July 2015

References

1. AE Rosenberg, CH Lee, FK Soong, in *ICSLP*. Cepstral channel normalization techniques for HMM-based speaker verification (ISCA, Yokohama, Japan, 1994), pp. 1835–1838
2. CJ Leggetter, PC Woodland, in *Computer Speech and Lang*. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models, vol. 9 (Academic Press Inc, 1995), pp. 171–185
3. M Gurelli, C Nikias, Evam: An eigenvector-based algorithm for multichannel blind deconvolution of input colored signals. IEEE Trans. Audio, Speech Lang. Process. **43**(1), 134–149 (1995)
4. M Delcroix, T Hikichi, M Miyoshi, in *ICASSP*. On the use of lime dereverberation algorithm in an acoustic environment with a noise source, vol. 1 (IEEE, Toulouse, France, 2006)
5. S Gannot, M Moonen, in *EURASIP J. Appl. Signal Process*. Subspace methods for multimicrophone speech dereverberation, vol. 11 (Springer, 2003), pp. 1074–1090
6. M Wu, D Wang, A two-stage algorithm for one-microphone reverberant speech enhancement. IEEE Trans. Audio Speech Lang. Process. **14**(3), 774–784 (2006)
7. K Kinoshita, M Delcroix, T Nakatani, M Miyoshi, Suppression of late reverberation effect on speech signal using long-term multiplestep linear prediction. IEEE Trans. Audio, Speech Lang. Process. **17**(4), 534–545 (2009)
8. R Gomez, T Kawahara, Robust speech recognition based on dereverberation parameter optimization using acoustic model likelihood. IEEE Trans. Audio Speech Lang. Process. **18**(7), 1708–1716 (2010)
9. L Deng, M Seltzer, D Yu, A Acero, A-r Mohamed, G Hinton, in *INTERSPEECH*. Binary coding of speech spectrograms using a deep auto-encoder (ISCA, Makuhari, Japan, 2010), pp. 1692–1695
10. T Ishii, H Komiyama, T Shinozaki, Y Horiuchi, S Kuroiwa, in *INTERSPEECH*. Reverberant speech recognition based on denoising autoencoder (ISCA, Lyon, France, 2013), pp. 3512–3516
11. X Feng, Y Zhang, J Glass, in *Proc. ICASSP*. Speech feature denoising and dereverberation via deep autoencoders for noisy reverberant speech recognition (IEEE, Florence, Italy, 2014), pp. 1778–1782
12. F Weninger, S Watanabe, Y Tachioka, B Schuller, in *Proc. ICASSP*. Deep reccurent de-noising auto-encoder and blind de-reverberation for reverberated speech recognition (IEEE, Florence, Italy, 2014), pp. 4656–4660
13. M Mimura, S Sakai, T Kawahara, in *HSCMA,*. Exploiting deep neural networks and deep autoencoders in reverberant speech recognition (IEEE, Nancy, France, 2014)

14. GE Dahl, D Yu, L Deng, A Acero, Context-dependent pre-trained deep neural networks for large vocabulary speech recognition. IEEE Trans. Audio Speech Lang. Proc. **20**(1), 30–42 (2012)
15. X Lu, Y Tsao, S Matsuda, C Hori, in *INTERSPEECH*. Speech enhancement based on deep denoising autoencoder (ISCA, Lyon, France, 2013), pp. 436–440
16. CM Bishop, *Neural networks for pattern recognition*. (Oxford University Press, Oxford, 1995)
17. GE Hinton, S Osindero, Y Teh, A fast learning algorithm for deep belief nets. Neural Comput. **18**, 1527–1554 (2006)
18. GE Hinton, L Deng, D Yu, G Dahl, A Mohamed, N Jaitly, A Senior, V Vanhoucke, P Nguyen, T Sainath, B Kingsbury, Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Process. Mag. **29**(6), 82–97 (2012)
19. A Mohamed, G Dahl, G Hinton, Acoustic modelling using deep belief networks. IEEE Trans. Audio Speech Lang. Proc. **20**(1), 14–22 (2012)
20. F Seide, G Li, D Yu, in *INTERSPEECH*. Conversational speech transcription using context-dependent deep neural networks (ISCA, Florence, Italy, 2011), pp. 437–440
21. N Morgan, Deep and wide: Multiple layers in automatic speech recognition. IEEE Trans. Audio Speech Lang. Proc. **20**(1), 7–13 (2012)
22. GSVS Sivaram, H Hermansky, Sparse multilayer perceptron for phoneme recognition. IEEE Trans. Audio Speech Lang. Proc. **20**(1), 23–29 (2012)
23. PJ Bell, MJF Gales, P Lanchantin, X Liu, Y Long, S Renals, P Swietojanski, PC Woodland, in *Proc. SLT*. Transcriptions of multi-genre media archives using out-of-domain data (IEEE, Miami, USA, 2012), pp. 324–329
24. GE Hinton, RR Salakhutdinov, Reducing the dimensionality of data with neural networks. Science. **313**, 504–507 (2006)
25. Y Bengio, P Lamblin, D Popovici, H Larochelle, in *Advances in Neural Information Processing Systems 19 (NIPS06)*. Greedy layer-wise training of deep networks (MIT Press, 2007), pp. 153–160
26. P Vincent, H Larochelle, Y Bengio, PA Manzagol, in *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML'08)*. Extracting and composing robust features with denoising autoencoders (ACM Helsinki, Finland, 2008), pp. 1096–1103
27. J Du, Q Wang, T Gao, Y Xu, L Dai, C-H Lee, in *INTERSPEECH*. Robust speech recognition with speech enhanced deep neural networks (ISCA, Singapore, 2014), pp. 616–620
28. G Saon, H Soltau, D Nahamoo, M Picheny, in *Proc. ASRU*. Speaker adaptation of neural network acoustic models using I-Vevtors (IEEE, Olomouc, Czech, 2013), pp. 55–59
29. P Karanasou, Y Wang, MJF Gales, PC Woodland, in *INTERSPEECH*. Adaptation of deep neural network acoustic models using factorised i-vectors (ISCA, Singapore, 2014), pp. 616–620
30. M Seltzer, D Yu, Y Wang, in *Proc. ICASSP*. An Investigation of deep neural networks for noise robust speech recognition (IEEE, Vancouver, Canada, 2013), pp. 7398–7402
31. A Sankar, C-H Lee, A maximum-likelihood approach to stochastic matching for robust speech recognition. IEEE Trans. Speech Audio Process. **4**(3), 190–202 (1996)
32. K Kinoshita, M Delcroix, T Yoshioka, T Nakatani, E Habets, R Haeb-Umbach, V Leutnant, A Sehr, W Kellermann, R Maas, S Gannot, B Raj, in *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA-13)*. The reverb challenge: a common evaluation framework for dereverberation and recognition of reverberant speech (IEEE, New York, USA, 2013)
33. V Mnih, in *Department of Computer Science, University of Toronto, Tech. Rep. UTML TR*. Cudamat: a CUDA-based matrix class for python (University of Toronto, 2009)