EURASIP Journal on Audio, Speech, and Music Processing
a SpringerOpen Journal

RESEARCH Open Access

Single-channel dereverberation by feature mapping using cascade neural networks for robust distant speaker identification and speech recognition

Aditya Arie Nugraha[1*], Kazumasa Yamamoto[1,2] and Seiichi Nakagawa[1]

Abstract

We present a feature enhancement method that uses neural networks (NNs) to map the reverberant feature in a log-melspectral domain to its corresponding anechoic feature. The mapping is done by cascade NNs trained using Cascade2 algorithm with an implementation of segment-based normalization. Experiments using speaker identification (SID) and automatic speech recognition (ASR) systems were conducted to evaluate the method. The experiments of SID system was conducted by using our own simulated and real reverberant datasets, while the CENSREC-4 evaluation framework was used as the evaluation for the ASR system. The proposed method could remarkably improve the performance of both systems by using limited stereo data and low speaker-variant data as the training data. From the evaluation using SID, we reached 26.0% and 34.8% of error rate reduction (ERR) relative to the baseline by using simulated and real data, respectively, by using only one pair of utterances for matched condition cases. Then, by using combined dataset containing 15 pairs of utterances by one speaker from three positions in a room, we could reach 93.7% of average identification rate (three known and two unknown positions), which was 42.2% of ERR relative to the use of cepstral mean normalization (CMN). From the evaluation using ASR, by using 40 pairs of utterances as the NN training data, we could reach 78.4% of ERR relative to the baseline by using simulated utterances by five speakers. Moreover, we could reach 75.4% and 71.6% of ERR relative to the baseline by using real utterances by five speakers and one speaker, respectively.

Keywords: Dereverberation; Feature enhancement; Cascade neural network; Stereo training data; Speech recognition; Speaker identification

1 Introduction

The use of distant-talking microphones for automatic speech recognition (ASR) system or automatic speaker identification (SID) system can improve user convenience. The use of such microphones is essential for certain applications, e.g., the application of ASR and/or SID for smart home, where it will be not practical if the users have to hold or wear microphone anytime they want to interact with the system. However, the use of distant-talking microphones will make the captured signal be vulnerable to the phenomenon known as reverberation, where the signal not only travels directly from the speaker to the microphone but also through reflections, which can be seen as delayed and attenuated versions of the direct signal. Thus, reverberation will cause *smearing effect* because the microphone captures the currently spoken utterance along with other utterance spoken in the past [1]. Because of this signal degradation, the use of reverberant signal captured by the microphone will degrade the ASR or SID system performance, which is usually trained using anechoic speech data.

Several approaches already proposed to deal with this reverberation problem from ASR point of view.

*Correspondence: arie@slp.cs.tut.ac.jp
[1]Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Aichi 441-8580, Japan
Full list of author information is available at the end of the article

According to [2] in which the state-of-the-art in reverberant speech processing is discussed, there are two classes of approaches in dealing with reverberation problem, i.e., front-end-based and back-end-based approaches. The front-end-based approaches attempt to remove the effect of reverberation from the observed feature vectors. It can be divided into linear filtering, spectrum enhancement, and feature enhancement. The linear filtering dereverberates time-domain signals or STFT coefficients, e.g., [3,4], the spectrum enhancement dereverberates the corrupted power spectra of signal, e.g., [5-7], and the feature enhancement dereverberates the corrupted feature vectors, e.g., [8-10]. Meanwhile, the back-end-based approaches attempt to modify the acoustic model and/or decoder so that they are suitable for reverberant environment, e.g., [11,12].

Among the front-end-based approaches, there are single-channel approaches (by using single microphone) and multi-channel approaches (by using microphone array). Many recently proposed dereverberation researches focus on the use of microphone array, e.g., multi-channel linear prediction [13], minimum variance distortionless response (MVDR) beamformer [14], multi-channel least mean squares (LMS) [15]. Comparing to the use of single microphone, the main benefit of microphone array is the spatial information it can provide. Despite the benefits of microphone array, the use of single microphone is much easier and cheaper to be implemented for real applications. Thus, the research on single-channel dereverberation method is still worth to be considered.

Many works focused on feature enhancement approach. Several single-channel feature enhancement methods have been proposed. Some of them do not need stereo data at all, e.g., cepstral mean normalization (CMN) [16,17], long-term feature normalization [18], vector Taylor series (VTS) [19], particle filter [8,20], and extended Kalman filter [9,21]. Meanwhile, some of them assume that *stereo training data* can be acquired. In the context of distant SID or ASR system, stereo data are simultaneously recorded pairs of close-talking and distant-talking utterances. In general, the stereo training data is used to train a mapping function from the distant-talking utterance to its corresponding close-talking utterance. Several existing approaches which need stereo training data will be reviewed in Section 2.

This research focused on developing a single-channel dereverberation method for automatic speaker identification and speech recognition under real environmental conditions by doing feature enhancement and assuming that stereo data can be acquired. In order to increase its feasibility for real-world applications, the method should have good performance by using a limited number of stereo data.

We proposed a single-channel non-linear regression-based dereverberation method using cascade neural networks (NNs). The NNs were trained on stereo data to compensate the reverberation effect by mapping a segment of reverberant 24-dimensional log-melspectral feature vectors to its corresponding anechoic feature vector. Two most important parts of the proposed method are the segment-based normalization and the feature mapping using NN. The segment-based normalization is done by normalizing the current frame of the anechoic and the reverberant feature and also preserving the power envelope of reverberant input segment. For the feature mapping, cascade NNs trained using the Cascade2 algorithm with the Resilient Backpropagation (RPROP) weight update algorithm, which is a variation of batch backpropagation algorithm, were used. These two most important parts are most likely the reason why the proposed method could generalize and perform remarkably well for a limited number of stereo data (one or five pairs of utterances; corresponds to less than 1 min of utterance).

The proposed method was evaluated on SID and ASR systems. Both evaluations were done by using simulated and real data. The evaluation using SID system used our own simulated and real data, while the evaluation using ASR system used simulated and real data from CENSREC-4 [22]. The proposed method could perform very well by using only few stereo data and also low speaker-variant data as the NN training data.

The experimental result of SID using simulated data shows that in matched condition cases, the error rate reduction (ERR) relative to the baseline by using only *one pair of utterances* could reach 26.0% when single NN ('1 NNs') configuration was used. Meanwhile, by using 15 pairs of utterances and multiple NNs ('24 NNs') configuration, the ERR reached 62.6%. Also in matched condition cases, the experimental result using real data shows that by using '6 NNs' configuration and only *one pair of utterances* for the NN training data, we could reach 34.8% of ERR relative to the use of CMN. Then, by combining the training data from known positions in a room, we could train NNs which performed well for unknown positions in the same room. By using multiple NNs ('24 NNs') and 15 pairs of utterances by one speaker from three positions, we could reach 93.7% of average identification rate over three known and two unknown positions, which was 42.2% of ERR relative to the use of CMN.

The experimental result of ASR using simulated data shows that we could reach 78.4% of ERR relative to the baseline by using dataset containing 40 pairs of utterances by five speakers as the NN training data. Meanwhile, by using the same number of real reverberant data, we could reach 75.4% and 71.6% of ERR relative to the baseline by using dataset containing utterances by five speakers and one speaker, respectively.

2 Related works

The most popular feature enhancement method using stereo data is stereo-based piece-wise linear compensation for environments (SPLICE), which estimates the clean cepstral feature from the noisy feature using a Gaussian mixture model (GMM) of noisy feature [23]. In general, SPLICE tries to represent a non-linear relation by using piece-wise linear relation in each subspace of noisy feature. SPLICE could perform well for both simulated and real noisy data [24]. However, SPLICE is designed specifically for dealing with the noise problem.

For the reverberation problem, in [25], 13 multi-layer perceptron (MLP) NNs were trained using stereo data to map the 13-dimensional reverberant cepstral feature, where one NN was used for one dimension of feature, to its corresponding anechoic feature. The input of each NN was a sequence of cepstral feature coefficients from nine consecutive frames, and the output was a cepstral feature coefficient. The approach was evaluated using vector quantization (VQ)-based speaker identification method and could reach 80.2% of ERR relative to baseline.

In [26], a linear regression by least squares method (LSM) was used to do a mapping of melspectral feature vectors from a four-frame sequence of reverberant speech to a frame of clean speech. Several schemes of dynamic time warping (DTW) were introduced because there was only non-stereo dataset for the experiments. The non-stereo dataset contained close-talking utterances recorded from the distance of 25 cm and distant-talking utterances recorded from various positions in a room. These close- and distant-talking utterances were not recorded simultaneously, although the speakers and the utterances were the same. The DTW was used to align frames of a distant-talking utterance to frames of its corresponding close-talking utterance before they were used as the training data. Nonetheless, the approach should be also worked on stereo dataset.

In [10,27], a joint sparse representation (JSR) technique was used to capture the relationship between clean and reverberant speech. The dictionary for clean feature space and the dictionary for reverberant feature space were jointly trained using the stereo data in order to have common representation coefficients. Basically, the approach did a mapping of log-melspectral feature vectors from N frames of reverberant speech to N frames of estimated clean speech. In [27], besides the 24-dimensional log-melspectral feature vectors, the mapping included the log-energy coefficients. In the same paper, the sequence of N frames included the use of left and right context (past, current, and future frames).

Stereo data also found to be used in linear filtering approaches. In [28], the dereverberation was done using linear and binary-weighted least squares techniques on time and fast Fourier transform (FFT) domain. The stereo data was needed to calculate the inverse filter coefficients, which then was used to transform N frames of reverberant complex-valued FFT coefficient vectors to N frames of estimated clean FFT coefficient vectors. The experiments were done using 512-, 1,024-, and 2,048-dimensional vectors. The length of vector corresponds to the length of DFT.

Recently, in [29,30], a denoising autoencoder (DAE), which is one of deep neural network (DNN) approaches, was used to do a mapping of coefficient vectors from a sequence of reverberant speech to a sequence of clean speech. They also introduced the use of short and long window. The short window is used to extract 256 dimensions of power spectral coefficients and the log energy. On the other hand, the long window is used to extract 24 dimensions of melspectral coefficients and the log energy. Thus, by using both windows, the DNN was used to map from and to 2,538-dimensional vectors, which are constructed by power spectral, melspectral, and log-energy coefficients of a nine-frame segment. In addition, DAE was also used for speech enhancement by mapping the power spectral coefficients [31] and the melspectral coefficients [32]. These DNN-based approaches are effective, but they require much training data for training a huge number of parameters.

In summary, the approaches proposed in [25,26] did a mapping from a N-frame segment to a one-frame segment. Meanwhile, the approaches proposed in [10,27-30] did a mapping from a N-frame segment to a N-frame segment. The method proposed in this work does a mapping from a N-frame segment to a one-frame segment of log-melspectral coefficients by using cascade NNs and requires only few training data. The NN is used because it should be able to capture a non-linear relation across the frames, which is caused by the insufficiency of analysis window (frame) length in capturing the reverberation effect and other complex factors.

3 Overview of neural network

3.1 Artificial neural network

NN, or more properly called artificial neural network (ANN), is a computational model inspired by the biological nervous systems, such as the brain. In a simple way, a biological nervous system consists of interconnected webs of neurons, where each neuron has dendrites, soma, and axon. The dendrites receive input signals and when the soma feels that the input signals are strong enough, it emits an output signal through the axon. This signal then can be sent to other neuron's dendrites through the synapses, which are the end points of axon's branches.

How a biological neuron works is modeled by an artificial neuron, as depicted by Figure 1. The neuron has inputs x_n with their associated weights w_n. The weighted

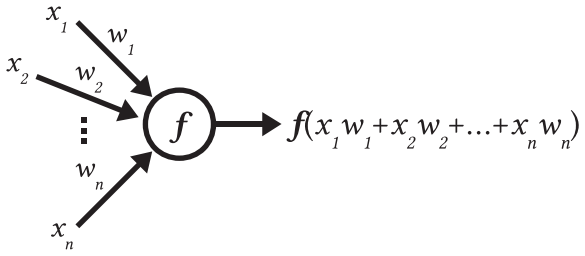

Figure 1 Artificial neuron [33].

inputs are integrated in the neuron, which in most cases is simply done by summation, and then evaluated by an activation function f, e.g., hyperbolic tangent function, to determine the output. An ANN is simply a network that consists of interconnected artificial neurons and has three important elements, i.e., the structure of the nodes/neurons (how the inputs are integrated, the activation function), topology/architecture (the way artificial neurons interconnected), and the training algorithm (to determine the weights in the network) [33].

Various types of NN are defined based on their architectures. The architecture itself is a combination of their framework and their interconnection scheme [34].

The framework is defined by the number of clusters and the number of neurons in each cluster. The clusters are called layers if they are ordered and are called slabs otherwise. There are input, hidden, and output clusters, where each cluster contains one or more neurons. The neurons within a cluster are not necessarily ordered.

The interconnection scheme is mainly defined by the connectivity (describes which neurons are connected) and the types of connections. In layered NN, connections can be divided into interlayer, intralayer, and supralayer connections. Interlayer connection connects neurons from adjacent layers, intralayer connection connects neurons within a layer, and supralayer connection connects neurons from different (non-adjacent) layers. In slabbed NN, where the clusters are not ordered, there are only interslab connection, which connects neurons from different slabs, and intraslab connection, which connects neurons within a slab. Further, in regard to the directionality, connections can be divided into symmetric (bidirectional) and asymmetric (unidirectional) connections.

3.2 Conventional multi-layer perceptron and cascade networks

Both conventional MLP and the cascade networks used in the proposed method may use the same structure of neurons. The main differences between them are in the architecture and how to build the architecture, which then cause a difference in the training algorithm.

3.2.1 Conventional MLP network

In conventional MLP approach, the NN is fully defined in advance before the training is started. The NN is a layered NN with asymmetric interlayer connections (Figure 2A). The NN contains an input layer, one or more hidden layers, and an output layer. Except the output layer, each layer commonly contains more than one neuron. The training (weight update) algorithm is then used to update the previously initialized weights and determine the most appropriate weights based on the training data. Backpropagation can be regarded as the most popular weight update algorithm for training MLP. It propagates an input through the network, then propagates back the error and adjusts the weights to minimize the error. The algorithm can be used in both an incremental training, in which the weights are updated for each training datum in the training set, and a batch training, in which the weights are updated only after all training data in the training set are presented.

3.2.2 Cascade network

Besides the common approach above, there are dynamic approaches in which the architecture of NN is altered during the training by adding neurons and/or clusters. Thus, the training algorithm not only consists of the weight update algorithm but also consists of the architecture algorithm, e.g., cascade. Two most common cascade algorithms are Cascade-Correlation (CasCor) and Cascade2 [35]. Cascade2 algorithm is a variation of CasCor algorithm. Instead of using covariance maximization as in CasCor, Cascade2 uses direct error minimization. By doing so, Cascade2 is better algorithm for regression task, while CasCor is better for classification task [36].

In cascade algorithm, the NN can be regarded as layered NN with asymmetric interlayer and supralayer connections (Figure 2B). Usually, the NN contains an input layer,

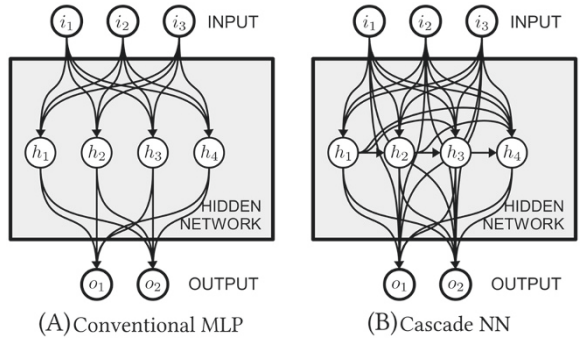

Figure 2 The architectures of conventional MLP and cascade NN depicted in conventional MLP representation. (A) Conventional MLP: layered NN with asymmetric interlayer connections. **(B)** Cascade NN: layered NN with asymmetric interlayer and supralayer connections.

many hidden layers, and an output layer. Different from conventional MLP approach, each hidden layer in cascade algorithm contains one hidden neuron. The other difference is that in the cascade NN, every hidden neuron and also the output neurons are directly connected to the input neurons. Meanwhile, in a conventional MLP, only hidden neurons of the first hidden layer are directly connected to the input neurons and the output neurons are directly connected only to the hidden neurons of the last hidden layer.

Before the training of a cascade NN is started, the NN only contains an input layer and an output layer with interlayer connections connect neurons from these two layers. The NN is then grown by adding hidden neurons/layers during the training. Each newly added hidden neuron connects to the neurons from input layer and output layer. The newly added hidden neuron also connects to the previously added hidden neuron. The hidden neuron addition is controlled by the cascade algorithm and done after the weight update algorithm cannot find a proper weight to generate the correct output by using the existing architecture. The same backpropagation weight update algorithm as for conventional MLP training can also be used for cascade NN training.

According to [37] and [35], the cascade algorithm offers several advantages, including:

- The algorithm will automatically build a reasonably small network, so there is no need to define the NN in advance,
- The algorithm learns fast because it employs weight freezing to overcome moving target problem by training one unit at a time instead of training the whole network at once as in conventional MLP, and
- The algorithm can build deep network (high-order feature representation) without dramatic slowdown as seen in conventional MLP with more than one hidden layers.

The deep network generated by cascade algorithm can represent very strong non-linearity. It is good for some problems but may be bad for other problems. It can be regarded as overfitting problem caused by the use of too many layers and neurons. As explained in [37], CasCor employs 'patience' parameter to stop the training when the error has not changed significantly for a period of time. However, according to [35], overfitting will still occurr if the NN is allowed to grow too much. Therefore, we need to define the proper maximum number of hidden neurons. Besides using less hidden neurons, we can also use more training data to minimize the overfitting possibility.

For further details of cascade NN in general and Cascade2 algorithm in particular, please refer to [35].

4 The estimation function

4.1 Reverberation model

On the time domain, the relation between anechoic and reverberant signal (regardless the noise) can be expressed as

$$y(t) = s(t) * h(t), \tag{1}$$

where $s(t)$ and $y(t)$ are the clean and reverberant signals, respectively, and $h(t)$ is the room impulse response (RIR), which defines the room transfer function (RTF).

The relation between anechoic and reverberant signal in log-melspectral domain should be represented as a nonlinear model as shown in [12,21]. However, for simplicity, we defined it as

$$Y(t) = \sum_{i=0}^{N} \alpha_i S(t-i) \tag{2}$$

$$= \alpha_0 S(t) + \sum_{i=1}^{N} \alpha_i S(t-i), \tag{3}$$

where $S(t)$ and $Y(t)$ represent the log-melspectral coefficients of anechoic and reverberant signal, respectively, for frame index t. While, $\alpha_0, \alpha_1, \ldots, \alpha_N$ represent the RTF. This formulation was introduced in [26] and also employed in [38-40].

The first term of Equation 3 corresponds to the direct-path signal captured by the microphone and is represented by the solid line in Figure 3. Meanwhile, the second term corresponds to the sum of signal reflections and are represented by the dotted lines in Figure 3. The reflection can be regarded as an attenuated and delayed version of the direct signal.

4.1.1 Causal model

From Equation 3, $S(t)$ could be expressed as

$$S(t) = \frac{1}{\alpha_0} Y(t) - \sum_{i=1}^{N} \frac{\alpha_i}{\alpha_0} S(t-i), \tag{4}$$

and by recursively substituting the last term, $S(t)$ could be expressed as

$$S(t) = \frac{1}{\alpha_0} Y(t) - \sum_{i=1}^{N} \frac{\alpha_i}{\alpha_0} \frac{1}{\alpha_0} Y(t-i) + \sum_{i=1}^{N} \frac{\alpha_i}{\alpha_0} \sum_{j=1}^{N} \frac{\alpha_j}{\alpha_0} S(t-i-j), \tag{5}$$

$$S(t) = \frac{1}{\alpha_0} Y(t) - \sum_{i=1}^{N} \frac{\alpha_i}{\alpha_0^2} Y(t-i) + \sum_{i=1}^{N} \sum_{j=1}^{N} \frac{\alpha_i \alpha_j}{\alpha_0^3} Y(t-i-j) - \ldots. \tag{6}$$

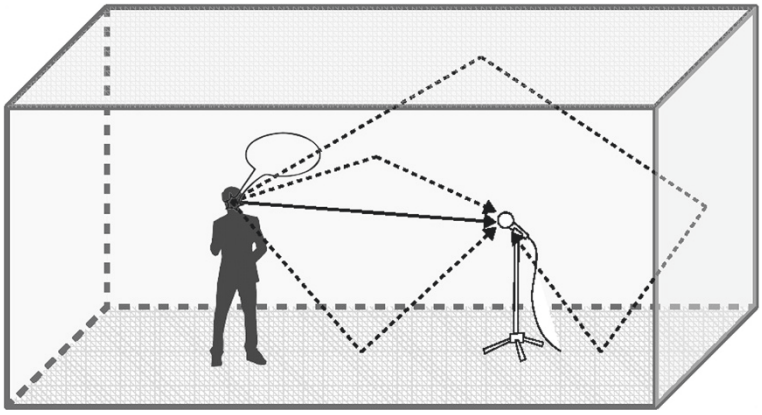

Figure 3 Illustration of reverberation in a closed room. The solid line represents the direct-path signal. The dotted lines represent the reflections.

The first term of Equation 6 considers only the current frame $Y(t)$, while the succeeding terms consider the left context. The second term considers $Y(t-N)$ until $Y(t-1)$, the third term considers $Y(t-2N)$ until $Y(t-2)$, the fourth term considers $Y(t-3N)$ until $Y(t-3)$, and so on.

RIR is characterized by reverberation time (T_{60}), which is the time required for reflections of a direct-path signal to decay by 60 dB or one-millionth of the original energy. If N is selected such that the N-segment covers the T_{60}, we may limit the calculation up to the frame N because the energy of reflections in the frames after N is very low and negligible. If the practicality is considered, N should be a trade-off between the dereverberation performance and the computational cost. A longer segment may capture the reverberation effect better which then potentially improve the dereverberation, but the computational cost for processing this segment will be higher. In order to simplify the equation, β_i is used for representing the variables formed by the combination of α_i, e.g. $\beta_0 = \alpha_0^{-1}$, $\beta_1 = -\alpha_1\alpha_0^{-2}$, $\beta_2 = -\alpha_2\alpha_0^{-2} + \alpha_1^2\alpha_0^{-3}$, $\beta_3 = -\alpha_3\alpha_0^{-2} + (2\alpha_1\alpha_2 - \alpha_1^3)\alpha_0^{-3}$, and so on. Thus, the estimated anechoic coefficient $\hat{S}(t)$ could be expressed as a function of reverberant signal $Y(t)$

$$\hat{S}(t) = \beta_0 Y(t) + \sum_{k=1}^{L} \beta_k Y(t-k) + \varepsilon, \tag{7}$$

$$\hat{S}(t) \approx \sum_{k=0}^{L} \beta_k Y(t-k), \tag{8}$$

where $\beta_0, \beta_1, \ldots, \beta_L$ denotes the weights which are used to compensate the RTF and L denotes the number of past frames in the segment. L is used to substitute N in order to indicate that the frames are the left context. By using Equation 8, we could estimate current source signal $S(t)$ by using an $(L+1)$-frame segment of observed signal consisting of current observed signal $Y(t)$ and L frame(s) of past observed signal.

4.1.2 Non-causal model

By considering a typical RIR, intuitively, we know that the information of current frame will remain in its reflections in the future, especially in its early reflections part where the reflections still have considerable amount of energy.

Let $0 < n < N$ and $N > 1$, Equation 3 can be rewritten as

$$Y(t) = \sum_{i=0}^{n-1} \alpha_i S(t-i) + \alpha_n S(t-n) + \sum_{i=n+1}^{N} \alpha_i S(t-i), \tag{9}$$

and $S(t-n)$ could be expressed as

$$S(t-n) = \frac{1}{\alpha_n} Y(t) - \sum_{i=0}^{n-1} \frac{\alpha_i}{\alpha_n} S(t-i) - \sum_{i=n+1}^{N} \frac{\alpha_i}{\alpha_n} S(t-i). \tag{10}$$

Then, by substituting $S(t-i)$ using the causal model on Equation 8, $S(t-n)$ could be expressed as

$$S(t-n) = \frac{1}{\alpha_n} Y(t) - \sum_{i=0}^{n-1} \frac{\alpha_i}{\alpha_n} \sum_{j=0}^{N} \beta_j Y(t-i-j) - \sum_{i=n+1}^{N} \frac{\alpha_i}{\alpha_n} \sum_{j=0}^{N} \beta_j Y(t-i-j). \tag{11}$$

Equation 11 comprises three terms. The first term considers only $Y(t)$. The second term considers the left context, which is $Y(t)$ until $Y(t-n+1)$; the current frame, which is $Y(t-n)$; and the right context, which is $Y(t-n-1)$ until $Y(t-n-N+1)$. Meanwhile, the third term considers only the right context, which is $Y(t-n-1)$ until $Y(t-2N)$. In order to simplify the equation, γ_i is used to substitute the variables formed by the combination of α_i

and βi. Thus, the estimated anechoic coefficient $\hat{S}(t-n)$ could be expressed as

$$\hat{S}(t-n) = \sum_{k=0}^{n-1} \gamma_k Y(t-k) + \gamma_n Y(t-n) + \sum_{k=n+1}^{2N} \gamma_k Y(t-k) + \varepsilon. \quad (12)$$

Then, by making generalization on Equation 12, the estimated anechoic coefficient $\hat{S}(t)$ could be expressed as

$$\hat{S}(t) = \sum_{k=-R}^{-1} \gamma_k Y(t-k) + \gamma_0 Y(t) + \sum_{k=1}^{L} \gamma_k Y(t-k) + \varepsilon, \quad (13)$$

$$\hat{S}(t) \approx \sum_{k=-R}^{L} \gamma_k Y(t-k), \quad (14)$$

where $\gamma_K, \ldots, \gamma{-}1, \gamma_0, \gamma_1, \ldots, \gamma_L$ denotes the weights which are used to compensate the RTF, L denotes the number of past frames in the segment, and R denotes the number of future frames in the segment. L and R are used to indicate that the frames are the left context and right context, respectively. By using Equation 14, we could estimate current source signal $S(t)$ by using an $(L+1+R)$-frame segment of observed signal consisting of current observed signal $Y(t)$, L frame(s) of past observed signal (left context), and R frame(s) of future observed signal (right context).

Equation 14 could be seen as the general form of reverberation model. We could get Equation 8 from Equation 14 by setting $R = 0$.

Hereafter, we refer to Equation 8 as causal reverberation model and Equation 14 as non-causal reverberation model (for $R > 0$). For the causal reverberation model, the estimation of $\hat{S}(t)$ can be seen as removing the unwanted information (reflections of previous frames) which is estimated from the left context. While, for the non-causal reverberation model, besides removing the unwanted information, the estimation can be seen as gathering more information about the current frame to be processed, which is estimated from the right context.

4.2 Frame selection

Inspired by the use of window skipping in [41], besides Equation 14 we also defined

$$\hat{S}(t) \approx \sum_{k=-R}^{L} \gamma_k Y(t-2k). \quad (15)$$

Hereafter, we refer to Equation 8 as *'linear' frame selection* and Equation 15 as *'skip1' frame selection*.

In this work, we named the frame selection used in the experiments using *L-C-R* notation. *L*, *C*, and *R* show the number of left context (past), current, and right context (future) frames, respectively. For example, frame selection *4-1-0* means that four past and one current frames of reverberant speech are used to estimate the dereverberated version of current frame, and frame selection *4-1-4* means that four past, one current, and four future frames are used to do the dereverberation.

The use of skipping frame selection could be regarded as dimensionality reduction strategy by minimizing the redundant parts caused by the windowing. Therefore, we could get a representation of longer context of time-domain signal using smaller number of frames, which is beneficial for the NN training. For example, if we use 25 ms window with 10 ms shift, 145 ms of context can be represented by using 13-frame linear frame selection, e.g., 12-1-0 or 6-1-6, or 7-frame skip1 frame selection, e.g., 6-1-0 or 3-1-3.

4.3 Assumptions on the log-melspectral feature

In our works, we made several assumptions on the log-melspectral feature. The first assumption is about the dependency of certain feature dimension to the other feature dimensions, and the second is about the RTF of feature dimension.

On the dependency of feature dimension, we defined several *channel selection*, as follows:

- All-dimension selection, where the dimensions are assumed to be fully dependent on each other.
- Single-dimension selection, where the dimensions are assumed to be independent on each other and certain dimension is only affected by the same dimension.
- Neighboring-dimension selection, where certain dimension is assumed to be affected by the same dimension and its neighbor dimension.

The estimation function above was derived using the assumption used in *all-dimension selection*. However, our experiments mainly used the *single-dimension selection*, which can be expressed as

$$\hat{S}_d(t) \approx \sum_{k=-R}^{L} \gamma_{d,k} Y_d(t-k), \text{ for } d = 1, 2, \ldots, D, \quad (16)$$

where d is the feature dimension number and D is the total number of feature dimension.

By using the single-dimension selection, we could define several assumptions on the transformation of feature dimension caused by the RTF. In our case, the assumptions on the transformation affects the number of NNs that should be used. Thus, we defined several *NN configurations* as follows:

- *Single NN* configuration, where it is assumed that the transformation on each dimension is the same as each other, so one NN is used to transform all dimensions of feature.
- *Basic multiple NNs* configuration, where it is assumed that the transformation on each dimension is different from each other, so one NN is used to transform one specific dimension.
- *Modified multiple NNs* configuration, where it is assumed that the transformation on certain several neighboring dimensions are the same, one NN is used to transform *more than* one dimension (neighboring dimensions).

The basic multiple NNs configuration, in which one transformation function (in the form of NN) is used to transform one dimension, corresponds to Equation 16. Meanwhile, the single NN configuration, in which one transformation function (in the form of NN) is used to transform all dimensions, corresponds to Equation 17. Note that Equations 16 and 17 are linear mapping functions, while transformation by NN is a non-linear mapping. As shown in [26], the use of linear mapping is not good enough to do dereverberation.

$$\hat{S}_d(t) \approx \sum_{k=-R}^{L} \gamma_k Y_d(t-k), \text{ for } d = 1, 2, \ldots, D. \tag{17}$$

5 The proposed dereverberation method

Figure 4 shows the block diagram of proposed dereverberation method. In general, the method can be divided into segment-based normalization, feature scaling, and feature mapping using NNs. The inputs of the method are $Y(t-L), \ldots, Y(t-1), Y(t), Y(t+1), \ldots, Y(t+R)$, which are past frames, current frame, and future frames of reverberant log-melspectral coefficient vector, and the output is $\hat{S}(t)$, which is the estimated current anechoic log-melspectral coefficient vector.

5.1 Segment-based normalization

Segment-based normalization is employed to deal with the power difference between the anechoic speech signal and the reverberant signal captured by a distant-talking microphone and to normalize the loudness of speech utterance. In the NN training stage, it is done by normalizing the current reverberant feature vector and the current anechoic feature vector (which is the target of training) to the normalization target. Besides, the segment-based normalization is employed to preserve the relative variation of power envelope in a segment by normalizing the past frames relative to the current frame. The normalization is done using Equations 18 and 19 below.

$$\delta(t) = \Delta - \frac{1}{D} \sum_{d=1}^{D} Y_d(t), \tag{18}$$

$$\bar{Y}_d(t-k) = Y_d(t-k) + \delta(t), \text{ for } d = 1, 2, \ldots, D, \text{ for } -R \leq k \leq L, \tag{19}$$

where $\delta(t)$ is the normalization factor, $\bar{Y}_d(t)$ is the normalized log-melspectral coefficient for feature dimension d and time index t, D is the number of feature dimensions, and Δ is the normalization target.

The mean of NN output $\bar{S}(t)$ should be equal to the normalization target because the target of NN training was also normalized. Therefore, the denormalization (Equation 20) is used to recover its original mean of power.

$$\hat{S}_j(t) = \bar{S}_j(t) - \delta(t). \tag{20}$$

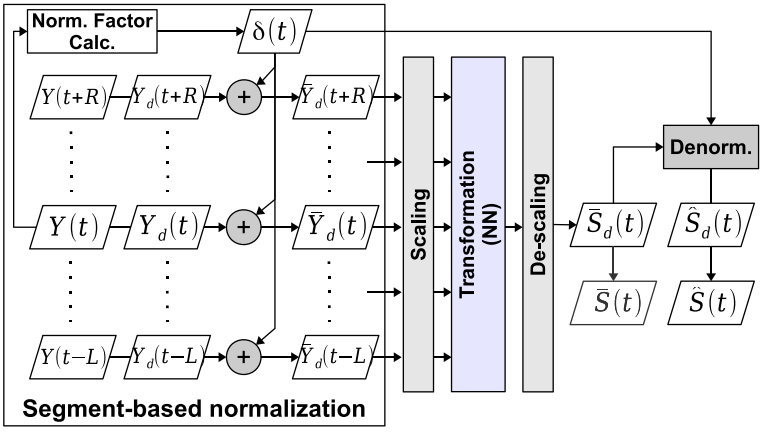

Figure 4 Block diagram of proposed dereverberation method.

Actually, the use of normalization factor $\delta(t)$ calculated from the reverberant feature vector (distant-talking speech utterance) is not the best way to calculate the estimation of clean feature vector (close-talking speech utterance) because the power levels of clean and reverberant signals are most likely not identical due to the influence of the RIR. However, the distant-talking speech utterance is the only input for the dereverberation method, and we assume that the distance is unknown, so the use of normalization factor $\delta(t)$ is the most reasonable way we can do to recover the power level of a frame relative to its surrounding frame. Thus, the estimation $\hat{S}(t)$ in Equation 20 can be regarded as an attenuated close-talking speech utterance. Although, it is not the best approach, the denormalization could remarkably improve the output, as can be seen in Figure 5.

Figure 5 shows spectrograms of an input and outputs of a dereverberation process. Utterance-based normalization (based on maximum value) was done in creating the spectrogram to ease the observation because there is power difference between close-talking speech utterance (clean feature vectors) and distant-talking speech utterance (reverberant feature vectors). Figure 5A shows the feature vectors of close-talking utterance, which was recorded from the distance of 25 cm. Figure 5B shows the feature vector of corresponding distant-talking utterance, which was recorded from the distance of about 4.0 m. Figure 5C,D are the dereverberated feature vectors without denormalization and with denormalization, respectively. We can observe that by using denormalization, we could get better estimation of the clean feature vectors, especially for non-speech segments. In Figure 5C,D, the difference of non-speech segments can be easily observed, for example, between frames 290 and 330 (in frames).

5.2 Feature scaling

The feature scaling consists of scaling and de-scaling processes. In general, the scaling and de-scaling can be regarded as the pre-processing and post-processing for the NNs. The scaling is done so that the NN input and output have values ranging from *about* -1 until 1. The constants τ and κ were used for this purpose, and the value of these constants were determined empirically from preliminary experiments. In contrast, the de-scaling is used to recover the log-melspectral coefficient value from its scaled value. The scaling and de-scaling are done using Equations 21 and 22, respectively.

$$\bar{Y}'_d(t-k) = \frac{\bar{Y}_d(t-k)+\tau}{2^\kappa}, \text{ for } -R \le k \le L, \tag{21}$$

$$\bar{S}_d(t) = \bar{S}'_d(t) * 2^\kappa - \tau. \tag{22}$$

5.3 Feature mapping using neural networks

In matrix form, Equation 14 could be written as

$$\hat{\boldsymbol{S}} = \boldsymbol{G}\,\boldsymbol{Y}, \tag{23}$$

where $\hat{\boldsymbol{S}}$ denotes the estimated anechoic feature vector, $\boldsymbol{Y}$ denotes the supervector which consists of reverberant feature vectors, and $\boldsymbol{G}$ denotes the transformation matrix which represents the RTF compensation. In our works, a non-linear regression is done to determine the function $\boldsymbol{G}$ such that

$$\underset{\boldsymbol{G}}{\operatorname{argmin}} \|\boldsymbol{S} - (\boldsymbol{G} \otimes \boldsymbol{Y})\|^2, \tag{24}$$

where $\boldsymbol{S}$ is the anechoic (reference) feature vector and $\otimes$ denotes a non-linear transformation. The regression is done by NN training algorithm and the NNs resulted from the training are used as the transformation function $\boldsymbol{G}$. Thus, the NNs are the functions for mapping the reverberant feature vectors $\boldsymbol{Y}$ to the anechoic feature vector $\boldsymbol{S}$.

We use cascade NNs trained using the Cascade2 algorithm. The algorithm is chosen because our task is a regression task and Cascade2 is better algorithm for the task than CasCor [36].

Figure 6 shows an illustration of cascade NN that we used in our works, with N input neurons, M hidden neurons, and one output neuron. Figure 7 shows the same NN in conventional MLP representation. The input neurons are represented by $y_1, y_2, \ldots, y_N$, the hidden neurons are represented by $h_1, h_2, \ldots, h_M$, and the output neuron is represented by s. Besides, we also use one bias neuron b. The neurons y_i and b are connected to $h_1, h_2, \ldots, h_M, s$ and the neuron s is connected to $y_1, y_2, \ldots, y_N, h_1, h_2, \ldots, h_M, b$. The connection weight between neuron n_1 and n_2 is represented by $w(n_1, n_2)$.

The NN input $y_1, y_2, \ldots, y_N$ in Figure 6 correspond to $\bar{Y}'_d(t-L), \ldots, \bar{Y}'_d(t-1), \bar{Y}'_d(t), \bar{Y}'_d(t+1), \ldots, \bar{Y}'_d(t+R)$ in Figure 4, which are the scaled value of the dereverberation input segment. While, the NN output s corresponds to $\bar{S}'_d(t)$, which is the scaled value of the estimated clean log-melspectral coefficient for frame t and dimension d.

We use the implementation of the Cascade2 algorithm with RPROP (resilient propagation) weight update algorithm, which is an advanced variation of batch backpropagation algorithm [35,42], in Fast Artificial Neural Network library (FANN) [43,44]. A linear activation function is used for the output neuron, while the hidden neurons use a symmetric sigmoid (tanh) function. For defining these hidden neurons, we use four options of steepness value, i.e., 0.25, 0.50, 0.75, and 1.00. The training algorithm will choose the best steepness value for each hidden neuron. Equation 25 expresses the linear activation function and Equation 26 expresses the tanh activation function, where

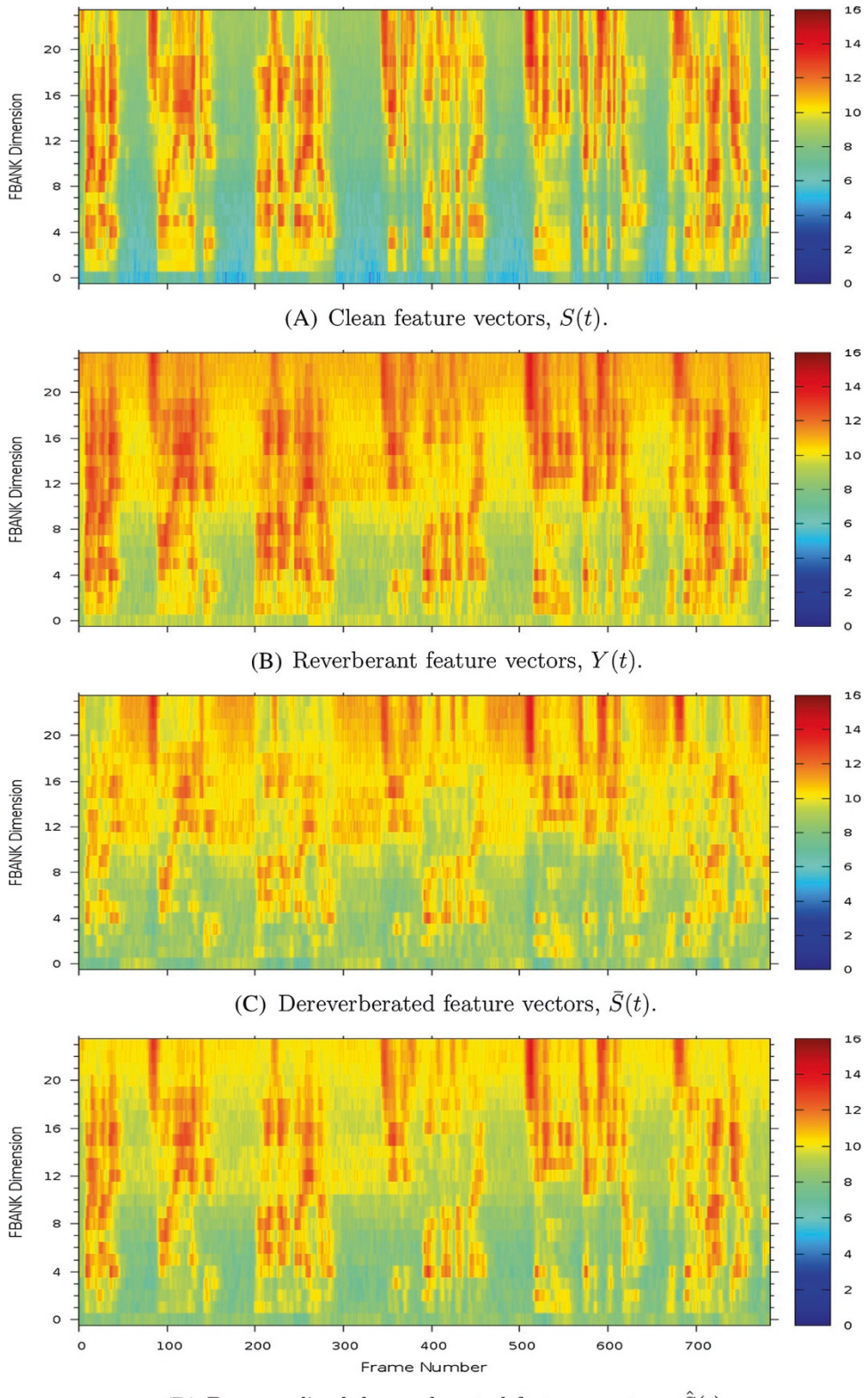

Figure 5 An example of dereverberation input and its outputs, with its clean version as the reference. (A) Clean feature vectors. **(B)** Reverberant feature vectors. **(C)** Dereverberated feature vectors. **(D)** Denormalized dereverberated feature vectors.

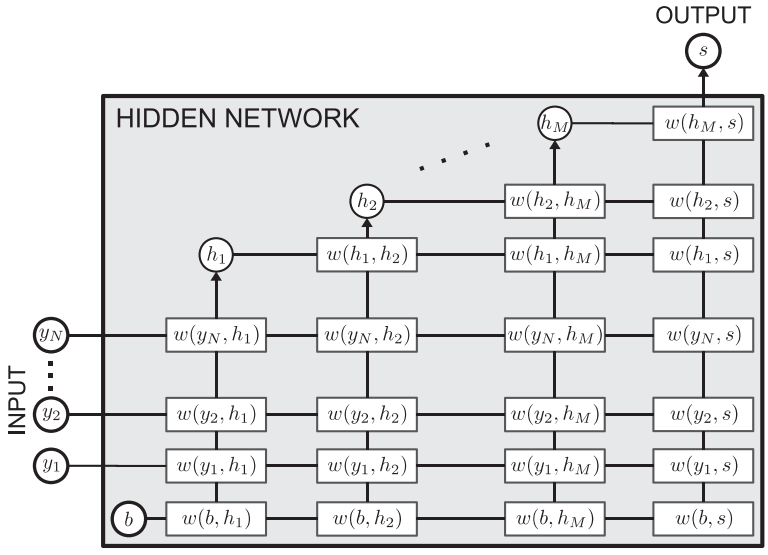

Figure 6 Illustration of the cascade neural network architecture used in this work.

x_{af} and y_{af} are the input and output of activation function, respectively, and z is the steepness.

$$y_{af} = z x_{af}, \tag{25}$$

$$y_{af} = \tanh(z x_{af}) = \frac{2}{1 + \exp(-2 z x_{af})} - 1. \tag{26}$$

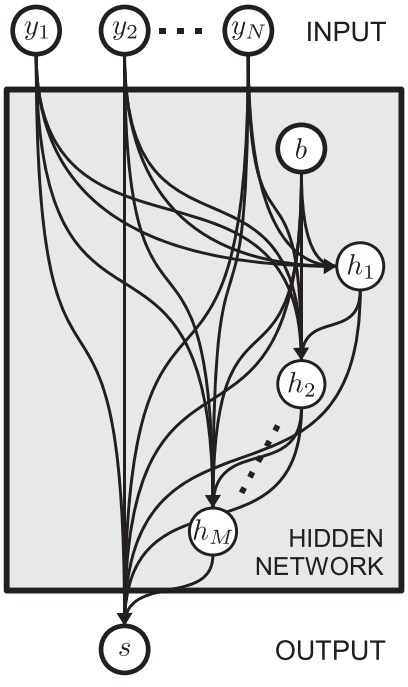

Figure 7 Illustration of the cascade neural network architecture used in this work depicted in conventional MLP representation.

Thus, the function of our output neuron s, which corresponds to $\hat{\boldsymbol{S}}$, can be expressed as

$$s = z\left(\sum_{i=1}^{N} y_i w(y_i, s) + \sum_{j=1}^{M} h_j w(h_j, s) + w(b, s)\right), \tag{27}$$

where z is the activation function steepness, y_i are the input neurons, which corresponds to $\boldsymbol{Y}$, $w(n1, n2)$ are the connection weights between neurons n_1 and n_2, b is the bias neuron, and h_j are the hidden neurons, whose function can be expressed as

$$h_j = \begin{cases} \tanh(zp) & \text{for } j = 1, \\ \tanh(zq) & \text{else}, \end{cases} \tag{28}$$

in which

$$p = \sum_{i=1}^{N} y_i w(y_i, h_j) + w(b, h_j), \tag{29}$$

$$q = \sum_{i=1}^{N} y_i w(y_i, h_j) + \sum_{k=1}^{j-1} h_k w(h_k, h_j) + w(b, h_j). \tag{30}$$

By using a linear activation function for the output neuron in the CasCor algorithm, it means that, initially, we use a linear regression to fit the transformation function. Then, it becomes a non-linear regression when the training process starts to add hidden neuron to the network.

We use several termination criteria for the training, i.e., maximum number of hidden neurons, maximum and

minimum epochs for candidates and output training, and mean squared error. The number of hidden neurons is set proportionally to the number of NN inputs (neurons). Therefore, the size of network depends on the length of input segment. For the default configuration, we set the maximum number of hidden neurons to be equal to twice of the number of NN inputs ($M \leq 2N$). The number of hidden layers will be equal to the number of hidden neurons.

By using the above termination criteria of hidden neuron number, our NN will be compact. For example, if we use nine-frame segment of input, we use in total 29 neurons, consisting of 9 input neurons, 1 bias neuron, 18 hidden neurons, and 1 output neuron. In addition, from our experiments, we could have good performance by training this compact NN using relatively few training samples. It fits the statement in [37] that the CasCor algorithm can learn fast while still create a reasonably small network that *generalizes* well.

6 Evaluation using automatic speaker identification system

6.1 Overview of the automatic speaker identification system

Figure 8 depicts the general experimental setup used in the evaluation using the SID system. The NN training and feature mapping used 24-dimensional log-melspectral feature, while the speaker model training and identification used 12-dimensional melcepstral feature (MFCC). For these experiments, we needed to create three datasets, i.e., speaker model training dataset, NN training dataset, and testing dataset.

The figure also shows the use of CMN after cepstral feature extraction. For the experiments using real dataset, it is necessary to use CMN because we need to remove the noise and reverberation in the close-talking utterances used to train the speaker models. However, it may not necessary for the experiments using simulated dataset because we have anechoic utterances to train the speaker models. Nevertheless, we used CMN in the experiments using simulated dataset. For the experiments using real dataset, we experimented on the use of speaker models trained using original MFCC and also normalized MFCC (by CMN). Meanwhile, for the experiments using simulated dataset, we experimented on the use of speaker models trained using normalized MFCC only.

The SID system used speaker-specific GMMs as the speaker models [45]. Each speaker was represented by a D-variate GMM as

$$\lambda = \{c_i, \mu_i, \Sigma_i\}, \quad \text{for } i = 1, 2, \ldots, M, \tag{31}$$

where c_i is the mixture weight, μ_i is the mean vector, Σ_i is the covariance matrix, and M is the component number. In our experiments, $M = 32$ was used. GMM parameters were estimated using the standard maximum likelihood (ML) estimation method via the expectation maximization (EM) algorithm. For a sequence of T test vectors $X = x_1, x_2, \ldots, x_T$, the GMM likelihood can be calculated using

$$L(X|\lambda) = \log p(X|\lambda) = \sum_{t=1}^{T} \log p(x_t|\lambda). \tag{32}$$

GMM is used to model the speaker identity because the Gaussian components can represent some general speaker-dependent spectral shapes and the Gaussian mixtures can model arbitrary densities [46]. In this work, we focused on developing a dereverberation approach, instead of improving the identification accuracy based on discriminative classification approach, so the use of GMM approach should be sufficient for our purpose in evaluating the proposed dereverberation method. Consideration of the state-of-the-art in speaker identification is beyond the scope of this work.

Voice activity detection (VAD) was also employed to remove the silence parts in the beginning and ending of recordings. For the simulated reverberant data, the VAD was done automatically on the melcepstral domain based on the frame log-energy coefficients. While, for the real reverberant data, the VAD was done manually by hand because the SNRs were low in utterances recorded from very distant position, e.g., from 4.0 m, and made our current automatic VAD unreliable.

6.2 Experiments using simulated reverberant data

6.2.1 Dataset description

The clean speech data was taken from the newspaper reading part of JNAS database [47]. The speech data of 100 male speakers were used. In average, each speaker has 105 utterances. The RIR and noise data were taken from Aurora-5 [48], while the simulation program was SImulation of REal Acoustics (SIREAC) [49,50]. The simulation program was used to generate the reverberant speech data from the clean speech and the RIR. The program can also add additive noise to the signal.

We created the simulated reverberant data by using the RIR of 'office' and 'livingroom' with reverberation time (T_{60}) of 400 ms. Besides, we also created a simulated noisy reverberant data by adding noise of 'office' with signal-to-noise ratio (SNR) of 20 and 10 dB to the 'office' reverberant data.

6.2.2 Experimental setup

The GMMs for the speaker identification system was trained using 500 clean utterances (100 speakers, 5 utterances for each speaker). The utterances were selected

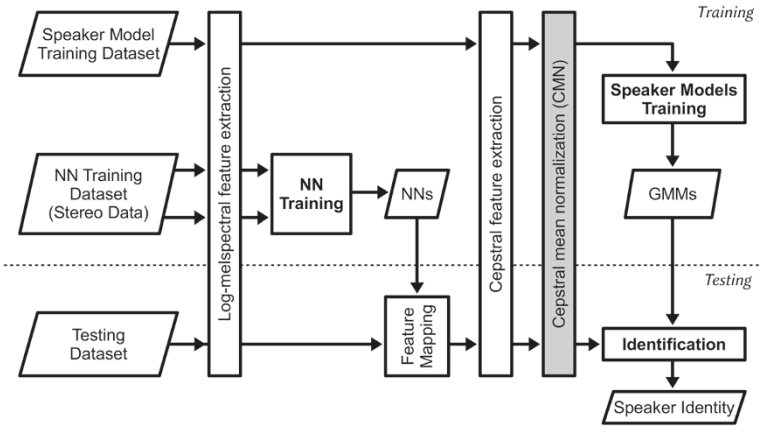

Figure 8 General experimental setup in the evaluation using SID system. The CMN was only used in some experiments.

randomly but constrained by the file size requirement so that the average duration of the utterances after VAD was about 3 s. CMN was also employed as pre-processing of GMM training data.

A pool of training data was created for each environment. This pool of data consisted of 25 pairs of clean utterance and simulated (noisy) reverberant utterance. The utterances were selected randomly but constrained by the file size requirement so that the average duration of the utterances after VAD was about 7 s. From this pool of training data, '1u' (1 pair of utterances by 1 speaker), '5u' (5 pairs of utterances by 5 speakers), '10u' (10 pairs of utterances by 10 speakers), and '15u' (15 pairs of utterances by 15 speakers) NN training datasets were created.

A testing dataset was also created for each type of simulated (noisy) reverberant data. Each dataset consisted of 1,000 simulated (noisy) reverberant utterances (100 speakers, 10 utterances for each speaker). The utterances were selected randomly but constrained by the file size requirement so that the average duration of the utterances after VAD was about 5 s. Note that the utterances in the testing dataset contained different contents (sentences) from the utterances used to train the GMMs.

The experiments were done by using causal reverberation model (using left context only) and non-causal reverberation model (using left and right context) on known environments (matched conditions). We did experiments on the use of single NN (1 NN for 24 feature dimensions) and multiple NNs (1 NN for 1 feature dimension) configurations. In addition, we did experiments using linear and skip1 frame selection.

The NN training used random weight initialization, and variations in the final NN were not unexpected. All experimental results below show the average of three experimental results, where each experiment consists of training phase and testing phase.

6.2.3 Experimental results

The baseline for each type of simulated (noisy) reverberant data is shown in Table 1. For the noisy reverberant, we only experimented on the RIR of 'office', so the baseline for the RIR of 'livingroom' is not available. The identification rate for the clean version of testing dataset was 98.0%. Note that we can regard this baseline as the result of enhancement using CMN because it was used as pre-processing of GMM training data.

We used several frame selection numbers in the experiments. In the experimental result tables, we divided the frame selection numbers into three groups, i.e., (1) left context only (L), (2) left and right context (L+R), and (3) left and shorter right context (L+sR). Group L represents the causal reverberation model, while group L+R and group L+sR represent the non-causal reverberation model. Our hypothesis in deriving the non-causal model was that the early reflections part still has considerable amount of energy. That was why the addition of short right context to the left context should be enough. The italicized text in Tables 2 and 3 represents the best performance for each training data number in a frame selection number group. The bold and italicized text represents the best performance for each training data number regardless the frame selection number group.

For a limited number of stereo data, we believed that the performance of feature mapping depends on the context

Table 1 Speaker identification baseline for each type of simulated (noisy) reverberant data

RIR		Speaker identification rate (%)			
Type	T_{60} (ms)	Clean	Reverb.	Noisy reverberant	
			SNR = ∞	SNR = 20 dB	SNR = 10 dB
Office	400	98.0	74.3	55.3	20.6
Livingroom			70.9	-	-

Table 2 Experimental results by using simulated reverberant data

NN conf.	RIR	Frame sel. type	Speaker identification rate (%)													
			Left context only (L)					Left+right context (L+R)					Left+short right context (L+sR)				
			Frame sel.	Training data				Frame sel.	Training data				Frame sel.	Training data			
				1u	5u	10u	15u		1u	5u	10u	15u		1u	5u	10u	15u
Multiple NNs	Office	Linear	3-1-0	71.6	76.7	76.8	77.0	–	–	–	–	–	–	–	–	–	–
			7-1-0	59.9	79.6	81.7	81.9	3-1-3	*70.0*	82.3	82.7	83.1	–	–	–	–	–
			15-1-0	33.4	65.3	78.8	80.9	7-1-7	55.5	76.9	83.3	85.1	7-1-3	56.2	81.3	85.4	85.8
		Skip1	3-1-0	***74.4***	79.5	79.4	80.1	–	–	–	–	–	–	–	–	–	–
			7-1-0	57.1	*81.3*	*82.4*	*84.0*	3-1-3	69.2	***83.8***	***85.8***	*86.1*	–	–	–	–	–
			–	–	–	–	–	7-1-7	52.7	72.0	82.2	85.0	7-1-3	*59.1*	*83.1*	*85.7*	***87.1***
	Livingroom	Linear	3-1-0	60.1	69.6	70.6	70.8	–	–	–	–	–	–	–	–	–	–
			7-1-0	52.3	*76.0*	78.1	78.9	3-1-3	52.3	75.4	75.8	75.7	–	–	–	–	–
			15-1-0	23.6	58.4	72.2	76.5	7-1-7	35.5	62.4	74.4	78.4	7-1-3	*32.5*	*74.1*	79.4	81.1
		Skip1	3-1-0	***63.2***	74.0	74.1	74.5	–	–	–	–	–	–	–	–	–	–
			7-1-0	39.8	75.5	*78.8*	*79.7*	3-1-3	*52.4*	***77.8***	*79.5*	79.1	–	–	–	–	–
			–	–	–	–	–	7-1-7	25.6	61.3	74.6	*79.2*	7-1-3	*32.5*	72.2	***79.6***	***82.1***
Single NN	Livingroom	Linear	3-1-0	64.9	71.6	70.6	71.3	–	–	–	–	–	–	–	–	–	–
			7-1-0	71.8	75.4	75.4	75.4	3-1-3	72.0	75.4	74.9	74.0	–	–	–	–	–
			15-1-0	70.5	*77.0*	*77.2*	*77.7*	7-1-7	73.6	77.5	79.2	78.4	7-1-3	***76.9***	77.8	78.8	78.6
		Skip1	3-1-0	71.1	73.4	74.3	74.4	–	–	–	–	–	–	–	–	–	–
			7-1-0	*72.6*	75.9	76.1	76.2	3-1-3	*73.7*	76.2	76.2	77.0	–	–	–	–	–
			–	–	–	–	–	7-1-7	71.4	*79.3*	*79.5*	***79.9***	7-1-3	74.6	***79.6***	***79.7***	*79.7*

Table 3 Experimental results by using simulated noisy reverberant data (RIR = 'office')

NN conf.	RIR	Frame sel. type	Speaker identification rate (%)													
			Left context only (L)					Left+right context (L+R)					Left+short right context (L+sR)				
			Frame sel.	Training data				Frame sel.	Training data				Frame sel.	Training data			
				1u	5u	10u	15u		1u	5u	10u	15u		1u	5u	10u	15u
Multiple NNs	20 dB	Linear	3-1-0	***53.0***	59.0	*63.5*	61.8	–	–	–	–	–	–	–	–	–	–
			7-1-0	38.9	*60.5*	62.9	*64.6*	3-1-3	42.3	***64.9***	***66.6***	65.4	–	–	–	–	–
			15-1-0	15.3	40.7	55.1	60.5	7-1-7	24.7	50.8	61.8	65.7	7-1-3	*27.8*	*58.6*	65.8	67.0
		Skip1	3-1-0	48.6	58.8	63.4	62.2	–	–	–	–	–	–	–	–	–	–
			7-1-0	32.1	*60.5*	61.8	62.9	3-1-3	*46.3*	63.1	66.0	*67.0*	–	–	–	–	–
			–	–	–	–	–	7-1-7	22.7	45.8	57.3	62.9	7-1-3	27.6	54.1	*66.2*	***67.1***
	10 dB	Linear	3-1-0	20.7	*34.8*	32.0	35.7	–	–	–	–	–	–	–	–	–	–
			7-1-0	18.3	34.1	*37.6*	*38.4*	3-1-3	*25.6*	***37.4***	38.6	41.1	–	–	–	–	–
			15-1-0	3.2	20.4	31.7	33.9	7-1-7	6.1	25.2	36.9	41.0	7-1-3	*10.1*	32.3	***40.8***	***42.8***
		Skip1	3-1-0	***31.9***	32.1	34.1	35.1	–	–	–	–	–	–	–	–	–	–
			7-1-0	13.2	32.0	36.5	37.0	3-1-3	20.7	37.3	*39.8*	*41.3*	–	–	–	–	–
			–	–	–	–	–	7-1-7	6.2	19.8	31.4	37.2	7-1-3	8.1	*32.5*	37.5	41.2

length represented by the frame selection. Based on the context length it represents, the frame selection numbers used in these experiments could be divided into:

- Short context (<100 ms): linear 3-1-0 (55 ms of context), linear 7-1-0 (95 ms), linear 3-1-3 (85 ms), and skip1 3-1-0 (85 ms).
- Medium context (100 to 200 ms): linear 15-1-0 (175 ms), linear 7-1-7 (165 ms), linear 7-1-3 (125 ms), skip1 7-1-0 (165 ms), and skip1 3-1-3 (145 ms).
- Long context (>200 ms): skip1 7-1-7 (305 ms), skip1 7-1-3 (225 ms).

In general, the feature mapping using short context could perform well. It could be improved by using medium or long context as long as the NN training data is sufficient because the use of longer context will make the NN vulnerable to high variance (overfitting) problem, where the use of more NN training data might help.

We can observe that the skip1 frame selection could almost always outperform the linear frame selection for the simulated noiseless reverberant data (Table 2). While for the simulated noisy reverberant data (Table 3), the linear frame selection was better, although the skip1 frame selection could be better for some cases.

We can also observe that we could get better performance by adding the right context. The use of left context only (group L) was the best for the small training dataset ('1u'), but for the other training datasets, the use of right context (group L+R and L+sR) could perform better. When multiple NNs configuration is used, the group L+sR could always be the best for the '15u' dataset and almost always be the best for the '10u' dataset.

In Table 2, we can observe that the single NN (1 NN) configuration was better than the multiple NNs (24 NNs) configuration for the '1u' and '5u' training datasets. For the bigger datasets, the best performance of single NN configuration could not surpass the best performance of multiple NNs configuration. In fact, the addition of training data number did not affect the performance of single NN configuration and we can observe it on each frame selection number. By using single NN configuration, it means that the NN training dataset contains the data of 24 dimensions of log-melspectral features and the NN training generalized the RTF of these 24 feature dimensions. For small training dataset, it is beneficial because we could have sufficient training dataset for training an NN. At least, it could prevent us from overfitting problem. However, if we have much more training data, the RTF generalization over 24 dimensions is not good enough because the RTF should be frequency-dependent. Consequently, the performance of single NN can not surpass the performance of multiple NNs when many training data were used. Therefore, the multiple NNs configuration should be the best choice, especially if the NN training data is sufficient. For the single NN configuration, the combination of the use of left and right context with skip1 frame selection was almost always give the best performance for each NN training data size.

6.3 Experiments using real reverberant data

6.3.1 Dataset description

The real reverberant data used in these experiments was recorded in a recording room whose dimensions were about $5 \times 6.4 \times 2.65$ m, as depicted in Figure 9 [51,52]. There were eight T-shape microphone arrays installed in the room, and each array was consisted of four microphones. Half of the arrays were installed on the ceiling, and the other half were on the wall. There was no material that was intentionally installed to reduce reverberation or noise, except the materials for microphone array that was used to place the microphones. The reverberation time was approximately 330 ms and the background noise was approximately 35 dBA, measured from the middle of the room (denoted by 'RIR' in Figure 9). The recording process was done using 32-channel recording system with 16 kHz sampling rate.

We created two real reverberant datasets in the recording room as described above. Both datasets consisted of close-talking utterances, which were recorded from the distance of 25 cm, and distant-talking utterances, which uttered from five different positions (P01 to P05) and captured by eight microphone arrays. However, for the experiments which are presented here, we only considered the first microphone of microphone array A (hereafter, simply referred as microphone A). Figure 9 shows the speaker positions, the microphone A position, and also rough distance between the speaker and the microphone.

The distances from P01, P02, P03, P04, and P05 to microphone A were about 4.0, 2.5, 2.2, 2.2, and 1.5 m, respectively. Theoretically, the direct-to-reverberant ratios (DRRs) of these distant-talking recordings were small because the distances were greater than the critical distance for the room, which is around 0.9 m (calculated using approximation of Sabine's formula [53]).

The first dataset was a non-stereo dataset. It contained two sessions of 20 speakers' recordings where each speaker uttered 10 utterances from each position in each session. In this dataset, there were also close-talking utterances (25 cm), but they were not simultaneously recorded with the distant-talking utterances. Besides, there were also utterances recorded from the distance of 50 cm, but we did not use this data. Thus, for these experiments, there were 2,000 distant-talking utterances (2 sessions $\times$ 20 speakers $\times$ 10 utterances $\times$ 5 positions) and 400 close-talking utterances (2 sessions $\times$ 20

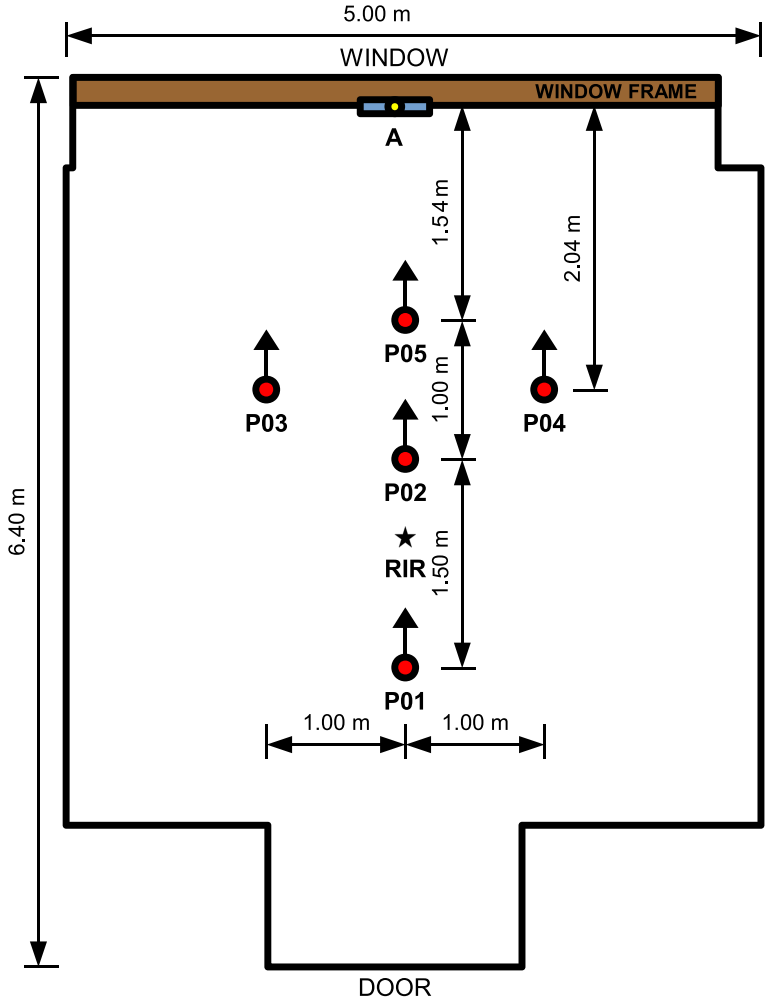

Figure 9 Illustration of the recording room used in acquiring the real reverberant data. 'P01' until 'P05' denote the speaker positions, while 'A' denotes the microphone position. The arrows represent the face direction of the speakers.

speakers × 10 utterances). This dataset was also used in [26].

The second dataset was a stereo dataset, where the close-talking and distant-talking utterances were recorded simultaneously. It contains one session of three speakers' recordings where each speaker uttered 10 utterances from each position. Thus, for these experiments, there were 150 pairs of distant-talking and close-talking utterances (1 session × 3 speakers × 10 utterances × 5 positions).

6.3.2 Experimental setup

The GMMs for the SID system were trained using 10 close-talking utterances from the non-stereo dataset for each speaker. We used two schemes for training the GMMs. Consequently, there were also two schemes in creating the testing dataset which consisted of 200 distant-talking utterances (20 speakers × 10 utterances) for each position.

- First scheme: the training data for GMMs consisted of the first five utterances (utterance A0-A4) from two sessions, while the testing data consisted of the second five utterances (utterance A5-A9) from two sessions.
- Second scheme: the training data for GMMs consisted of 10 utterances (utterance A0-A9) from the first session, while the testing data consisted of 10 utterances (utterance A0-A9) from the second session.

By using the first scheme, we could show that the SID system was text-independent because the sentences of the training dataset and testing dataset were completely different. However, the second scheme could represent a real application better, where the data of an earlier session was used as the training dataset and the later session was used as the testing dataset, although the contents of training and testing utterances are the same.

The NN training datasets were created using utterances from the stereo dataset. Although they were recorded from the distance of 25 cm, we regarded the close-talking utterances as our clean speech signals and tried to map the distant-talking utterances to close-talking utterances using NNs. The NN training datasets were created for each position and consisted of one or five pairs of utterances (utterance A0-A4) from each speaker. We defined three kinds of dataset, i.e., '1s.1u' (1 pair of utterances, 1 speaker), '1s.5u' (5 pairs, 1 speaker), and '3s.15u' (15 pairs, 3 speakers). The average duration of the training utterances after VAD was about 6.3 s. Thus, the '1s.1u', '1s.5u', and '3s.15u' datasets correspond to about 6.3, 32, and 94 s of speech signal, respectively.

We did preliminary experiments by using small NN training dataset, including '1s.1u' dataset, which contained only one pair of utterances. By using this constraint, we experimented on modified multiple NN by modifying our assumptions on the transformation of each feature dimension. In these experiments, we only considered position P01, P02, and P05.

In the next experiments, we divided the data into known conditions/positions (P01, P03, P05) and unknown conditions/positions (P02 and P04). By using the known conditions, we created position-specific and combined NN training data. Then, the NNs were used to dereverberate the known conditions and also the unknown conditions.

Besides, we tried to employ a decision-making process on which known condition was the most similar to the condition of particular input (utterance) by using environment models (GMMs), which were trained using the same data for training the position-specific NNs. It can also be seen as a way to choose the best NN sets to be used. Let $Y_t = [\, Y_{t,1} \;\; Y_{t,2} \;\; \cdots \;\; Y_{t,d}]^T$ be a d-dimensional log-melspectral feature vector for frame t. We created supervector $\hat{\boldsymbol{Y}}_{\boldsymbol{t}} = [Y_{t+N-1} \, Y_{t+N-2} \, \cdots \, Y_t]^T$ and then normalized it using segment-based normalization (as in Subsection 5.1) relative to the latest frame (Y_{t+N-1}). In the experiments, we used $d = 24$ and $N = 8$. Thus, the supervector $\hat{\boldsymbol{Y}}_{\boldsymbol{t}}$ was a 192-dimensional feature vector and the environment models (GMMs) were trained using this supervector. By using 32-mixture GMM λ for each known condition (position), the likelihood of an utterance $Y = Y_1, Y_2, \ldots, Y_T$ was determined using $\hat{\boldsymbol{Y}} = \hat{\boldsymbol{Y}}_{\boldsymbol{1}}, \hat{\boldsymbol{Y}}_{\boldsymbol{2}}, \ldots, \hat{\boldsymbol{Y}}_{\boldsymbol{T-N+1}}$ as

$$L(Y|\lambda) = \log p(\hat{\boldsymbol{Y}}|\lambda) = \sum_{t=1}^{T-N+1} \log p(\hat{\boldsymbol{Y}}_t|\lambda). \tag{33}$$

Thus, for each input utterance, we calculate the GMM likelihood of each known condition (position). Then, we selected the condition (position) having the highest likelihood from the likelihood calculation result and used its position-specific NN sets to do feature mapping on the input.

In other words, we tried to determine the most appropriate position-specific NNs to be used by evaluating segments of the input utterance using the environment models (GMMs). The GMM itself tried to model the RTF for each known reverberant environment (position). It is impossible to capture the characteristic of reverberant environment (RTF) by using a single frame, so we used the GMM to model segments (eight-frame, corresponds to 95 ms of context) of reverberant utterances. However, by using segment, the approach will be text-dependent because the segment is long enough to capture one or more phonemes, which are also affected by the adjacent phonemes beyond the segment length. Therefore, it is important to do the experiment using the first scheme of training and testing, where the spoken utterances of training and testing were different. In addition, it is possible to improve the approach by training and evaluating using segments which are ended by silence only.

As comparison, we tried to implement the mapping method on *melcepstrum (MFCC) domain* using MLP network proposed in [25]. Because we only use 12-dimensional MFCC for the identification, we only created 12 MLPs, instead of 13 MLPs. Moreover, there was no further explanation about the frame selection used in [25], so we did two implementations, i.e., by using linear frame selection 8-1-0 and 4-1-4.

All experimental results below show the average performance of several experiments. For the preliminary experiments (**Fundamental results**), the experimental results of '1s.1u' represent the average of 45 experiments (3 speakers × 5 utterances/speaker × 3 experiment sets) and the experimental results of '1s.5u' represent the average of 9 experiments (3 speakers × 3 experiment sets). For the use of '1s.5u' dataset in the other experiments (**Position-dependent/independent results**), we calculated the average of three experimental results for each training speaker, and then we average them. Thus, they also represent the average of nine experiments. While for all use of '3s.15u' dataset, we calculated the average of three experimental results. Thus, they represent the average of three experiments.

6.3.3 Experimental results

Fundamental results Table 4 shows the baseline for the first and second scheme of GMM training and testing. The tables also show the result of CMN enhancement.

The baselines for both schemes were quite low. Note that the speaker model trained using utterances recorded from the distance of 25 cm, which most likely contained noise and reverberation. The CMN was very effective to

Table 4 Baseline for the speaker identification experiments using the real reverberant data

Experiment scheme	Method	Speaker identification rate (%)							
		P01	P03	P05	P02	P04	Avg. known	Avg. unknown	Avg. all
I	Baseline	45.0	49.0	45.5	44.0	46.0	46.5	45.0	45.9
	CMN	83.0	88.0	91.5	93.0	90.0	87.5	91.5	89.1
II	Baseline	45.0	50.5	54.0	55.5	49.5	49.8	52.5	50.9
	CMN	84.0	89.5	91.0	91.5	89.5	88.2	90.5	89.1

The known environments include P01, P03, and P05, while the unknown environments include P02 and P04.

remove that distortions and improve the speaker identification rate. The CMN could reach 79.9% and 77.8% of ERR relative to the baseline for the first and second testing schemes, respectively. By using CMN, the identification rates for the close-talking version of testing dataset were 99.5% and 100% for the first and second schemes, respectively. For the evaluation using SID system, our proposed method could also perform very well when combined with CMN and was not effective when it worked alone, as shown later.

Table 5 shows our first experimental results using the real reverberant data. The experiments used the first scheme on creating the GMM training dataset and testing dataset. Besides, we only considered the utterances from positions P01, P02, and P05, because these three positions and microphone A were in-line and the speakers' utterances were directed to microphone A.

Similar to the experiment results using the simulated reverberant data, the multiple NNs configuration showed better performance for the '3s.15u' dataset but worse for the '1s.1u' and '1s.1u' datasets. Therefore, we tried to look for a good performance trade-off for all datasets by using modified multiple NNs configuration, where one NN was used for more than one dimension of feature. In our experiments, the number of dimension in each NN was divided evenly. For example, in six NNs configuration, the first NN was for the dimension 1-4 of 24-dimensional log-melspectral feature, the second NN was for 5-8, and so on. It meant that we assumed that the RTF for the dimension 1-4 was the same, the RTF for the dimension 5-8 was the same, and so on.

Table 5 shows the experimental results in terms of ERR relative to the CMN for the use of causal skip1 3-1-0 frame selection (corresponds to 85 ms of context). '1 NN' represents the single NN configuration, and '24 NNs' represents the *original* multiple NNs configuration.

We can observe that the use of modified multiple NNs configuration ('6 NNs' and '12 NNs') could be better than single NN configuration ('1 NN') and also original multiple NNs configuration ('24 NNs'). The averages of ERR (relative to the CMN) across the types of dataset for the '6 NNs' and '12 NNs' configuration were 41.7% and 41.1%. So, the '6 NNs' configuration gave us the best trade-off. Comparing to '1 NN' configuration, '6 NNs' significantly

Table 5 Experimental results on the use of modified multiple NNs configuration for known positions (matched conditions)

Method	Dataset	Error rate reduction (%)			
		P01	P02	P05	Avg.
Proposed (1 NN) + CMN	1s.1u	30.8	19.4	21.7	24.0
	1s.5u	34.7	27.8	23.5	28.7
	3s.15u	35.4	33.3	23.5	30.8
Proposed (6 NNs) + CMN	1s.1u	40.8	44.0	19.7	**34.8**
	1s.5u	45.5	62.7	16.3	41.5
	3s.15u	51.0	71.4	23.5	48.7
Proposed (12 NNs) + CMN	1s.1u	33.0	33.7	22.9	29.8
	1s.5u	40.6	54.0	32.0	**42.2**
	3s.15u	55.2	71.4	27.5	**51.4**
Proposed (24 NNs) + CMN	1s.1u	17.9	2.5	10.5	10.3
	1s.5u	30.2	31.0	22.2	27.8
	3s.15u	46.9	42.9	13.7	34.5

The experiments were done by using the first testing scheme and skip1 3-1-0 frame selection. The results are shown in term of ERR relative to the 'CMN'. The bold text represents the best average performance for each training data number.

improved the performance of '1s.5u' and '1s.1u' datasets. Surprisingly, comparing to '24 NNs' configuration, it also improved the performance of '3s.15u' dataset. The averages of ERR (relative to the CMN) for '1s.1u', '1s.5u', and '3s.15u' were 34.8%, 41.5%, and 48.7%, respectively. In ddition, we could get better performance for '3s.15u' dataset by using '12 NNs' configuration, where its average ERR was 51.4%.

Position-dependent/independent results For the next experiments, we considered the utterances from all positions but divided the data into known conditions/positions (P01, P03, P05) and unknown conditions/positions (P02, P04). The data of known conditions is used for training and/or testing, while the data of unknown conditions is used for testing only.

Table 6 shows the experimental results by using position-specific training data for the first testing scheme. In these experiments, the NN sets were trained using the utterances from one position and then were used to dereverberate the utterances from all positions (including the position used for training).

Table 6 shows that by using '1s.5u' datasets, the average identification rates of '24 NNs', '12 NNs', and '6 NNs' configurations for matched conditions were 88.4%, 90.3%, and 90.3%. While, by using '3s.15u' datasets, the average identification rates were 93.2%, 93.7%, and 92.2%. Besides the experiment using the first scheme, we also did it using the second scheme. For the '1s.5u' datasets, the '6 NNs' configuration performed best in the first and second schemes. Meanwhile, for the '3s.15u' datasets, the '12 NNs' configuration performed best in the first scheme and the '24 NNs' performed best in the second scheme.

From Table 6, we can also observe that the use of utterances from one position could perform reasonably well for the other (unknown) positions in the room. We can observe that by using more training data, we could improve the performance. Comparing to the identification rate of CMN (Table 4), the total averages of identification rate were often worse for '1s.5u' datasets but always better for the '3s.15u' datasets. It is similar as the experimental results for the matched conditions as discussed above. Besides, we can observe that the use of utterances from position P05 as NN training data could perform better than, or at least could perform as good as, the use of utterances from other positions. Moreover, by using a small dataset ('1s.5u') of position P05, we could get a reasonably good performance.

For Tables 7 and 8, we combined the utterances from known positions (P01, P03, P05) into a training dataset. Therefore, we created '1s.15u' dataset ('1s.5u' dataset × 3

Table 6 Experimental results by using position-specific training data

Method	Dataset		Speaker identification rate (%)							
			P01	P03	P05	P02	P04	Avg. known	Avg. unknown	Avg. all
Proposed (24 NNs) + CMN	P01	1s.5u	85.8*	86.0	85.9	87.1	85.5	85.8	86.1	86.1
		3s.15u	91.8*	91.8	92.0	93.2	90.0	91.8	91.8	91.8
	P03	1s.5u	84.0	86.8*	85.2	88.1	88.6	86.8	86.5	86.5
		3s.15u	90.3	92.7*	93.5	94.8	92.8	92.7	92.9	92.8
	P05	1s.5u	88.8	91.2	92.7*	93.3	89.7	92.7	90.8	91.1
		3s.15u	90.3	93.0	95.0*	96.2	91.7	95.0	92.8	93.2
Proposed (12 NNs) + CMN	P01	1s.5u	88.8*	88.6	88.6	89.9	90.4	88.8	89.4	89.3
		3s.15u	93.5*	94.3	94.2	95.0	93.3	93.5	94.2	94.1
	P03	1s.5u	87.8	89.6*	89.3	90.7	92.3	89.6	90.0	89.9
		3s.15u	91.5	94.7*	94.3	95.0	94.2	94.7	93.8	**93.9**
	P05	1s.5u	89.9	92.5	92.4*	94.2	92.7	92.4	92.3	**92.3**
		3s.15u	91.5	93.2	92.8*	96.5	92.7	92.8	93.5	93.3
Proposed (6 NNs) + CMN	P01	1s.5u	89.5*	88.4	87.9	90.9	91.7	89.5	89.7	89.7
		3s.15u	92.2*	91.7	91.0	94.7	93.3	92.2	92.7	92.6
	P03	1s.5u	89.1	89.7*	88.4	91.0	92.7	89.7	90.3	90.2
		3s.15u	91.5	92.0*	92.0	95.0	93.8	92.0	93.0	92.9
	P05	1s.5u	90.1	91.4	91.7*	94.7	93.4	91.7	92.4	**92.3**
		3s.15u	92.5	93.2	92.3*	96.3	94.2	92.3	94.0	93.7

The known environments include P01, P03, and P05, while the unknown environments include P02 and P04. The experiments were done by using the first testing scheme and skip1 7-1-0 frame selection. The asterisks (*) indicate known positions (matched conditions). The bold text represents the best average performance for each training data number.

Table 7 Experimental results by using combined training data (three known positions) on the first testing scheme

Method	Dataset		Speaker identification rate (%)							
			P01	P03	P05	P02	P04	Avg. known	Avg. unknown	Avg. all
Prop. (24 NNs) + CMN	combd	1s.15u	88.9	90.4	93.9	93.8	90.2	91.1	92.0	91.4
		3s.45u	90.5	93.7	94.5	96.2	92.0	92.9	94.1	93.4
Prop. (12 NNs) + CMN	combd	1s.15u	90.9	92.5	93.0	94.2	92.2	92.1	93.2	**92.6**
		3s.45u	93.5	94.8	94.2	96.2	93.2	94.2	94.7	**94.4**
Prop. (6 NNs) + CMN	combd	1s.15u	90.8	91.2	91.6	94.4	93.8	91.2	94.1	92.4
		3s.45u	92.7	92.2	92.5	96.0	94.0	92.4	95.0	93.5

The known environments include P01, P03, and P05, while the unknown environments include P02 and P04. The experiments were done by using skip1 7-1-0 frame selection. The bold text represents the best average performance for each training data number.

positions) and '3s.45u' dataset ('3s.15u' dataset × 3 positions). From Table 6, the use of 15 pairs of utterances as NN training data ('15u') seems enough for our method to perform well. So, one possibility that may make a difference of performance between '1s.15u' and '3s.45u' datasets is speaker variation in the NN training data. However, these two types of combined datasets could perform well, and the performance difference between both datasets was not significant. For the first scheme, the total average identification rates were 91.4% to 94.4%, which is 21.1% to 48.6% of ERR relative to the CMN. Meanwhile, for the second scheme, the total average identification rates were 92.6% to 94.8%, which corresponds to 32.1% to 52.3% of ERR relative to the CMN. Thus, eventually, the similarity of utterances' content between training and testing data was affecting the performance. In addition, similar as experimental results shown in Table 6, the '12 NNs' and '24 NNs' configurations performed best in the first and second schemes, respectively.

The identification rates for unknown conditions were also improved. It shows that by combining the training data from known positions in a room, we could train NN sets which generalize well, so that the NN sets could also perform well for unknown positions in the same room.

Table 9 shows the performance of GMM as position-specific NNs selector. We can compare it to the performance of combining training data approach (Table 7). For '6 NNs' configuration, both approaches showed similar performance. However, for '12 NNs' and '24 NNs' configuration, the combining training data approach could perform better. Besides, we could also compare it to the results of matched condition cases in Table 6.

Table 9 shows that by using '1s.5u' datasets, the average identification rates of '24 NNs', '12 NNs', and '6 NNs' configurations for known conditions were 90.5%, 91.4%, and 91.1%. Meanwhile, by using '3s.15u' datasets, the average identification rates were 92.4%, 93.0%, and 92.2%. Thus, we could see that the use of environment models (GMMs) as NNs selector could actually perform well because it could perform as good as the matched condition case, especially for the small datasets ('1s.5u'). However, the combining training data approach (Table 7) was slightly better than the use of environment models (GMMs).

Comparison to the feature mapping on MFCC domain

In addition, we implemented the mapping method on melcepstrum (MFCC) domain by MLP network as proposed in [25] and experimented by using position-specific training data. For the first testing scheme, the average identification rates of 'linear 8-1-0' and 'linear 4-1-4' frame selections for matched conditions were 69.7% and 76.2% by using '1s.5u' datasets and 78.5% and 82.1% by using '3s.15u' datasets. Meanwhile, for the second scheme, they were 74.1% and 78.9% by using '1s.5u' datasets and 79.2% and 83.0% by using '3s.15u' datasets. Thus, our proposed

Table 8 Experimental results by using combined training data (three known positions) on the second testing scheme

Method	Dataset		Speaker identification rate (%)							
			P01	P03	P05	P02	P04	Avg. known	Avg. unknown	Avg. all
Prop. (24 NNs) + CMN	combd	1s.15u	89.7	92.4	94.6	96.9	91.0	92.2	94.0	**92.9**
		3s.45u	91.5	92.8	95.8	99.0	92.7	93.4	95.8	**94.4**
Prop. (12 NNs) + CMN	combd	1s.15u	89.5	92.4	93.6	96.1	91.6	91.8	93.9	92.6
		3s.45u	90.5	92.2	94.5	98.0	92.5	92.4	95.3	93.5
Prop. (6 NNs) + CMN	combd	1s.15u	90.5	90.8	93.7	94.8	93.5	91.7	94.2	92.7
		3s.45u	91.5	92.0	95.0	96.0	93.0	92.8	94.5	93.5

The known environments include P01, P03, and P05, while the unknown environments include P02 and P04. The experiments were done by using skip1 8-1-0 frame selection. The bold text represents the best average performance for each training data number.

Table 9 Experimental results by using environment models (GMMs) as NNs selector

Method	Dataset		Speaker identification rate (%)							
			P01	P03	P05	P02	P04	Avg. known	Avg. unknown	Avg. all
GMM32 + Prop. (24 NNs) + CMN	P01/3/5	1s.5u	88.6	90.2	92.8	93.4	90.0	90.5	91.7	91.0
		3s.15u	89.5	93.3	94.3	95.3	91.0	92.4	93.2	92.7
GMM32 + Prop. (12 NNs) + CMN	P01/3/5	1s.5u	89.4	92.3	92.4	93.5	92.5	91.4	93.0	92.0
		3s.15u	91.7	93.5	93.8	96.5	93.5	93.0	95.0	**93.8**
GMM32 + Prop. (6 NNs) + CMN	P01/3/5	1s.5u	90.4	90.9	91.9	94.3	93.7	91.1	94.0	**92.2**
		3s.15u	92.0	92.7	92.0	96.7	94.2	92.2	95.4	93.5

The known environments include P01, P03, and P05, while the unknown environments include P02 and P04. The experiments were done by using the first testing scheme and skip1 7-1-0 frame selection. The bold text represents the best average performance for each training data number.

method is much better. In fact, the mapping method by MLP network could not be better than the use of CMN alone. Nevertheless, these experiments also show that the use of right context is beneficial.

7 Evaluation using automatic speech recognition system

7.1 Overview of CENSREC-4

CENSREC-4 is an evaluation framework for distant-talking speech recognition in reverberant environments [22]. The task in CENSREC-4 is grammar-based connected digit recognition. The vocabulary consists of eleven Japanese numbers: 'ichi', 'ni', 'san', 'yon', 'go', 'roku', 'nana', 'hachi', 'kyu', 'zero', and 'maru'. All recordings are sampled at 16 kHz.

CENSREC-4 data is divided into 'basic dataset' and 'extra dataset'. The basic dataset contains simulated reverberant data made by convolving eight kinds of RIRs with the clean speech, as shown in Table 10 [22]. In this dataset, there are two sets of testing data and two sets of training data. The testing data is divided into test set A (office, elevator hall, in car, living room) and test set B (lounge, Japanese-style room, meeting room, Japanese-style bath). In total, each testing dataset consists of 4,004 utterances by 104 speakers (52 females and 52 males). Thus, there are 1,001 utterances for each reverberant environment.

The training data is divided into clean and multi-condition datasets. The clean dataset consists of 8,440 utterances by 110 speakers (55 females and 55 males). The multi-condition dataset consists of simulated reverberant data generated by convolving four kinds of RIRs from test set A with the utterances from clean dataset. Thus, there are 2,110 utterances for each reverberant environment. Because of the availability of multi-condition training data, test set A can be regarded as *known* reverberant environments and test set B can be regarded as *unknown* environments.

The extra dataset contains two sets of testing data. The testing data is divided into test set C, which contains simulated reverberant data with multiplicative and additive noise, and test set D, which contains real reverberant data. The utterances of test set D were recorded in a car, lounge, meeting room, and office (Table 10) by 10 speakers (five females and five males) using close- and distant-talking microphones. The close-talking microphone was using headset, while the distant-talking microphone was 50 cm away from the speaker. For each environment, the data is divided into testing dataset (493 utterances) and adaptation dataset (110 utterances). Thus, the testing dataset of set D contains 1,972 utterances by 10 speakers.

Test sets A and B were used to evaluate the proposed method on simulated reverberant environments, while

Table 10 Environment conditions for CENSREC-4 data acquisition

Room	Test set	Room size	Mic. distance	Reverb. time (T_{60})	Ambient noise (dBA)
Office	A/D	9.0 × 6.0 m	0.5 m	0.25 s	36.5 dB
Elevator hall	A	11.5 × 6.5 m	2.0 m	0.75 s	39.0 dB
In car	A/D	Middle-sized sedan	0.4 m	0.05 s	32.0 dB
Living room	A	7.0 × 3.0 m	0.5 m	0.65 s	34.0 dB
Lounge	B/D	11.5 × 27.0 m	0.5 m	0.50 s	52.5 dB
Japanese-style room	B	3.5 × 2.5 m	2.0 m	0.40 s	30.0 dB
Meeting room	B/D	7.0 × 8.5 m	0.5 m	0.65 s	48.5 dB
Japanese-style bath	B	1.5 × 1.0 m	0.3 m	0.60 s	29.5 dB

test set D was used to evaluate the proposed method on real reverberant environments. For both evaluations, the acoustic models were trained using clean and/or multi-condition training datasets. The acoustic models consist of 18 phoneme models (5 states; 20-mixture GMM), silence 'sil' (5 states; 36-mixture GMM), and short pause 'sp' (3 states; 36-mixture GMM).

7.2 Experiments using simulated reverberant data

7.2.1 Experimental setup

The experimental setup for the simulated reverberant data is depicted by Figure 10. The NN training and the feature mapping were done using 24-dimensional log-melspectral feature vectors, while the acoustic model (AM) training and the recognition were done using 39-dimensional melcepstral feature vectors consisting of 12-dimensional MFCC parameters and the log energy, with their delta and delta-delta parameters. The benefit of CMN use *after* our proposed method (feature mapping) was also investigated.

Some pairs of utterances from set A were used as the NN training data. We created multi-condition five-speaker datasets ('5s') as described in Table 11. We selected 5 speakers (2 females and 3 males) from the 110 available speakers, then randomly selected some pairs of utterances from each environment (4 environments) for each selected speaker (5 speakers) and combined them to make the dataset.

We also created cross validation dataset consisting of randomly selected 400 utterances from simulated reverberant data by 104 speakers. The NN training used random weight initialization and variations in the final NN were not unexpected. Therefore, in our experiments, we did the NN training five times which resulted five NNs for each dimension of feature. Then, we selected the NNs which yielded the lowest MSE for the cross validation dataset. Finally, the best set of NNs was used in the feature mapping.

In all experimental results below, the original AMs (refer to [22]) trained using only clean data and only multi-condition data are denoted by 'clean' and 'multi', respectively. The retrained AM is denoted by '-rt'. For example, 'multi-rt' denotes the AM retrained using dereverberated multi-condition data, and 'cln+mlt-rt' denotes the AM retrained using clean and processed multi-condition data. The experiments were done by using causal reverberation model and skip1 frame selection only.

7.2.2 Experimental results

Table 12 shows the upper bound and the baseline for set A and set B. In the table, 'close' denotes the clean utterances and represents the upper bound. The table shows that the use of multi-condition AM and CMN were effective. By using that combination, we could reach 44.9% of average ERR relative to the baseline. The baseline (by using clean and multi-condition AMs) is the same as shown in [22,54].

The experimental results on the simulated reverberant data of CENSREC-4 by using Hybrid Delta [55] reached 95.7% and 94.7% of average digit accuracy for sets A and B. Thus, it reached 95.2% of total average digit accuracy, or 71.3% of average ERR relative to the baseline. Meanwhile, the DAE-based approach proposed in [29,30] reached 98.4% and 97.0% of average digit accuracy for sets A and B. Thus, it reached 97.7% of total average digit accuracy, or 86.2% of average ERR relative to the baseline, which was better than the ideal case (upper bound) for multi-condition AM (96.2% of digit accuracy). Both works used retrained multi-condition AM.

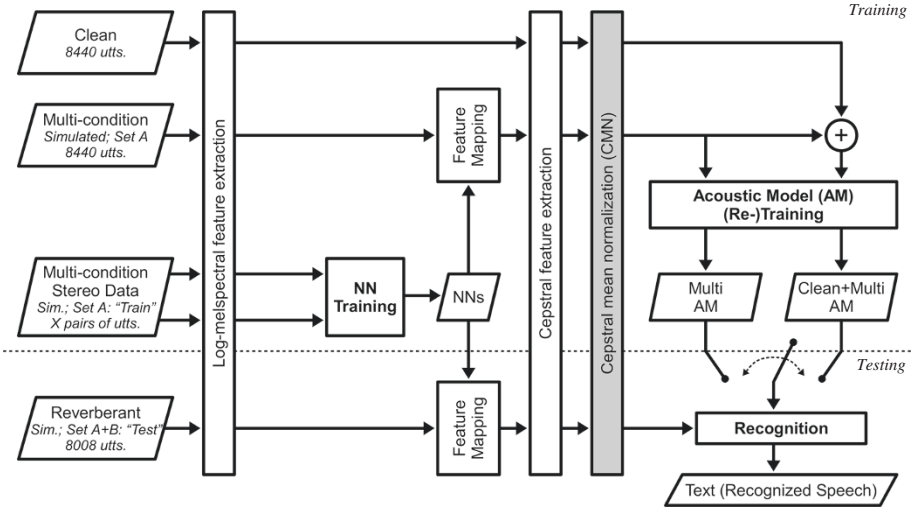

Figure 10 Setup for the experiments using simulated reverberant data. The CMN was only used in some experiments.

Table 11 Multi-condition (combined) NN training datasets for the experiments using simulated reverberant data

Dataset name	Speaker number	Utterances num. per spk. per env. (pairs of utts.)	Total duration of utterances (seconds)
5s.20u	5	1	61
5s.40u	5	2	110

The total duration of utterances is after removing the silence parts in the beginning and ending of each recording.

Table 13 shows the experimental results of our proposed approach on set A and set B using '5s.20u'. The table shows the performance of skip1 frame selection with segment length of nine frames ('8-1-0'). We can observe that the use of CMN after our proposed method is not effective. We can also observe that the use of 'multi-rt' AM was effective and the use of 'cln+mlt-rt' AM, which used more training data, could not improve the performance further. The best average performance of our proposed method shown in Table 13 is 96.1%, or 76.6% of ERR relative to the baseline, which was almost the same as the ideal case (upper bound) for multi-condition AM. It is better than the performance of Hybrid Delta approach, but worse than the DAE approaches.

We tried to double the number of NN training data ('5s.40u') and also the context length (17 frames, '16-1-0'). By using more training data, the performance could be improved, but the DAE approaches were still better. The best average performance of our proposed method shown is 96.4%, which is 78.4% of ERR relative to the baseline. Comparing to our approach, besides using different NN architecture, DAE used a bigger neural network, which was trained using more advanced training algorithm and also much more NN training data (2,110 pairs of utterances).

7.3 Experiments using real reverberant data

7.3.1 Experimental setup

The experimental setup for the real reverberant data is depicted by Figure 11. The features used in these experiments were the same as in the experiments using simulated reverberant data (Subsubection 7.2.1).

Some pairs of utterances from the adaptation set of set D were used as the NN training data. We created *multi-condition (combined)* five-speaker datasets ('5s') and one-speaker datasets ('1s') as described in Table 14. We selected 5 speakers (2 females and 3 males) from the 10 available speakers. For the five-speaker datasets, we randomly selected some pairs of utterances from each environment (four environments) *for each selected speaker (five speakers)* and then combined them to make the dataset. For the one-speaker datasets, we randomly selected some pairs of utterances from each environment (four environments) and then combined them to make the dataset *for each selected speaker (five speakers in total)*. The total duration of utterances for each speaker varies from 37 to 84 s (20-utterance datasets) and 75 to 166 s (40-utterance datasets). We also created a multi-condition (combined) cross validation dataset consists of randomly selected 200 utterances (50 utterances for each environment) of 10 speakers.

In addition, slightly different from the experimental setup shown in Figure 11, we also used *environment-specific* five-speaker ('5s') and one-speaker ('1s') datasets as described in Table 15 for training the NNs and environment-specific cross validation datasets for selecting the best set of NNs. In fact, the environment-specific datasets were the datasets which compose the multi-condition (combined) datasets, for example the multi-condition (combined) '5s.20u' datasets were composed by environment-specific '5s.5u' datasets of four environments. The environment-specific cross validation datasets were also the datasets which compose the multi-condition (combined) cross validation dataset.

Table 12 Upper bound and baseline for the experiments using the simulated reverberant data

Method	Acoustic model	Digit accuracy (%)										
		Set A				Set B				Avg. set A	Avg. set B	Avg.
		Office	EvHall	InCar	LivingR	Lounge	JPstyR	ConfR	BathR			
Close (Ideal)	Clean	99.5	99.4	99.5	99.3	99.5	99.4	99.5	99.3	99.4	99.4	99.4
	Multi	96.2	96.5	96.1	96.2	96.2	96.5	96.1	96.2	96.2	96.2	96.2
Baseline	Clean	97.5	57.9	95.6	84.4	74.0	89.5	89.8	78.0	83.8	82.8	83.3
	Multi	94.3	90.5	94.7	91.5	79.6	93.5	93.4	84.2	92.8	87.7	90.2
CMN	Clean	97.8	66.0	98.7	83.5	87.3	92.2	93.3	81.7	86.5	88.6	87.6
	Multi	92.8	91.9	92.5	90.0	92.6	91.9	93.1	81.3	91.8	89.7	90.8

Table 13 Experimental results on the simulated reverberant data by using NN training dataset '5s.20u'

Method	Frame selection	Acoustic model	Digit accuracy (%)										
			Set A				Set B				Avg. set A	Avg. set B	Avg.
			Office	EvHall	InCar	LivingR	Lounge	JPstyR	ConfR	BathR			
Proposed (24 NNs)	skip1 8-1-0	Clean	98.6	66.3	98.8	89.8	90.1	95.2	92.7	91.5	88.4	92.4	90.4
		Multi-rt	97.4	95.6	97.8	97.4	92.1	97.3	96.1	94.5	97.1	95.0	96.1
		Cln+mlt-rt	97.6	94.2	98.0	97.0	93.5	97.7	96.5	94.2	96.7	95.5	96.1
Proposed (24 NNs) + CMN	skip1 8-1-0	Clean	97.7	67.8	98.4	86.0	93.3	93.4	92.4	86.6	87.5	91.4	89.5
		Multi-rt	96.6	94.7	96.8	96.2	94.7	96.5	95.9	93.6	96.1	95.2	95.7
		Cln+mlt-rt	97.1	94.7	97.1	95.2	94.1	96.2	96.2	92.5	96.0	94.7	95.4

We used clean and multi-condition datasets to train the new acoustic models. The multi-condition dataset was dereverberated first using the feature mapping method. Because of the multi-condition dataset is noiseless while NN training data is noisy, we implemented log-melspectral channel mean normalization (hereafter, referred as LMCN) before the NN training and the feature mapping to deal with the mismatch. The LMCN, which works the same as CMN, can be expressed as

$$c'_{t,d} = c_{t,d} - \frac{1}{T}\sum_{t=1}^{T} c_{t,d}, \quad \text{for } d = 1, 2, \ldots, D, \tag{34}$$

where $c'_{t,d}$ and $c_{t,d}$ are the normalized and original coefficients of frame index t and feature dimension d, T is the number of frames, and D is the number of feature dimension.

In all experimental results below, the original AMs (refer to [22]) trained using only clean data and only multi-condition data are denoted by 'clean' and 'multi', respectively. The retrained AM is denoted by '-rt'. Besides 'multi-rt' AM, we have 'clean+multi-rt' AM, which is the same as 'cln+mlt-rt' AM in the experiments using the simulated reverberant dataset, and we also create 'clean-rt' AM specifically for the experiments using the real reverberant dataset. The difference between 'clean' and 'clean-rt' AMs was in the cepstal feature (MFCCs) extraction. The cepstral feature for 'clean' was extracted from the log-melspectral feature (as in standard MFCC extraction), while the MFCCs for 'clean-rt' were extracted from the *normalized* log-melspectral feature. The experiments were done by using causal reverberation model only. Besides, we mainly used skip1 frame selection.

7.4 Experimental results

7.4.1 Fundamental results

Table 16 shows the upper bound and the baseline for set D. In the table, 'close' denotes close-talking utterances and represents the upper bound. The table shows that the use of multi-condition AM and CMN were effective. By using that combination, we could reach 40.9% of ERR relative to

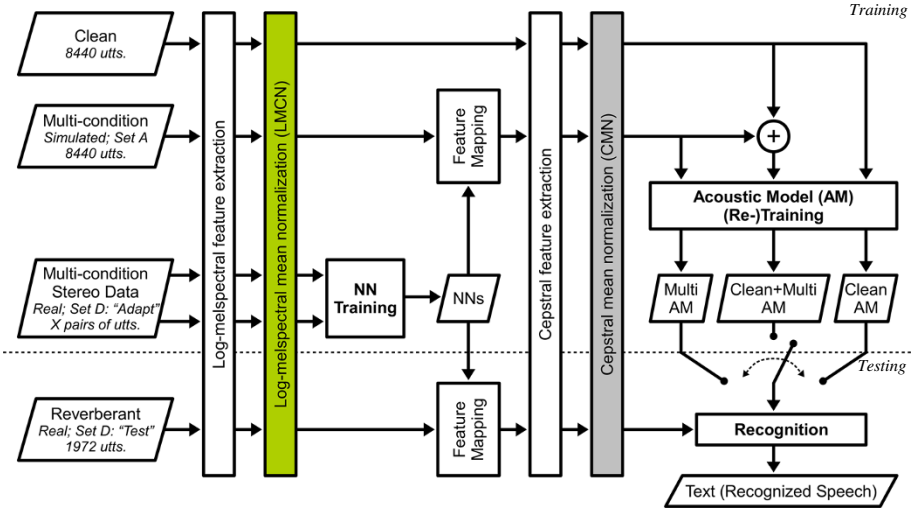

Figure 11 Setup for experiments using real reverberant data. The CMN was only used in some experiments.

Table 14 *Multi-condition (combined)* NN training datasets for the experiments using real reverberant data

Dataset name	Speaker number	Utterances num. per spk. per env. (pairs of utts.)	Total duration of utterances (seconds)
5s.20u	5	1	54
5s.40u	5	2	108
5s.60u	5	3	162
5s.80u	5	4	213
1s.20u	1	5	70
1s.40u	1	10	138

The total duration of utterances is after removing the silence parts in the beginning and ending of each recording. For the one-speaker datasets ('1s'), the total duration is the average from five speakers' datasets.

the baseline. The baseline (by using clean AM) is the same as shown in [10,54].

As a comparison, we show the experimental results from [10] in Table 17. It shows the performance of generalized spectral subtraction (GSS) and joint sparse representation (JSR) on the real reverberant dataset of CENSREC-4. In the GSS approach, the AM was retrained using processed (dereverberated) multi-condition reverberant data. Meanwhile, the JSR approach worked on clean AM, assumed that the environment is known, and used 110 pairs of utterances in training the sparse representation for each environment. The best performances for both approaches were when the CMN was employed. The performance of 'GSS+CMN' and 'JSR+CMN' reached 56.3% and 53.4%, respectively, of ERR relative to the baseline.

Besides, we also compare our experimental results to the performance of Hybrid Delta, which is a new scheme for calculating the delta and delta-delta coefficients of static MFCC coefficients by using linear-logarithmic hybrid domain [55]. A multi-condition AM was retrained using the proposed features. The performance of Hybrid Delta reached 31.4% of ERR relative to the baseline.

Table 18 shows the experimental results of our proposed approach on set D using '5s.20u' dataset. The table shows the performance of linear and skip1 frame selection with segment length of 9 frames ('8-1-0'), 17 frames ('16-1-0'), and 33 frames ('32-1-0'). Note that the segment of skip1 8-1-0 represents the same context length to linear 16-1-0, which is 185 ms, and skip1 16-1-0 represents the same context length to linear 32-1-0, which is 345 ms.

We can observe the performance difference caused by the difference of AM used in the recognition. The 'clean+multi-rt' AM could perform much better than the 'multi-rt' AM. We can observe that the frame selection used in the feature mapping was affecting the performance. The differences of performance between the use of linear 16-1-0 and the use of skip1 8-1-0 were not significant, except when we used 'multi-rt' AM. Meanwhile, the use of skip1 16-1-0 was better than the use of linear 32-1-0. Because of the long segment it used, the use of linear 32-1-0 most likely had high variance (overfitting) problem. That was also why the use of linear 32-1-0 was worse than the use of linear 16-1-0. We can observe that the use of longer segment for linear frame selection was not effective. However, we can see the benefit of using longer segment for skip1 frame selection. The use of 17-frame segment (skip 16-1-0) could perform better than the use of nine-frame segment (skip1 8-1-0). As an additional information, the CMN is not effective when it was used for the retrained AMs, which may be caused by the fact that the LMCN works the same as CMN. The difference is only the domain they work on. LMCN works on log-melspectral domain, while CMN works on melcepstral domain.

The best average performance of our proposed method shown in Table 18 is 93.2%, or 74.2% of ERR relative to the baseline, which is better than the ideal case (upper bound) for multi-condition AM. It is better than the performance of GSS and JSR approaches, where both approaches employ CMN. Comparing to our approach,

Table 15 *Environment-specific* NN training datasets for the experiments using real reverberant data

Dataset name	Speaker number	Utterances num. per spk. per env. (pairs of utts.)	Total duration of utterances (seconds)
5s.5u	5	1	14
5s.10u	5	2	27
1s.5u	1	5	17
1s.10u	1	10	35

The total duration of utterances is after removing the silence parts in the beginning and ending of each recording.

Table 16 Upper bound and baseline for the experiments using the real reverberant data

Method	Acoustic model	Digit accuracy (%)				
		InCar	Lounge	MeetR	Office	Average
Close (ideal)	Clean	97.3	96.5	97.3	98.1	97.3
	Multi	90.0	88.7	91.6	89.1	89.9
Baseline	Clean	76.4	43.8	89.2	85.1	73.6
	Multi	82.9	74.3	93.3	85.6	84.0
CMN	Clean	84.9	59.8	90.7	93.5	82.2
	Multi	80.1	83.1	89.9	84.4	84.4

JSR worked on clean AM, assumed that the environment is known, and used more training data (110 pairs of utterances). Our best performance is also better than Hybrid Delta approach. However, we should notice that comparing to our approach, Hybrid Delta do not need stereo data.

Table 19 shows the experimental results of our proposed approach on set D using '1s.20u' datasets. We had five '1s.20u' datasets (one dataset for each selected speaker), and Table 19 shows the average performance of these five datasets. The table shows that by using only one speaker, the proposed method could still perform relatively well. The best average performance of our proposed method shown in Table 19 is 91.8%, which is 68.9% of ERR relative to the baseline. It is also better than the performance of other approaches which were already presented above.

7.4.2 Further results and analyses

Figure 12 tries to analyze the performance by using five-speaker datasets ('5s'), 'clean+multi-rt' AM, various numbers of NN training data, various lengths of segment, and the use of CMN. The lengths of frame segment range from 5-frame until 33-frame. Because of the use of skip1 frame selection, it means that the context length we considered varies from 145 ms (5-frame) until 985 ms (33-frames).

In general, CMN did not improve the performance. We can observe that the use of CMN gave lower performance in most cases, although it was not significant. If we only observe this experimental results (using real reverberant data), we may think that the CMN was not effective because we included the function of CMN as LMCN. However, the experimental results for simulated reverberant data (Subsubsection 7.2.2) also show that the use of CMN after the feature mapping was not effective. Note that in the experiments using simulated reverberant data, we did not employ LMCN. This lack of performance is also confirmed and shown in [54]. Because of this lack of performance, Hybrid Delta also did not employ CMN [55].

We can observe that by limited stereo data, we could get good performance, as long as we use appropriate segment length. The small dataset ('20u') tends to give the best performance when we used frame selections of skip1 12-1-0 and 16-1-0, while the large dataset ('80u') tends to give the best performance when we used longer frame selection (skip1 24-1-0). The best average performance of our proposed method shown in Figure 12 is 93.5%, which is 75.4% of ERR relative to the baseline, reached by using skip1 24-1-0 on '60u' and '80u' datasets.

Figure 13 tries to analyze the performance by using one-speaker datasets ('1s'), 'clean+multi-rt' AM, various numbers of NN training data, various lengths of segment, and the use of CMN. We can also observe that the limited stereo data ('20u' and '40u') tends to give the best performance when we used frame selections of skip1 12-1-0. The best average performance of our proposed method shown in Figure 13 is 92.5%, which is 71.6% of ERR relative to the baseline, reached by using skip1 12-1-0 on '40u' datasets.

In a real application, the use of one-speaker dataset is more practical because it is easier to acquire stereo data of many utterances by one speaker. The performance was not as good as the performance of five-speaker

Table 17 Experimental results on the real reverberant data using GSS and JSR [10]

Method	Acoustic model	Digit accuracy (%)				
		InCar	Lounge	MeetR	Office	Average
GSS	Multi-rt	86.32	76.67	85.65	83.29	82.98
GSS + CMN	Multi-rt	88.88	83.07	89.32	92.52	88.45
JSR	Clean	84.71	67.32	92.55	95.23	84.95
JSR + CMN	Clean	87.62	79.74	90.19	93.30	87.71

Table 18 Experimental results on the real reverberant data by using NN training data '5s.20u' (combined)

Method	Frame selection	Acoustic model	Digit accuracy (%)				
			InCar	Lounge	MeetR	Office	Average
LMCN + Proposed (24 NNs)	Linear 8-1-0	Clean	69.4	38.5	78.9	75.3	65.6
		Clean-rt	85.9	60.9	90.3	93.7	82.7
		Multi-rt	82.4	85.1	90.9	86.0	86.1
		Clean+multi-rt	89.6	91.0	94.4	93.5	92.1
	Linear 16-1-0	Clean	67.8	35.1	79.4	78.0	65.1
		Clean-rt	85.7	58.8	90.7	94.7	82.5
		Multi-rt	82.6	84.2	90.2	89.3	86.6
		Clean+multi-rt	89.2	90.9	94.4	93.5	92.0
	Linear 32-1-0	Clean	67.0	30.0	74.6	77.6	62.3
		Clean-rt	84.7	53.0	89.2	94.1	80.3
		Multi-rt	78.4	82.0	88.4	87.9	84.2
		Clean+multi-rt	86.5	89.2	93.1	92.0	90.2
	Skip1 8-1-0	Clean	69.1	38.0	78.4	75.6	65.3
		Clean-rt	85.3	61.3	90.9	93.5	82.8
		Multi-rt	81.0	82.5	91.2	84.2	84.7
		Clean+multi-rt	89.2	90.7	95.1	93.4	92.1
	Skip1 16-1-0	Clean	67.2	34.7	75.8	77.6	63.8
		Clean-rt	85.2	58.5	89.6	93.6	81.7
		Multi-rt	83.2	84.8	91.5	87.1	86.6
		Clean+multi-rt	90.0	92.9	95.5	94.3	**93.2**

The bold text represents the best average performance.

dataset because the lack of speaker variation in the training data may cause the NNs become speaker-dependent. However, the experimental results show that by using one-speaker dataset, we could still get a good robust speaker-independent system. Figures 12 and 13 show that the best performance for one-speaker and five-speaker datasets were 92.3% and 93.2%, respectively, for the dataset containing 20 pairs of utterances. Meanwhile, for the dataset containing 40 pairs, the best performance for one-speaker and five-speaker datasets were 92.5% and 93.5%, respectively.

8 Conclusions

In this work, we propose a single-channel non-linear regression-based dereverberation method using cascade NNs. The NNs were trained on stereo data to compensate the reverberation effect by mapping the reverberant feature in a log-melspectral domain to its corresponding anechoic feature.

In Section 6, we present the evaluation using SID system and the proposed method could perform very well by using only few stereo data (five pairs of utterances) as the NN training data. In Subsection 6.2, we did

Table 19 Experimental results on the real reverberant data by using NN training dataset '1s.20u' (combined)

Method	Frame selection	Acoustic model	Digit accuracy (%)				
			InCar	Lounge	MeetR	Office	Average
LMCN + Proposed (24 NNs)	Skip1 8-1-0	Clean-rt	84.9	58.5	90.7	93.6	81.9
		Multi-rt	80.9	82.7	90.4	83.6	84.4
		Clean+multi-rt	89.1	90.7	94.7	93.0	**91.8**
	Skip1 16-1-0	Clean-rt	84.5	55.4	89.5	93.7	80.7
		Multi-rt	78.2	80.5	88.5	84.5	82.9
		Clean+multi-rt	88.5	90.6	94.0	93.9	91.7

The bold text represents the best average performance.

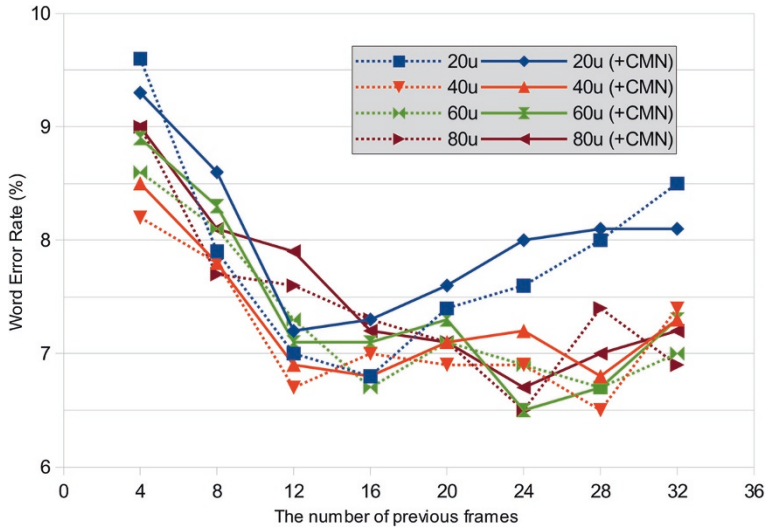

Figure 12 Performance analysis on the real reverberant data of CENSREC-4 by using five-speaker datasets ('5s'). The acoustic model was retrained by using clean and dereverberated multi-condition data ('clean+multi-rt').

experiments using various frame selection types, frame selection numbers, and training data numbers on several simulated noiseless and noisy reverberant environments. In general, the skip1 frame selection could perform better than the linear one. The use of longer context (frame selection number) could improve the performance, but the data for NN training should be sufficient. The non-causal reverberation model could perform better than the causal reverberation model. However, the use of left context only is still reasonable on a theoretical point of view. The single NN configuration could perform better for small training dataset, but the performance could not surpass the multiple NNs when much more training data was used, which is most likely because of channel transformation overgeneralization problem. Note that the RTFs, which are represented by the NNs, should be frequency-dependent. For example, the best ERR (regardless the frame selection) for RIR type of 'living room' by using only 1 pair of utterances and single NN reached 26.0%, while by using 15 pairs of utterances and multiple NN reached 62.6% (Table 2).

In Subsection 6.3, we did experiments on modified multiple NNs (by modifying our assumptions on the channel transformation) and using NNs trained using combined

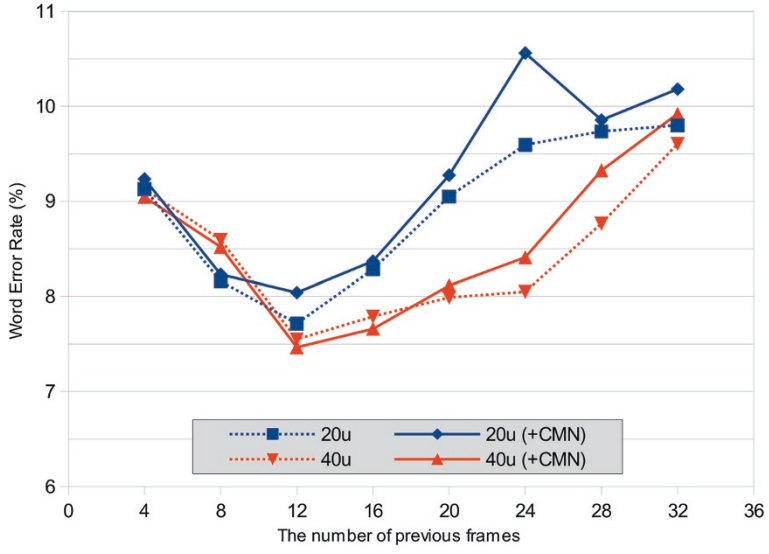

Figure 13 Performance analysis on the real reverberant data of CENSREC-4 by using 1-speaker datasets ('1s'). The acoustic model was retrained by using clean and dereverberated multi-condition data ('clean+multi-rt').

training data of known conditions in a room for dereverberating utterances from unknown condition in the same room. For a very limited stereo data (e.g., one or five pairs of utterances), the use of *modified* multiple NNs configuration could improve the performance. By using the '6 NNs' configuration, the skip1 3-1-0 frame selection, and only one pair of utterances for the NN training data, we could reach 34.8% of ERR relative to the use of CMN. Our results also show that by combining the training data from known positions in a room, we could train NNs which generalize well, so that the NNs could also perform well for unknown positions in the same room. In addition, the use of utterances from one position could perform reasonably well for other unknown positions in the same room. By using multiple NNs and 15 pairs of utterances by 1 speaker from 3 positions, we could reach 93.7% of average identification rate, which was 42.2% of ERR relative to the use of CMN.

In Section 7, we present the evaluation using ASR system, where CENSREC-4 framework [22] was used. In the experiments using simulated reverberant data, we could reach 78.4% of ERR relative to the baseline by using '5s.40u' dataset (5 speakers, 40 pairs of utterances). By using the same number of utterances (40 pairs of utterances) in the experiments using real reverberant data, we could reach 75.4% and 71.6% of ERR relative to the baseline by using '5s.40u' (5 speakers) and '1s.40u' (1 speaker).

Our experimental results could show that by using limited numbers of stereo data, our proposed method could perform remarkably well. However, the need of stereo data in implementing the method could be regarded as troublesome. Thus, possible future research directions are omitting the need of stereo data and developing unsupervised dereverberation method.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]Department of Computer Science and Engineering, Toyohashi University of Technology, Toyohashi, Aichi 441-8580, Japan. [2]Department of Information and Computer Engineering, Toyota National College of Technology, Toyota, Aichi 471-8525, Japan.

Received: 27 September 2013 Accepted: 28 March 2014
Published: 10 April 2014

References

1. B Raj, T Virtanen, R Singh, The problem of robustness in automatic speech recognition, in *Techniques for Noise Robustness in Automatic Speech Recognition*, ed. by T Virtanen, R Singh, and B Raj (Wiley West Sussex, 2013)
2. T Yoshioka, A Sehr, M Delcroix, K Kinoshita, R Maas, T Nakatani, W Kellermann, Making machines understand us in reverberant rooms: robustness against reverberation for automatic speech recognition. IEEE Signal Process. Mag. **29**(6), 114–126 (2012)
3. T Nakatani, T Yoshioka, K Kinoshita, M Miyoshi, B-H Juang, Speech dereverberation based on variance-normalized delayed linear prediction. IEEE Trans. Audio Speech Lang. Process. **18**(7), 1717–1731 (2010)
4. T Yoshioka, T Nakatani, M Miyoshi, HG Okuno, Blind separation and dereverberation of speech mixtures by joint optimization. IEEE Trans. Audio Speech Lang. Process. **19**(1), 69–84 (2011)
5. K Lebart, JM Boucher, A new method based on spectral subtraction for speech dereverberation. Acta Acustica. **87**, 359–366 (2001)
6. EAP Habets, Single-channel speech dereverberation based on spectral subtraction, in *Proceedings of the 15th Annual Workshop on Circuits, Systems and Signal Processing (ProRISC)* (STW Technology Foundation Utrecht, Netherlands, 250)
7. JS Erkelens, R Heusdens, Correlation-based and model-based blind single-channel late-reverberation suppression in noisy time-varying acoustical environments. IEEE Trans. Audio Speech Lang. Process. **18**(7), 1746–1765 (2010)
8. A Mushtaq, C-H Lee, An integrated approach to feature compensation combining particle filters and hidden Markov models for robust speech recognition, in *Proceedings of IEEE ICASSP* (IEEE Piscataway, NJ, USA, 2012), pp. 4757–4760
9. V Leutnant, A Krueger, R Haeb-Umbach, Bayesian feature enhancement for reverberation and noise robust speech recognition. IEEE Trans. Audio Speech Lang. Process. **21**(8), 1640–1652 (2013)
10. W Li, L Wang, F Zhou, Q Liao, Joint sparse representation based cepstral-domain dereverberation for distant-talking speech recognition, in *Proceedings of IEEE ICASSP* (IEEE Piscataway, NJ, USA, 2013), pp. 7117–7120
11. H-G Hirsch, H Finster, A new approach for the adaptation of HMMs to reverberation and background noise. Speech Comm. **50**(3), 244–263 (2008)
12. A Sehr, R Maas, W Kellermann, Reverberation model-based decoding in the logmelspec domain for robust distant-talking speech recognition. IEEE Trans. Audio Speech Lang. Process. **18**(7), 1676–1691 (2010)
13. M Delcroix, T Hikichi, M Miyoshi, Dereverberation and denoising using multichannel linear prediction. IEEE Trans. Audio Speech Lang. Process. **15**(6), 1791–1801 (2007)
14. E Habets, J Benesty, I Cohen, S Gannot, J Dmochowski, New insights into the MVDR beamformer in room acoustics. IEEE Trans. Audio Speech Lang. Process. **18**(1), 158–170 (2010)
15. L Wang, K Odani, A Kai, Dereverberation and denoising based on generalized spectral subtraction by multi-channel LMS algorithm using a small-scale microphone array. EURASIP J. Adv. Signal Process. **2012**(1) (2012)
16. BS Atal, Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification. J. Acoust. Soc. Am. **55**(6), 1304–1312 (1974)
17. S Furui, Cepstral analysis technique for automatic speaker verification. IEEE Trans. Acoust. Speech Signal Process. **29**(2), 254–272 (1981)
18. C Avendaño, S Tibrewala, H Hermansky, Proceedings of EUROSPEECH (ISCA Baixas, France, 1997), pp. 1107–1110
19. PJ Moreno, B Raj, RM Stern, A vector Taylor series approach for environment-independent speech recognition, in *Proceedings of IEEE ICASSP*, vol. 2 (IEEE Piscataway, NJ, USA, 1996), pp. 733–736
20. M Wolfel, Enhanced speech features by single-channel joint compensation of noise and reverberation. IEEE Trans. Audio Speech Lang. Process. **17**(2), 312–323 (2009)
21. A Krueger A, R Haeb-Umbach, Model-based feature enhancement for reverberant speech recognition. IEEE Trans. Audio Speech Lang. Process. **18**(7), 1692–1707 (2010)
22. T Fukumori, T Nishiura, M Nakayama, Y Denda, N Kitaoka, T Yamada, K Yamamoto, S Tsuge, M Fujimoto, T Takiguchi, C Miyajima, S Tamura, T Ogawa, S Matsuda, S Kuroiwa, K Takeda, S Nakamura, CENSREC-4: an evaluation framework for distant-talking speech recognition in reverberant environments. Acoust. Sci. Tech. **32**(5), 201–210 (2011)
23. L Deng, A Acero, L Jiang, J Droppo, X Huang, High-performance robust speech recognition using stereo training data, in *Proceedings of IEEE ICASSP*, vol. 1 (IEEE Piscataway, NJ, USA, 2001), pp. 301–304
24. J Droppo, L Deng, A Acero, Evaluation of SPLICE on the Aurora 2 and 3 tasks, in *Proceedings of ICSLP* (ISCA Baixas, France, 2002), pp. 29–32
25. Y Pan, A Waibel, *The effects of room acoustics on MFCC speech parameter* (ISCA, Baixas, France, 2000), pp. 129–132
26. K Shimada, K Yamamoto, S Nakagawa, Consideration of robust speaker recognition for reverberation in distant speech, in *Proceedings of the Spring Meeting of Acoustical Society of Japan* (ASJ Tokyo, Japan, 2012), pp. 199–202. (in Japanese)

27. W Li, Y Zhou, N Poh, F Zhou, Q Liao, Feature denoising using joint sparse representation for in-car speech recognition. E Signal Process. Lett. **20**(7), 681–684 (2013)
28. P Jinachitra, RE Prieto, Towards speech recognition oriented dereverberation, in *Proceedings of IEEE ICASSP* (IEEE Piscataway, NJ, USA, 2005), pp. 437–440
29. T Ishii, T Shinozaki, Y Horiuchi, S Kuroiwa, Reverberant speech recognition based on denoising autoencoder using multiple analysis window lengths, in *Proceedings of the Spring Meeting of Acoustical Society of Japan* (ASJ Tokyo, Japan, 2013), pp. 65–66. (in Japanese)
30. T Ishii, H Komiyama, T Shinozaki, Y Horiuchi, S Kuroiwa, Reverberant speech recognition based on denoising autoencoder, in *Proceedings of INTERSPEECH* (ISCA Baixas, France, 2013), pp. 3512–3516
31. Xia B-y, Bao C-c, Speech enhancement with weighted denoising auto-encoder, in *Proceedings of INTERSPEECH* (ISCA Baixas, France, 2013), pp. 3444–3448
32. X Lu, Y Tsao, S Matsuda, C Hori, Speech enhancement based on deep denoising autoencoder, in *Proceedings of INTERSPEECH* (ISCA Baixas, France, 2013), pp. 436–440
33. Rojas R (ed.), *Neural Networks: A Systematic Introduction*. (Springer, Berlin, 1996)
34. E Fiesler, Neural network classification and formalization. Comput. Stand. Interfac. **16**, 1994
35. S Nissen, *Large scale reinforcement learning using q-sarsa(λ) and cascading neural networks*. (PhD thesis, University of Copenhagen, Denmark, 2007)
36. L Prechelt, Investigation of the CasCor family of learning algorithms. Neural Netw. **10**, 885–896 (1996)
37. SE Fahlman, C Lebiere, The cascade-correlation learning architecture, in *Advances in Neural Information Processing Systems 2*, ed. by Touretzky DS (Morgan Kaufmann San Francisco, 1990), pp. 524–532
38. AA Nugraha, S Nakagawa, Proceedings of the Autumn Meeting of Acoustical Society of Japan (ASJ Tokyo, Japan, 2012), pp. 163–166
39. AA Nugraha, K Yamamoto, S Nakagawa, Single channel dereverberation method in log-melspectral domain using limited stereo data for distant speaker identification, in *Proceedings of APSIPA Annual Summit and Conference* (IEEE Piscataway, NJ, USA, 2013)
40. AA Nugraha, K Yamamoto, S Nakagawa, Single channel dereverberation method by feature mapping using limited stereo data. Technical Report. **113**(161), 7–12 (2013). SP2013-54, IEICE
41. L Wang, N Kitaoka, S Nakagawa, Distant-talking speech recognition based on spectral subtraction by multi-channel LMS algorithm. IEICE Trans. Inf. Syst. **E94-D**(3), 659–667 (2011)
42. M Riedmiller, H Braun, A direct adaptive method for faster backpropagation learning: the RPROP algorithm, in *Proceedings of IEEE International Conference on Neural Networks*, vol. 1 (IEEE Piscataway, NJ, USA, 1993), pp. 586–591
43. S Nissen, FANN: Fast Artificial Neural Network Library (2013). http://leenissen.dk/fann. Accessed 8 September 2013
44. S Nissen, Implementation of a fast artificial neural network library (FANN). Technical report, University of Copenhagen, Denmark (2007)
45. KP Markov, S Nakagawa, Text-independent speaker recognition using non-linear frame likelihood transformation. Speech Commun. **24**(3), 193–209 (1998)
46. S Nakagawa, L Wang, S Ohtsuka, Speaker identification and verification by combining MFCC and phase information. IEEE Trans. Audio Speech Lang. Process. **20**(4), 1085–1095 (2012)
47. K Itou, M Yamamoto, K Takeda, JNAS: Japanese speech corpus for large vocabulary continuous speech recognition research. J. Acoust. Soc. Jpn. **20**(3), 199–206 (1999)
48. HG Hirsch, D Pearce, Aurora-5 (2007). http://aurora.hsnr.de/aurora-5.html. Accessed 8 September 2013
49. HG Hirsch, SIREAC - SImulation of REal ACoustics (Web Demo) (2005). http://dnt.kr.hs-niederrhein.de/wwwsim. Accessed 8 September 2013
50. HG Hirsch, H Finster, Proceedings of INTERSPEECH (ISCA Baixas, France, 2005), pp. 2697–2700
51. AY Nakano, S Nakagawa, K Yamamoto, Automatic estimation of position and orientation of an acoustic source by a microphone array network. J. Acoust. Soc. Am. **126**(6), 3084–3094 (2009)
52. AY Nakano, *Exploring spatial information for distant speech recognition under real environmental conditions*. (PhD thesis, Toyohashi University of Technology, Japan, 2010)
53. PA Naylor, EAP Habets, JY-C Wen, ND Gaubitch, Models, measurement and evaluation, in *Speech Dereverberation*, ed. by Naylor PA, Gaubitch ND (Springer London, 2010), pp. 21–56
54. M Nakayama, T Nishiura, Y Denda, N Kitaoka, K Yamamoto, T Yamada, S Tsuge, C Miyajima, M Fujimoto, T Takiguchi, S Tamura, T Ogawa, S Matsuda, S Kuroiwa, K Takeda, S Nakamura, CENSREC-4: development of evaluation framework for distant-talking speech recognition under reverberant environments, in *Proceedings of INTERSPEECH* (ISCA Baixas, France, 2008), pp. 968–971
55. O Ichikawa, T Fukuda, M Nishimura, Dynamic features in the linear-logarithmic hybrid domain for automatic speech recognition in a reverberant environment. IEEE J. Sel. Top. Signal. Process. **4**(5), 816–823 (2010)

doi:10.1186/1687-4722-2014-13
Cite this article as: Nugraha *et al.*: **Single-channel dereverberation by feature mapping using cascade neural networks for robust distant speaker identification and speech recognition.** *EURASIP Journal on Audio, Speech, and Music Processing* 2014 **2014**:13.