IEICE TRANS. INF. & SYST., VOL.E101–D, NO.2 FEBRUARY 2018

PAPER

DNN-Based Speech Synthesis Using Speaker Codes*

Nobukatsu HOJO[†a)], **Yusuke IJIMA**[†], ***and*** **Hideyuki MIZUNO**[††], ***Members***

SUMMARY Deep neural network (DNN)-based speech synthesis can produce more natural synthesized speech than the conventional HMM-based speech synthesis. However, it is not revealed whether the synthesized speech quality can be improved by utilizing a multi-speaker speech corpus. To address this problem, this paper proposes DNN-based speech synthesis using speaker codes as a method to improve the performance of the conventional speaker dependent DNN-based method. In order to model speaker variation in the DNN, the augmented feature (speaker codes) is fed to the hidden layer(s) of the conventional DNN. This paper investigates the effectiveness of introducing speaker codes to DNN acoustic models for speech synthesis for two tasks: multi-speaker modeling and speaker adaptation. For the multi-speaker modeling task, the method we propose trains connection weights of the whole DNN using a multi-speaker speech corpus. When performing multi-speaker synthesis, the speaker code corresponding to the selected target speaker is fed to the DNN to generate the speaker's voice. When performing speaker adaptation, a set of connection weights of the multi-speaker model is re-estimated to generate a new target speaker's voice. We investigated the relationship between the prediction performance and architecture of the DNNs through objective measurements. Objective evaluation experiments revealed that the proposed model outperformed conventional methods (HMMs, speaker dependent DNNs and multi-speaker DNNs based on a shared hidden layer structure). Subjective evaluation experimental results showed that the proposed model again outperformed the conventional methods (HMMs, speaker dependent DNNs), especially when using a small number of target speaker utterances.

key words: *speech synthesis, acoustic model, deep neural network, speaker codes*

1. Introduction

Recent studies have shown that deep neural network (DNN)-based speech synthesis [2]–[4] can produce more natural synthesized speech than the conventional hidden Markov model (HMM)-based speech synthesis. However, DNN-based speech synthesis requires a considerable amount of speech data uttered by the target speaker to obtain sufficient performance. The problem then becomes the high cost required to generate speech from various speakers from a DNN because we need a considerable amount of speech data from all the speakers the system uses, and annotations of phonetic and prosodic contextual information on them.

Manuscript received May 18, 2017.
Manuscript revised September 24, 2017.
Manuscript publicized November 1, 2017.

[†]The authors are with NTT Media Intelligence Laboratories, NTT Corporation, Yokosuka-shi, 239–0847 Japan.

[††]The author is with Tokyo University of Science, Suwa, Chino-shi, 391–0292 Japan.

[*]A part of this paper was presented at INTERSPEECH 2016 [1].

a) E-mail: hojo.nobukatsu@lab.ntt.co.jp

DOI: 10.1587/transinf.2017EDP7165

In the field of HMM-based speech synthesis, many techniques have succeeded in generating speech from a smaller amount of target speaker data. A powerful method is an average-voice-based speech synthesis technique with model adaptation [5]. In this technique, average voice models are created from several speakers' speech data and are adapted to a small amount of speech data from a target speaker using model adaptation algorithms such as CSMAPLR [6]. Another successful method is based on cluster adaptive training (CAT) [7]. This model has multiple compact decision trees that are interpolated to produce a huge variety of possible contexts, and is trained using a multi-speaker speech corpus to improve the speech quality.

Motivated by these previous studies in HMM-based speech synthesis, we aimed in our work to improve the synthetic speech quality from a DNN by using a multi-speaker speech corpus. One possible approach to train a DNN using a multi-speaker speech corpus is adopting the shared hidden layer structure [8]. The model structure is composed of the same hidden layers shared among different speakers and the output layers composed of speaker-dependent nodes. It has been shown that this model structure can improve synthesized speech quality for both multi-speaker modeling and speaker adaptation tasks. Another approach to train a DNN using a multi-speaker speech corpus is to feed augmented speaker specific features to the network in order to incorporate speaker-level information to the DNNs. In the speech recognition, for example, this approach has been shown to be an effective way to feed i-vectors [9] or speaker codes [10], [11] as augmented features. As for DNN-based speech synthesis, a recent study [12] conducted an experimental analysis using i-vector based feature augmentation. This work is based on the assumption that speaker code-based features can also be effectively introduced to DNNs for speech synthesis.

This paper proposes to use augmented features based on speaker codes as a relatively simple method for multi-speaker modeling and speaker adaptation for DNN-based speech sythesis. Initially we proposed using speaker codes for DNN-based speech synthesis [1], and it has been shown that this approach can achieve better synthetic speech quality than can be obtained with conventional speaker dependent DNNs and speaker adaptation using HMMs. Luong et al. [13] also proposed using the features of speaker codes for DNN-based speech synthesis. Their main focus is to control the speech characteristics by modifying the augmented features and it has been shown that such modification can be

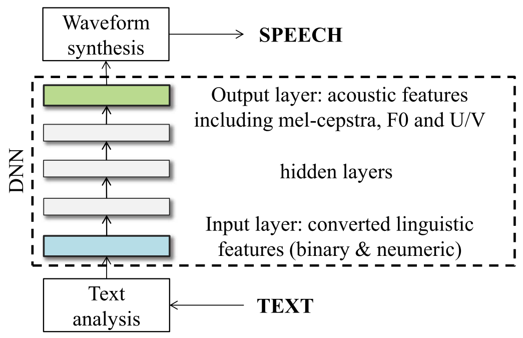

Waveform synthesis
SPEECH
DNN
Output layer: acoustic features including mel-cepstra, F0 and U/V
hidden layers
Input layer: converted linguistic features (binary & neumeric)
Text analysis
TEXT

Fig. 1 The model architecture of the conventional speaker dependent DNN.

performed effectively by manipulating the input code vectors. However, it still has not been revealed which approach, the augmented feature-based approach or the shared hidden layer approach, is better able to model multi-speaker characteristics for DNN-based speech synthesis. Adding to the contribution of our previous study [1], we describe in this paper how we conducted an objective evaluation experiment to compare the performance of the proposed speaker code-based approach with that of the conventional shared hidden layer approach.

2. Conventional DNN-Based Speech Synthesis

2.1 Speaker Dependent DNN [2]

The baseline model is a DNN acoustic model similar to the one described by Zen et al. [2]. The model is illustrated in Fig. 1. The DNN is used as a function to map the linguistic feature vectors to acoustic feature vectors. First, the input text is converted to the linguistic feature vector. The input features include binary answers to questions about linguistic contexts and numeric values. Then the linguistic feature vector is mapped to the output feature by forward propagation of DNN. The output features include spectral and excitation parameters and their time derivatives. The baseline model is trained using only the target speaker utterances in the training corpus.

3. Multi-Speaker Modeling Using Speaker Codes

There are two possible model architectures to introduce speaker codes into the DNN-based speech synthesis. One is similar to the architecture proposed in the speech recognition field [10], [11], which is illustrated in Fig. 2. We call this architecture “embedded speaker code” in this paper. The other is a simplified version of the embedded speaker code model illustrated in Fig. 3. This model architecture does not embed the speaker ID vector and directly feeds it to hidden layers. We call this architecture “one-hot speaker code” in this paper.

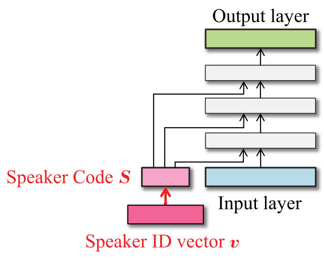
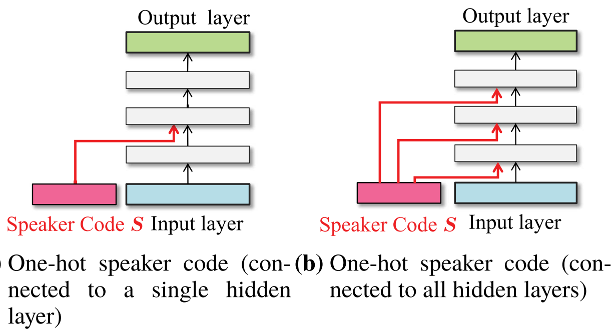

Output layer
Speaker Code $\mathcal{S}$
Input layer
Speaker ID vector $\boldsymbol{v}$

Fig. 2 The model architecture of the embedded speaker code-based speech synthesis. The red bold lines indicate the re-estimated parameters for speaker adaptation.



(a) One-hot speaker code (connected to a single hidden layer) **(b)** One-hot speaker code (connected to all hidden layers)

Fig. 3 The model architecture of the one-hot speaker code-based speech synthesis. The red bold lines indicate the re-estimated parameters for speaker adaptation.

3.1 Embedded Speaker Code-Based Speech Synthesis

As shown in Fig. 2, in this model architecture, a speaker ID is embedded to a speaker code $\mathcal{S}$ through a set of connection weights. Here the speaker code $\mathcal{S}$ represents the speaker information. The speaker codes are fed to all hidden layers. The speaker ID vector $\boldsymbol{v} = [v_1, \cdots, v_K]^{\mathrm{T}}$ for speaker m is set to the following fixed 1-of-K form:

$$v_k = \begin{cases} 1 & (k = m) \\ 0 & (k \neq m) \end{cases} \tag{1}$$

where K is the number of speakers in the training data.

For the formulation below, the connection weight from the speaker code $\mathcal{S}$ to the k-th hidden layer is represented by $W_{\mathcal{S}}^k$. The vector fed by speaker code $\mathcal{S}$ to the k-th hidden layer is represented by $\boldsymbol{y}^k$.

For embedded speaker codes, we can get

$$\begin{aligned} \boldsymbol{y}^k &= W_{\mathcal{S}}^k \mathcal{S} \\ &= W_{\mathcal{S}}^k W_E \boldsymbol{v}, \end{aligned} \tag{2}$$

where $\boldsymbol{v}$ is a one-hot vector that represents the speaker ID in (1) and W_E is the embedding connection weight from the speaker ID vector $\boldsymbol{v}$ to speaker codes $\mathcal{S}$. The size of each vector or matrix is:

$$\boldsymbol{y}^k : \ D_h^k \tag{3}$$

$$W_S : \ D_h^k \times D_S \tag{4}$$

$$W_E : D_S \times K \tag{5}$$

$$\boldsymbol{v} : K. \tag{6}$$

By setting $D_S < D_h^k$ and $D_S < K$, we can constrain the matrix $W_S^k W_E$ to be low rank (=D_S) and $\boldsymbol{y}^k$ to be in D_S-dimensional linear subspace. This constraint is expected to make the model robust even when the amount of training data for a speaker is limited.

3.2 One-Hot Vector Speaker Code-Based Speech Synthesis

In this model architecture, the speaker code $\mathcal{S} = [\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_K]^{\mathrm{T}}$ for speaker m is set to the same formulation as the one-hot speaker ID vector $\boldsymbol{v}$ in Eq. (1).

$$\mathcal{S}_k = \begin{cases} 1 & (k = m) \\ 0 & (k \neq m) \end{cases} \tag{7}$$

As shown in Fig. 3, a speaker code $\mathcal{S}$ is fed through an additional set of connection weights to a certain hidden layer (Fig. 3 (a)) or all hidden layers (Fig. 3 (b)).

For one-hot vector speaker codes, we can get

$$\boldsymbol{y}^k = W_S^k \mathcal{S}, \tag{8}$$

where $\mathcal{S}$ is a one-hot vector that represents the speaker ID in (7). The size of each vector or matrix is:

$$\boldsymbol{y}^k : D_h^k \tag{9}$$

$$W_S^k : D_h^k \times K \tag{10}$$

$$\mathcal{S} : K. \tag{11}$$

It can be seen that W_S^k is full rank and $\boldsymbol{y}^k$ has no constraint, unlike the embedded speaker code model in (2). This means that one-hot vector speaker code models may model speaker characteristics precisely using larger number of parameters while they can be less robust for limited training data when compared with embedded speaker code models.

3.3 Training Procedure for Multi-Speaker Modeling Using Speaker Codes

For multi-speaker modeling based on embedded speaker codes and one-hot speaker codes, the proposed methods train connection weights of the whole DNN using a multi-speaker speech corpus. When synthesizing a speech parameter sequence, a target speaker is chosen from the corpus and the speaker code corresponding to the selected target speaker is fed to the DNN to generate the speaker's voice. The proposed model is expected to generate more stable and natural speech compared with conventional speaker dependent DNNs because the networks from the linguistic feature to the acoustic feature are trained with a greater variety of contextual information by using a multi-speaker speech corpus.

4. Speaker Adaptation Using Speaker Codes

The multi-speaker modeling approach will need high computational cost every time we want to generate speech from a new target speaker. This is because the whole model needs to be retrained using a corpus including the new speaker's utterances. It is considered that model adaptation by re-estimating a subset of model parameters using the target speaker utterances can address this problem. Following up on the previous research work that has been done in the field of speaker adaptation for speech synthesis [12], [14]–[16], we consider that the speaker code based DNN can also be used as a speaker adaptation method, since this approach has been shown to be effective in speech recognition [10], [11]. The set of parameters re-estimated for adaptation is set differently than that for the model architectures.

4.1 Speaker Adaptation Using Embedded Speaker Codes

The adaptation procedure described in this section is similar to those in [10], [11]. First, the connection weights of the whole DNN are trained using a multi-speaker speech corpus. The speaker ID vector $\boldsymbol{v}$ is set in a form similar to that given in Sect. 3.1, but this time $\boldsymbol{v}$ is appended by an additional dimension to have $K+1$ dimensions in total. This additional dimension is used to represent unseen target speaker information, and always set to 0 in the training procedure. Second, the model is adapted to a new target speaker using only the target speaker utterances as adaptation data. This time the additional dimension of $\boldsymbol{v}$ is set to 1 and other dimensions to 0, and only the connection weights for embedding are re-estimated to minimize the distance between the output features of the adaptation data and predicted values. When synthesizing, a one-hot vector $\boldsymbol{v}$ whose additional dimension is set to 1 is used. This adaptation procedure corresponds to updating only the $K+1$-th column of W_E in (2).

The adaptation algorithm amounts searching a new point of $\boldsymbol{y}^k$ to model a new target speaker. Here, by setting $D_S < D_h^k$ we can constrain the search space of $\boldsymbol{y}^k$ to be in the D_S-th linear subspace. This constraint decreases the number of free parameters for adaptation. Therefore, it is expected to make the model robust even when the amount of adaptation data is limited. The adaptation procedure for embedded speaker codes described above was originally proposed in the speech recognition field [10], [11] in order to make an adapted model robust to a limited amount of adaptation data by reducing the number of adaptation parameters. Specifically, the number of adaptation parameters for embedded speaker codes is reduced to D_S,while that in Sect. 4.2 is D_h. However, it is not clear whether this reduction is effective for speech synthesis. In order to confirm this point, we conducted experiments to compare these two methods.

4.2 Speaker Adaptation Using One-Hot Speaker Codes

While a DNN using one-hot speaker codes is adapted in the similar procedures similar to those of embedded speaker code models, this time the set of re-estimated parameters is different. First, the connection weights of the whole DNN are trained using a multi-speaker speech corpus. The speaker code $\mathcal{S}$ is set in a form similar to that of speaker ID vector $\boldsymbol{v}$ in the embedded speaker code. The vector $\mathcal{S}$ has $K + 1$ dimensions in total. The additional dimension is always set to 0 in the training procedure and 1 in the adaptation and synthesis procedure. For the adaptation procedure, only the connection weights from the speaker codes $\mathcal{S}$ to the hidden layers are re-estimated to minimize the distance between the output features of the adaptation data and predicted values. This adaptation procedure corresponds to updating only the $K + 1$-th column of $W^k_{\mathcal{S}}$ in (8). This time the number of parameters for adaptation is D^k_h.

This adaptation procedure is different from those reported in previous speech recognition studies [10], [11] and the procedure described in Sect. 4.1. These references for speaker adaptation for speech recognition mainly focus on fast and robust adaptation with a limited amount of adaptation data. On the other hand, for speech synthesis, it is necessary to model the speaker characteristics precisely to generate the target speaker's speech. The procedure for this model is expected to be flexible and more appropriate for speech synthesis, because it re-estimates a set of parameters whose dimensions are generally larger than those of $\mathcal{S}$ for adaptation.

Another possible adaptation procedure for one-hot speaker codes was proposed by Luong et al. [13]. They proposed to update only $\mathcal{S}$ in (8) for a new target speaker. This time the number of parameter for adaptation is K, which is usually smaller than the number for D^k_h. This constraint is expected to have effect similar to embedded speaker code models. We adopted the proposed adaptation procedure because it enabled us to see whether or not a large number of parameter for adaptation is needed by comparing the performance of one-hot and embedded speaker code models.

5. Experiments

As described in Sect. 3, there are various possible architectures for DNN-based speech synthesis using speaker codes. In this section, we will first explain how we investigated the relationship between the model architecture and the proposed method's performance through objective measurements. We determined the best architecture for the proposed method on the basis of the measurement results. We then conducted a subjective evaluation to confirm whether our method could improve the speech quality compared with the conventional speaker dependent DNN. We also compared the results with those obtained by HMMs as well as those by speaker dependent DNNs in these experiments. This is because prediction performance obtained by HMM is known to be superior to that obtained by speaker dependent DNN [2], In Sect. 5.1, we present the experimental conditions we used. We describe the objective evaluation experiments in Sect. 5.2 and the subjective experiments in Sect. 5.3.

5.1 Experimental Setup

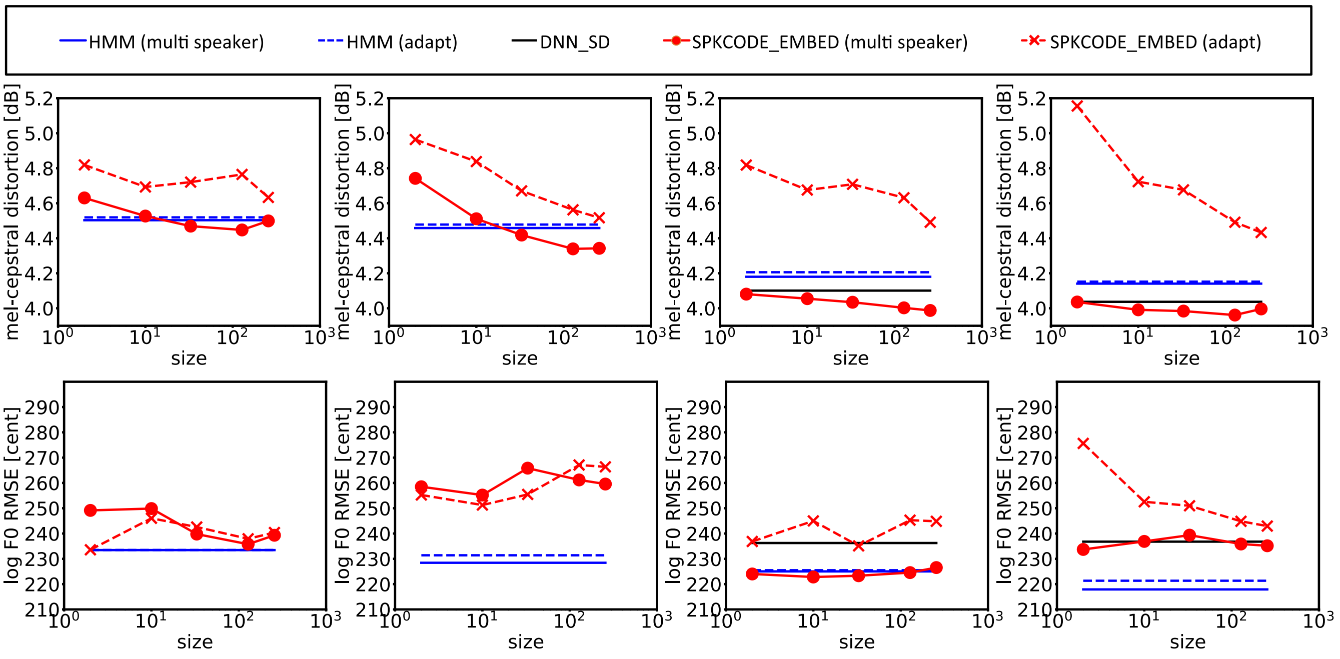
In the experiments, we used speech data in Japanese obtained from 35 speakers (17 males and 18 females). Two speakers, one male and one female, were used as target speakers. The training corpus included 7,260 utterances (about 1,340 minutes) from 33 speakers apart from the two target speakers. We conducted objective evaluation for two tasks: multi-speaker modeling and speaker adaptation. For each task, we considered two training conditions: five utterances (about 1.2 minutes) and 300 utterances (about 63 minutes) for the training corpus for each target speaker. Twenty utterances were used as a testing set for each target speaker. The sampling rate of the corpus was 22.05 kHz. The STRAIGHT vocoder [17] was employed to extract 40 dimensional mel-cepstral coefficients, five band aperiodicities, and F0 in log-scale at 5 msec steps.

We compared the performance obtained with the following five acoustic models.

- HMM: The conventional HMM-based average voice model with adaptation.
- DNN_SD: The conventional speaker dependent model [2].
- SPKCODE_EMBED: The proposed method using embedded speaker codes described in Sect. 3.1.
- SPKCODE_ONE-HOT: The proposed method using one-hot speaker codes described in Sect. 3.2.

The input linguistic feature vector of a DNN contained 506 dimensional linguistic features. Each observation vector consisted of 40 mel-cepstral coefficients, log F0, five band aperiodicities, their delta and delta-delta features, and a voiced/unvoiced binary value. The input numeric features were normalized to the 0.01-0.99 range, and the output features were normalized by speaker-dependent mean and variance. The DNN systems had five hidden layers and each hidden layer had 1024 units. A sigmoid activation function was used in the hidden layers followed by a linear activation at the output layer. For the training procedure, the weights of the DNN were initialized randomly with Gaussian distribution with zero mean and variance $\frac{1}{\sqrt{D}}$, where D denotes the dimension of an input vectors. The initial values for biases of the DNN were set to zero. The weights and biases were then optimized to minimize the mean squared error between the output features of the training data and predicted values, using the Adam-based back-propagation algorithm [18]. The parameters for the Adam algorithm were set as $\alpha = 0.0001, \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1\mathrm{e}-8$. Five percent of the utterances of the whole training data were used as a development set. The DNN_SD models were trained using only the target speaker utterances in the training corpus. We did not train DNN_SD models using five utterances because the amount of data in the training corpus was too small to train a model properly.

For the HMM, we used a five-state left-to-right hidden



(a) five utterances for male target speaker **(b)** five utterances for female target speaker **(c)** 300 utterances for male target speaker **(d)** 300 utterances for female target speaker

Fig. 4 The objective evaluation results for embedded speaker code model. The horizontal axis indicates the size (dimension) of the speaker codes.

semi-Markov model with no skip topology. Each observation vector consisted of 138 features (40 mel-cepstral coefficients, log F0, five band aperiodicities, and their delta and delta-delta features). The output distribution in each state was modeled as a single Gaussian density function and the covariance matrices were assumed to be diagonal. For the multi-speaker training task, the average voice model was trained using a multi-speaker speech corpus including the target speaker utterances. Then the average voice model was adapted using the target speaker utterances again. For the speaker adaptation task, on the other hand, the target speaker utterances were excluded from the corpus for training the average voice model. The average voice model was then adapted using the target speaker utterances. We used the combined technique of CSMAPLR and MAP adaptation as the speaker adaptation algorithm [6]. The model size was determined automatically by the minimum description length (MDL) criterion [19], where the control parameter of the model size was set to $\alpha = 1.0$.

For all of the four methods evaluated in these experiments, segmentations (phoneme durations) from natural speech were used instead of predicting duration. We applied MLPG [20] to the output features for all of the four methods. For DNN-based methods, we used pre-computed variances from the training data for MLPG. We did not apply spectral enhancement techniques such as global variance [21] to reduce factors considered in the experiments.

5.2 Objective Evaluation

We first investigated the relationship between the prediction performance and the architectures of the proposed method. We changed the embedded speaker code dimension for SPKCODE_EMBED models. The embedding dimensions were set to 2, 10, 33, 128 and 256. For SPKCODE_ONE-HOT models, the input hidden layer(s) for speaker codes were changed. The input hidden layer(s) for SPKCODE_ONE-HOT models were set to the 1st, 2nd, 3rd, 4th, 5th hidden layer and all hidden layers. We used objective measurements to compare the performances we obtained in these cases with those obtained with conventional HMM and DNN_SD.

5.2.1 Objective Evaluation for SPKCODE_EMBED

Figure 4 presents the mel-cepstral distortions (MCDs) and RMSEs of log F0 for SPKCODE_EMBED models. The horizontal axis indicates the size of the embedded speaker code vector. The results are presented with HMM and DNN_SD.

First, we compared the performance for the model structure of the SPKCODE_EMBED models. For both the conditions using five and 300 target speaker utterances, the experimental results showed the tendency that a larger embedding vector size achieves smaller MCDs. This tendency was common for both the multi-speaker modeling and speaker adaptation tasks and similar to that reported by Luong et al. [13]. As for RMSEs of log F0, on the other hand, there was no consistent relationship between the size of the embedding vector and F0 RMSEs for both the multi-speaker modeling and speaker adaptation tasks. This tendency contrasted with that reported in a previous study [13], in which it was stated that a larger embedding vector size effectively reduces RMSEs of log F0. We consider that this is because of the difference in the number of speakers in the training data. There were 33 speakers in our experiments, while there were 100 speakers in those conducted by Luong et al. [13]). Since the number of speakers was smaller in our experiments, the embedding speaker code vector space became sparse when the vector size was large, which led to over-fitting and diminished performance.

We then compared the performance of the conventional methods, i.e., HMM and DNN_SD. We found the relationship of

- DNN_SD < HMM
  (300 utterances, MCDs)
- HMM < DNN_SD
  (300 utterances, F0 RMSEs)

for both multi-speaker modeling and speaker adaptation tasks. The MCDs of DNN_SD were better than those of HMM because in modeling complex context dependencies DNN-based methods have an advantage over the tree-clustered HMM-based methods as Zen et al. described [2]. The F0 RMSEs of DNN_SD were higher than HMM, which is the same tendency as Zen et al. reported [2].

We then compared the performance of SPKCODE_EMBED with that obtained with these conventional methods. From the results of multi-speaker modeling tasks, we can see the relationship of

- SPKCODE_EMBED < HMM
  (5 utterances, MCDs)
- HMM < SPKCODE_EMBED
  (5 utterances, F0 RMSEs)
- SPKCODE_EMBED < DNN_SD < HMM (300 utterances, MCDs)
- HMM < SPKCODE_EMBED < DNN_SD (300 utterances, F0 RMSEs).

From these results for multi-speaker modeling, we confirm that the SPKCODE_EMBED models can slightly improve the parameter estimation accuracy compared with DNN_SD models by utilizing a multi-speaker speech corpus.

From the results obtained from the speaker adaptation task in Fig. 4, we found the relationship of

- HMM < SPKCODE_EMBED
  (five utterances, MCDs)
- HMM < SPKCODE_EMBED
  (five utterances, F0 RMSEs)
- DNN_SD < HMM < SPKCODE_EMBED (300 utterances, MCDs)
- HMM < DNN_SD < SPKCODE_EMBED (300 utterances, F0 RMSEs).

The performance of SPKCODE_EMBED was comparable to or worse than that obtained with the conventional HMM and DNN_SD. This is because the embedded speaker code model architecture has too small a number of free parameters for adaptation. The number of free parameters for adaptation is equal to the size of the embedded speaker codes in SPKCODE_EMBED (e.g., 2, 10, 33, 128 and 256 in our experiments). This too small a number of free parameters led to poor reproducibility of speaker characteristics. From these results, we confirmed the embedded speaker code-based approach for speaker adaptation task was not effective because it was worse than DNN_SD when using 300 target speaker utterances and also worse than HMM when using five target speaker utterances.

5.2.2 Objective Evaluation for SPKCODE_ONE-HOT

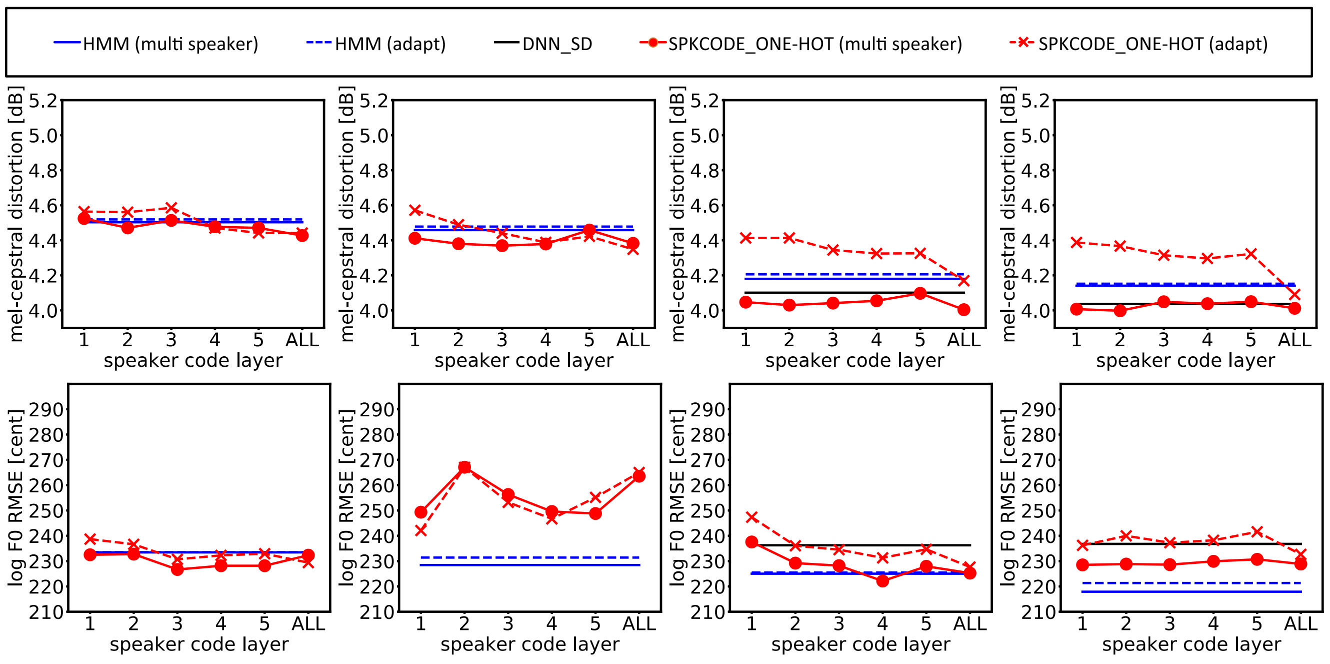
Figure 5 presents the MCDs and F0 RMSEs for SPKCODE_ONE-HOT models. The horizontal axis indicates the speaker code input layer. The results for conventional HMM and DNN_SD are the same as those in Fig. 4.

First, we compared the performance for the model structure of the SPKCODE_ONE-HOT models. For both the multi-speaker modeling and speaker adaptation tasks, we found that the models using the 4th hidden layer gave consistently low MCDs and F0 RMSEs when using five target speaker utterances while the models using all hidden layers gave lower MCDs and F0 RMSEs when using 300 target speakers' utterances. From these results, we determined that the models using the 4th hidden layer were optimal when using five target speaker utterances while the models using all the hidden layers were optimal when using 300 target speaker utterances, for both multi-speaker training and speaker adaptation tasks.

We then compared the performance for the SPKCODE_ ONE-HOT models with that of the conventional methods. From the results for the multi-speaker modeling task, we found the relationship of

- SPKCODE_ONE-HOT < HMM
  (five utterances, MCDs)
- HMM < SPKCODE_ONE-HOT
  (five utterances, F0 RMSEs)
- SPKCODE_ONE-HOT < DNN_SD < HMM (300 utterances, MCDs)
- HMM < SPKCODE_ONE-HOT < DNN_SD (300 utterances, F0 RMSEs)

when the model structure for SPKCODE_ONE-HOT was set at optimal. For each condition, we found that the performance of DNN_ONE-HOT was consistently better than that of DNN_SD models. These results confirmed the prediction performance improvement of SPKCODE_ONE-HOT using a multi-speaker speech corpus. When SPKCODE_ONE-HOT is compared with HMM, it is found that MCDs of SPKCODE_ONE-HOT are improved. The advantage of DNN over HMM in modeling mel-cepstra was also confirmed for SPKCODE_ONE-HOT models. We can also see that F0 RMSEs of SPKCODE_ONE-HOT were comparable to or



(a) five utterances for male target speaker **(b)** five utterances for female target speaker **(c)** 300 utterances for male target speaker **(d)** 300 utterances for female target speaker

Fig. 5 The objective evaluation results for one-hot speaker code models. The horizontal axis indicates the speaker code input layer.

higher than those of HMM. These results revealed that, relative to HMM, there is still room to improve F0 prediction in SPKCODE_ONE-HOT models, although their performance is considerably better than that of conventional DNN_SD models.

For the speaker adaptation results, we can see found the relationship of

- SPKCODE_ONE-HOT < HMM (five utterances, MCDs)
- HMM < SPKCODE_ONE-HOT (five utterances, F0 RMSEs)
- DNN_SD < SPKCODE_ONE-HOT < HMM (300 utterances, MCDs)
- HMM < SPKCODE_ONE-HOT < DNN_SD (300 utterances, F0 RMSEs)

when the model structure for SPKCODE_ONE-HOT was set at optimal. When the performance of SPKCODE_ONE-HOT was compared with that of DNN_SD, it was found that MCDs were degraded while F0 RMSEs were improved. Since the multi-speaker speech corpus for SPKCODE_ONE-HOT model training has a larger variety of context than those for DNN_SD, SPKCODE_ONE-HOT models are especially advantageous in F0 prediction. From the results of MCDs' degradation results we obtained, we were concerned that the adaptation by SPKCODE_ONE-HOT could not represent the precise speaker characteristics by effectively using the large amount of adaptation data. The results revealed that more flexible adaptation models were needed when given a large amount of adaptation data. To solve this problem, one promising approach would be a more elaborate speaker code-based adaptation method for which the number of parameters for adaptation can change in accordance with the amount of adaptation data, in the same way that for the HMM-based adaptation method, CSMAPLR reported by Yamagishi et al. [5]. When we compared SPKCODE_ONE-HOT with HMM, we found the relationship was the same as that for the multi-speaker modeling task; MCDs were improved while F0 RMSEs were degraded.

Finally, we compared the performance of the two proposed models. For multi-speaker modeling, the performance of SPKCODE_ONE-HOT and SPKCODE_EMBED was comparable regardless of the amount of training data when the model structure was set optimal. For speaker adaptation, SPKCODE_ONE-HOT models showed superior performance to that of the SPKCODE_EMBED models regardless of the amount of adaptation data or the objective measure. The difference between the two models was especially large when the amount of the adaptation data was large. This is because the performance of SPKCODE_EMBED models saturated with the amount of adaptation data while SPKCODE_ONE-HOT models showed greater performance for a larger amount of adaptation data. SPKCODE_ONE-HOT models can represent the speaker characteristics in more detail because they have a larger number of parameters for each speaker. The number of parameters for adaptation for these models, whose speaker code vector is connected to all hidden layers, is equal to unit number $\times$ the number of hidden layers. In our experimental condition, for example, it is equal to $1024 \times 5 = 5120$. Since this number is greater than that for SPKCODE_EMBED models (2, 10, 33, 128 or 256 as described in Sect. 5.2.1), SPKCODE_ONE-HOT models show more precise reproducibility for speaker characteristics. These results confirm that reducing adaptation parameters by adopting the SPKCODE_EMBED model was not effective for adapting speech synthesis models to speakers when five utterances were given as adaptation data. From the results of the objective evaluation experiment, we concluded that SPKCODE_ONE-HOT models generally perform better than the SPKCODE_EMBED and the best speaker code input layer for SPKCODE_ONE-HOT models for all the hidden layers.

5.3 Subjective Evaluation

We conducted subjective evaluations with respect to the naturalness and similarity of synthesized speech to confirm the effectiveness of the proposed method. We used four models for these evaluations, DNN_SD, HMM (multi-speaker modeling), SPKCODE_ONE-HOT (multi-speaker modeling) and SPKCODE_ONE-HOT (speaker adaptation). The proposed SPKCODE_EMBED models were eliminated from these experiments because the objective evaluation experiments revealed that the proposed SPKCODE_ONE-HOT models perform better and they are not necessary to reveal the effectiveness of the proposed method. We used the optimal architectures for SPKCODE_ONE-HOT models as discussed in the last section, the model using the 4th hidden layer when using five target speaker utterances and the model using all hidden layers when using 300 target speaker utterances. The number of listeners was 24 for a naturalness test and 22 for a similarity test. We conducted five-point MOS and DMOS tests. The scale for the MOS test was five points for "very natural" and 1 points for "very unnatural". The scale for the DMOS test was five points for "very similar" and 1 point for "very dissimilar".

Figures 6 and 7 show the naturalness and similarity scores obtained in the subjective evaluations with confidence intervals of 95%. We found the relation of HMM < SPKCODE_ONE-HOT for both naturalness and similarity scores. Furthermore, the scores of SPKCODE_ONE-HOT using five target speaker utterances were equivalent to those of DNN_SD using 300 target speaker utterances. We can also see the relation of DNN_SD < SPKCODE_ONE-HOT for both naturalness and similarity when using 300 target speaker utterances. These results confirmed that the proposed method can improve the synthetic speech quality by using a multi-speaker speech corpus in DNN-based speech synthesis. On the other hand, there were no significant differences between HMM and SPKCODE_ONE-HOT when using 300 target speaker utterances.

Although the speaker adaptation scores obtained for SPKCODE_ONE-HOT were slightly worse than those obtained for multi-speaker modeling under each condition, they were higher than those obtained for HMM when using five target speaker utterances and comparable to those ob-

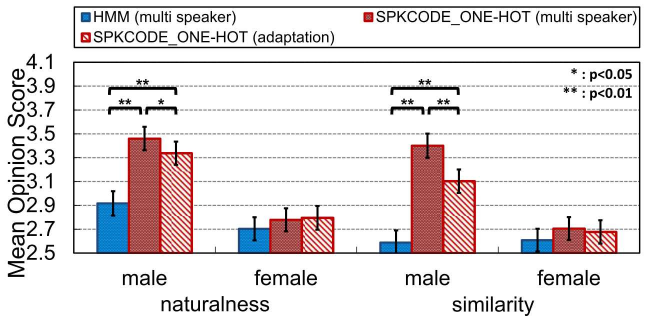


Fig. 6 Naturalness and similarity test results with their 95% confidence interval. (The number of target speaker utterances: 5)

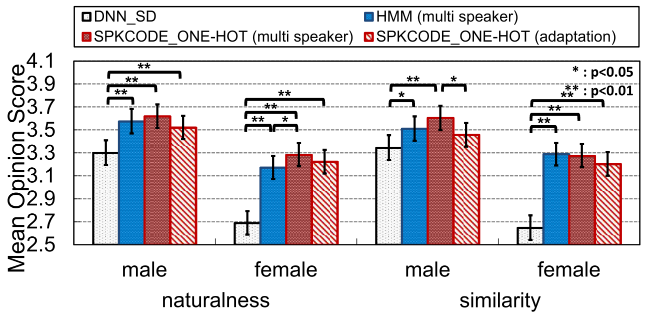


Fig. 7 Naturalness and similarity test results with their 95% confidence interval. (The number of target speaker utterances: 300)

tained for HMM and higher than DNN_SD when using 300 target speaker utterances. These results confirmed that the adaptation based on speaker codes generates speech whose quality is comparable to or higher than that obtained with a conventional speaker dependent DNN, or a speaker adapted HMM.

Although we were able to observe performance improvement with the proposed method for the two target speakers in our experiments, the obtained subjective evaluation results were not always consistent between the two target speakers. Therefore, our future work will include further investigating whether or not the improvements by the proposed method provides can be obtained for any speakers included in the training corpus. For HMM-based cluster adaptive training [7], it is known that the performance depends on the speaker similarity between a target speaker and those in the training corpus. Our future work will also include investigating the relationship between the performance and the speaker similarity between training and target speakers for the proposed DNN-based multi-speaker modeling and speaker adaptation.

6. Comparison with the Conventional Shared Hidden Layers (SHL) Model [8]

6.1 Objective of the Experiments

As a method of DNN-based multi-speaker modeling and speaker adaptation for DNN-based speech synthesis, a shared hidden layer (SHL) [8] model is proposed as well as the proposed speaker code-based method. In this section, we compare the performance of the conventional SHL and the proposed model through objective measurements.

Figure 8 shows the architecture of the conventional shared hidden layer structure. In this model, hidden layers are shared across all the speakers in the training corpus, and can be considered as the global linguistic feature transformation shared by all the speakers. Conversely, all speakers have their own output layers, the so-called regression layer, to model their own specific acoustic spaces. Compared with the speaker dependent DNN, the shared hidden layer model takes the same input linguistic feature, which is converted from text in the same manner, and the same output acoustic feature for each speaker.

For the multi-speaker modeling task performed by the shared hidden layer model, connection weights of the whole DNN are trained using a multi-speaker speech corpus. When synthesizing a speech parameter sequence, a target speaker is chosen from the corpus and the corresponding output layer is selected to generate the speaker's voice.

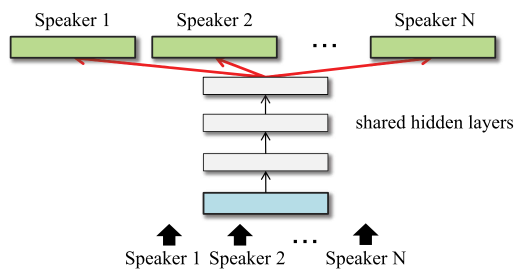
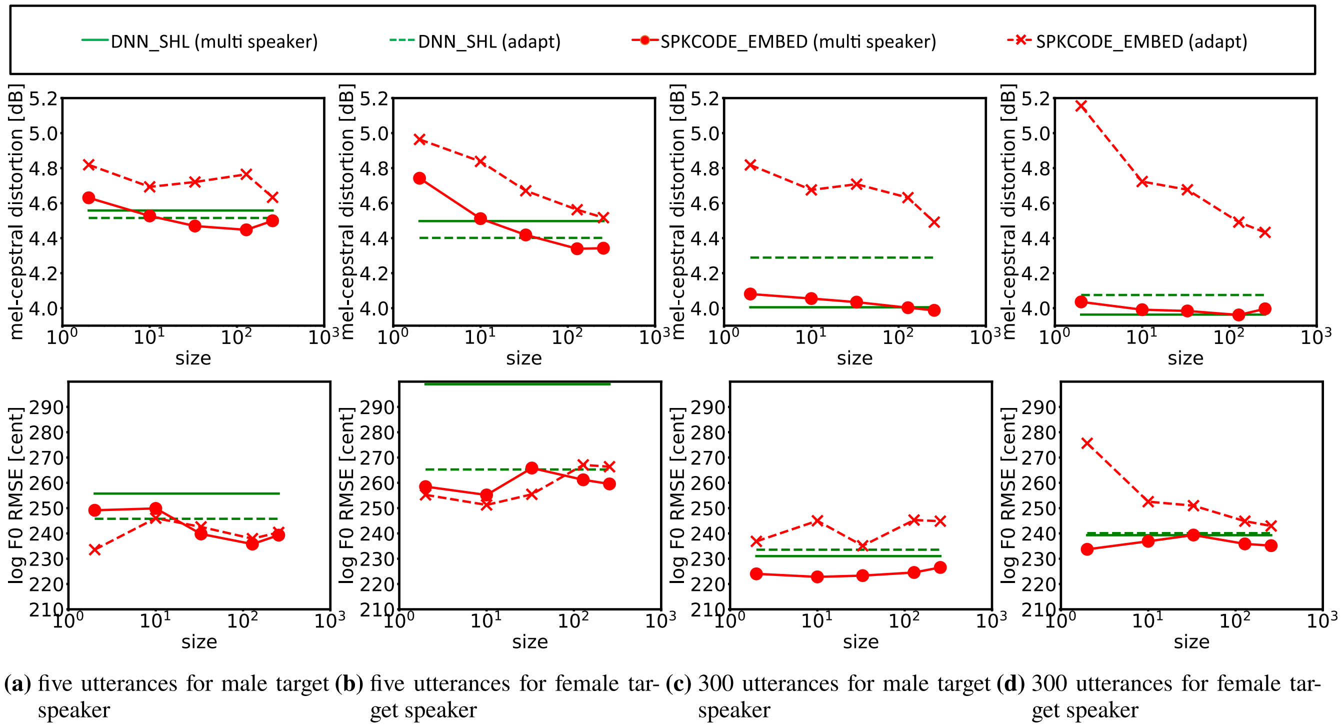


Fig. 8 The model architecture of the conventional shared hidden layer model [8]. The red bold lines indicate the re-estimated parameters for speaker adaptation.

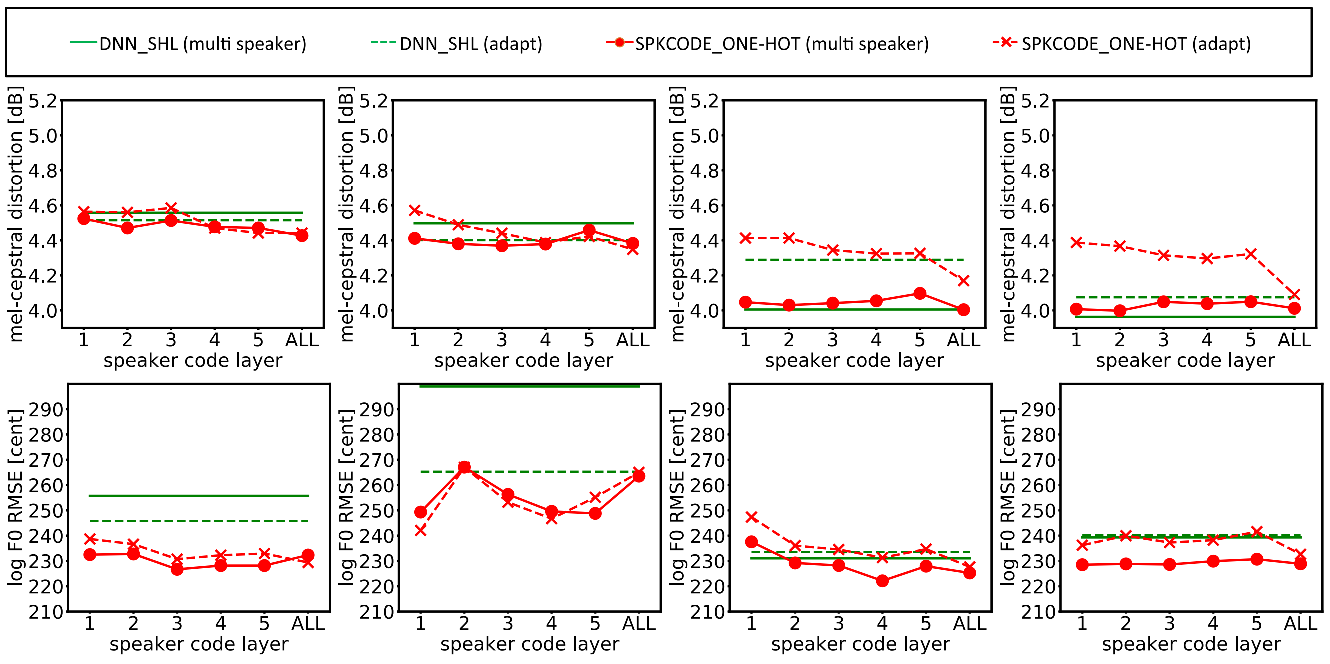
For the speaker adaptation task performed by the shared hidden layer model, first, the model is trained using multi-speakers' speech corpus. Then, the hidden layers transferred from multiple speakers' data are fixed and only the regression layer is updated using the target speaker's speech data. While this re-estimation can be conducted by a back propagation algorithm, it is also possible to use the least squares method to minimize the squared residuals between prediction and ground-truth since there is only a linear regression between the last hidden layer's output and the target [8]. In our experiments, we used a backpropagation algorithm to re-estimate these values. Specifically, we randomly initialized the connection weights to a new output layer and then optimized them to minimize the mean squared error between the output feature of the adaptation data and predicted values, using the Adam-based backpropagation algorithm [18].

The conventional SHL models are indicated by "DNN_SHL" below. For DNN_SHL, the output layers are composed of speaker-dependent nodes while the architecture of the shared hidden layers (the number of hidden layers and their units) was set to the same as that of other DNN models. The other experimental conditions were set in a way similar to those for the SPKCODE_ONE-HOT models.



(a) five utterances for male target speaker **(b)** five utterances for female target speaker **(c)** 300 utterances for male target speaker **(d)** 300 utterances for female target speaker

Fig. 9 The objective evaluation results for embedded speaker code model. The horizontal axis indicates the size (dimension) of the speaker codes.



(a) 5 utterances for male target speaker **(b)** 5 utterances for female target speaker **(c)** 300 utterances for male target speaker **(d)** 300 utterances for female target speaker

Fig. 10 The objective evaluation results for one-hot speaker code models. The horizontal axis indicates the speaker code input layer.

6.2 Experimental Results

Figure 9 and Fig. 10 show the objective evaluation results for SHL. When SPKCODE_EMBED is compared with DNN_SHL in Fig. 9, it is found that SPKCODE_EMBED shows performance comparable or superior or comparable performance to that of DNN_SHL for the multi-speaker modeling task when the size of the speaker code is set large. On the other hand, SPKCODE_EMBED show inferior performance to DNN_SHL for the speaker adaptation task. This is because the DNN_SHL has a larger number of free parameters for adaptation, which is equal to (1024×139 =) 142336. In comparison, SPKCODE_EMBED has free parameters for adaptation of 2, 10, 33, 128 or 256 as described in Sect. 5.2.1. The larger number of free parameters led to more precise reproducibility for speaker characteristics.

When SPKCODE_ONE-HOT is compared with DNN_SHL models in Fig. 5, it is found that MCDs for SPKCODE_ONE-HOT are lower than those for DNN_SHL models when using five target speaker utterances. It is also revealed that MCDs of SPKCODE_ONE-HOT are equivalent to those for DNN_SHL models when using 300 target speakers' utterances. We can also see that F0 RMSEs of SPKCODE_ONE-HOT are consistently lower than those for DNN_SHL. These results confirmed that the proposed SPKCODE_ONE-HOT models are advantageous compared with DNN_SHL in both multi-speaker modeling and speaker adaptation tasks, especially in modeling F0 contour. Since DNN_SHL models utilize the connections to the output layer to model the speaker characteristics, they can represent only the simple characteristics that can be interpreted by a single affine transformation. In contrast, since SPKCODE_ONE-HOT models utilize the augmented input vector to model complex context dependencies, they are advantageous in modeling speaker characteristics more precisely.

7. Conclusion

In this paper, we proposed a DNN-based speech synthesis method using speaker codes to improve speech quality by using a multi-speaker speech corpus. Objective evaluation results showed that the proposed model outperforms the speaker dependent DNNs and multi-speaker DNNs with shared hidden layers for both multi-speaker modeling and speaker adaptation tasks. Subjective evaluation results showed that the proposed model can produce more natural speech than the conventional speaker dependent DNNs and HMM-based speaker adaptation method, especially when using a small number of target speaker utterances.

References

[1] N. Hojo, Y. Ijima, and H. Mizuno, "An investigation of DNN-based speech synthesis using speaker codes," Proc. Intersepech 2016, pp.2278–2282, 2016.

[2] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," Proc. ICASSP 2013, pp.7962–7966, 2013.

[3] H. Zen and A. Senior, "Deep mixture density networks for acoustic

modeling in statistical parametric speech synthesis," Proc. ICASSP 2014, pp.3844–3848, 2014.

[4] Y. Fan, Y. Qian, F. Xie, and F.K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," Proc. Interspeech, pp.1964–1968, 2014.

[5] J. Yamagishi and T. Kobayashi, "Average-voice-based speech synthesis using HSMM-based speaker adaptation and adaptive training," IEICE Transactions on Information and Systems, vol.E90-D, no.2, pp.533–543, 2007.

[6] J. Yamagishi, T. Kobayashi, Y. Nakano, K. Ogata, and J. Isogai, "Analysis of speaker adaptation algorithms for HMM-based speech synthesis and a constrained SMAPLR adaptation algorithm," IEEE Transactions Audio, Speech, and Language Process., vol.17, no.1, pp.66–83, 2009.

[7] V. Wan, J. Latorre, K. Chin, L. Chen, M. Gales, H. Zen, K. Knill, and M. Akamine, "Combining multiple high quality corpora for improving HMM-TTS," Proc. Interspeech 2012, pp.1134–1137, 2012.

[8] Y. Fan, Y. Qian, F.K. Soong, and L. He, "Multi-speaker modeling and speaker adaptation for DNN-based tts synthesis," Proc. ICASSP 2015, pp.4475–4479, IEEE, 2015.

[9] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," Proc. ASRU 2013, pp.55–59, 2013.

[10] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," Proc. ICASSP 2013, pp.7942–7946, 2013.

[11] S. Xue, O. Abdel-Hamid, H. Jiang, L. Dai, and Q. Liu, "Fast adaptation of deep neural network based on discriminant codes for speech recognition," IEEE/ACM Trans. Audio, Speech, Lang. Process., vol.22, no.12, pp.1713–1725, 2014.

[12] Z. Wu, P. Swietojanski, C. Veaux, S. Renals, and S. King, "A study of speaker adaptation for DNN-based speech synthesis," Proc. Interspeech 2015, pp.879–883, 2015.

[13] H.-T. Luong, S. Takaki, G.E. Henter, and J. Yamagishi, "Adapting and controlling DNN-based speech synthesis using input codes," Proc. ICASSP 2017, pp.4905–4909, IEEE, 2017.

[14] B. Potard, P. Motlicek, and D. Imseng, "Preliminary work on speaker adaptation for DNN-based speech synthesis," tech. rep., Idiap, Rep. Idiap-RR-02-2015, 2015.

[15] Y. Fan, Y. Qian, F.K. Soong, and L. He, "Unsupervised speaker adaptation for DNN-based TTS synthesis," Proc. ICASSP 2016, pp.5135–5139, IEEE, 2016.

[16] Y. Fan, Y. Qian, F.K. Soong, and L. He, "Speaker and language factorization in DNN-based TTS synthesis," Proc. ICASSP 2016, pp.5540–5544, IEEE, 2016.

[17] H. Kawahara, I. Masuda-Katsuse, and A. De Cheveigné, "Restructuring speech representations using a pitch-adaptive time–frequency smoothing and an instantaneous-frequency-based f0 extraction: Possible role of a repetitive structure in sounds," Speech communication, vol.27, no.3-4, pp.187–207, 1999.

[18] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.

[19] K. Shinoda and T. Watanabe, "Acoustic modeling based on the MDL criterion for speech recognition," Proc. EUROSPEECH 1997, pp.99–102, 1997.

[20] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for HMM-based speech synthesis," Proc. ICASSP 2000, pp.1315–1318, 2000.

[21] T. Toda and K. Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," IEICE Transactions on Information and Systems, vol.E90-D, no.5, pp.816–824, 2007.

Nobukatsu Hojo recieved the B.E and M.E. degrees from the University of Tokyo, Japan, in 2012 and 2014, respectively. He joined NTT Media Intelligence Laboratories in 2014, where he engaged in the research and development of speech synthesis. He is a member of Acoustical Society of Japan (ASJ), the International Speech Communication Association (ISCA) and the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

Yusuke Ijima received the B.E. degree in electric and electronics engineering from National Institution for Academic Degrees and University Evaluation by graduation from Yatsushiro National College of Technology, Japan, in 2007, and the M.E. and Ph.D. degrees in information processing from Tokyo Institute of Technology, Japan, in 2009 and 2015, respectively. He joined NTT Cyber Space Laboratories (currently Media Intelligence Laboratories) in 2009, where he engaged in the research and development of speech synthesis. He is a member of the Acoustical Society of Japan (ASJ), the International Speech Communication Association (ISCA) and the Institute of Electronics, Information and Communication Engineers (IEICE) of Japan.

Hideyuki Mizuno received the B.E. and M.E. degrees from Nagoya University, Japan, in 1986 and 1988, respectively. He received the Ph.D. degree from University of Tsukuba, Japan, in 2006. From 1988 to 2015, He was with the laboratories of Nippon Telegraph and Telephone Corporation (NTT), Japan, where he engaged in the research and development of speech synthesis. In 2015, he joined the Department of Computer and Media Engineering of Tokyo University of Science, Suwa, Japan, where he is currently a Professor. His current research interests include multimedia processing and analysis. He received the Technical Development Award of the Acoustical Society of Japan in 1998. He is a member of the ASJ and the Institute of Electronics Information and Communication Engineers (IEICE) of Japan.