10th ISCA Speech Synthesis Workshop
20-22 September 2019, Vienna, Austria

DNN-based Speaker Embedding Using Subjective Inter-speaker Similarity for Multi-speaker Modeling in Speech Synthesis

Yuki Saito, Shinnosuke Takamichi, and Hiroshi Saruwatari

Graduate School of Information Science and Technology, The University of Tokyo,
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
{yuuki_saito, shinnosuke_takamichi, hiroshi_saruwatari}@ipc.i.u-tokyo.ac.jp

Abstract

This paper proposes novel algorithms for speaker embedding using subjective inter-speaker similarity based on deep neural networks (DNNs). Although conventional DNN-based speaker embedding such as a d-vector can be applied to multi-speaker modeling in speech synthesis, it does not correlate with the subjective inter-speaker similarity and is not necessarily appropriate speaker representation for open speakers whose speech utterances are not included in the training data. We propose two training algorithms for DNN-based speaker embedding model using an inter-speaker similarity matrix obtained by large-scale subjective scoring. One is based on similarity vector embedding and trains the model to predict a vector of the similarity matrix as speaker representation. The other is based on similarity matrix embedding and trains the model to minimize the squared Frobenius norm between the similarity matrix and the Gram matrix of d-vectors, i.e., the inter-speaker similarity derived from the d-vectors. We crowdsourced the inter-speaker similarity scores of 153 Japanese female speakers, and the experimental results demonstrate that our algorithms learn speaker embedding that is highly correlated with the subjective similarity. We also apply the proposed speaker embedding to multi-speaker modeling in DNN-based speech synthesis and reveal that the proposed similarity vector embedding improves synthetic speech quality for open speakers whose speech utterances are unseen during the training.

Index Terms: speaker embedding, subjective inter-speaker similarity, deep neural network, d-vector, multi-speaker modeling, speech synthesis

1. Introduction

Statistical parametric speech synthesis (SPSS) [1] is a technique for synthesizing naturally sounding and easily controllable synthetic speech. Recent developments of both training algorithms and acoustic modeling for SPSS using deep neural networks (DNNs) [2] have significantly improved quality of synthetic speech. For instance, training algorithms based on generative adversarial networks [3] have significantly improved synthetic speech quality by reducing the statistical differences between natural and generated speech parameters [4, 5, 6]. Acoustic model based on Tacotron [7] and WaveNet vocoder [8, 9] has achieved high fidelity natural speech [10]. Although improving the synthetic speech quality is one of the goals of SPSS research, learning interpretable representation for controlling characteristics of the synthetic speech (i.e., features that are highly correlated with human speech perception) is also important. We focus on learning the speaker representation for multi-speaker modeling in DNN-based SPSS.

The simplest way for representing speaker identity in DNN-based SPSS is to use a speaker code [11] that denotes one speaker by using a one-hot coded vector. Although the speaker code works well for reproducing the characteristics of *closed* speakers whose speech utterances are included in the training corpora, they have difficulties in 1) dealing with *open* speakers, i.e., previously unseen speakers during the training, because their speaker codes are not defined, and 2) finding a desired speaker when the number of the closed speakers is large. Some techniques for adapting the speaker codes have been proposed for alleviating the first problem [11, 12]; however, the characteristics of the open speakers are not fully reproduced. A more effective approach is utilizing DNNs trained to predict speaker identity from given acoustic features. A d-vector [13] is a well-known example of the DNN-based speaker embedding technique and has been applied to multi-speaker modeling in SPSS [14] using variational auto-encoders (VAEs) [15]. However, even the d-vector cannot solve the second problem because it is merely used for verifying a specific speaker, and its coordinates in the embedding space do not correlate with the subjective inter-speaker similarity, i.e., perceptually similar speakers are not necessarily embedded close to each other.

To learn interpretable speaker embedding, we propose novel algorithms incorporating the subjective inter-speaker similarity into training the DNN-based speaker embedding model. First, we conduct large-scale subjective scoring to obtain a matrix representing the subjective inter-speaker similarity. The model is trained to minimize the loss functions defined by the similarity matrix. We investigate two approaches for the training. The first is similarity *vector* embedding, which trains the model to predict a vector of the similarity matrix instead of the conventional speaker code. The second is similarity *matrix* embedding, which trains the model to minimize the squared Frobenius norm between the similarity matrix and Gram matrix of the d-vectors, i.e., inter-speaker similarity derived from the d-vectors. We crowdsourced the inter-speaker similarity scores of 153 Japanese female speakers, and the experimental results demonstrate that the proposed algorithms can learn speaker embedding that is highly correlated with the subjective similarity compared with the conventional d-vectors. We also investigate the effectiveness of the proposed speaker embedding for the VAE-based multi-speaker SPSS [14], and reveal that the similarity vector embedding improves synthetic speech quality for open speakers.

2. Conventional speaker embedding

2.1. One-hot speaker code

A speaker code [11] $\boldsymbol{c} = [c(1), \cdots, c(n), \cdots, c(N_{\mathrm{s}})]^{\top}$ is the 1-of-N_{s} representation for identifying the one of closed N_{s} speakers, which is the *discrete* representation of speaker identity. The speaker code $\boldsymbol{c}_i$ for the ith speaker is defined as fol-

10.21437/SSW.2019-10

lows:

$$c_i(n) = \begin{cases} 1 & \text{if } n = i \\ 0 & \text{otherwise} \end{cases} \quad (1 \leq n \leq N_{\mathrm{s}}). \tag{1}$$

Although the one-hot coded representation works reasonably well, it cannot define the identity of open speakers, and the size of the code increases in proportion to the number of the closed speakers. Moreover, it should be difficult for users to find their desired speaker in synthesizing speech because speaker codes completely ignore the subjective inter-speaker similarity.

2.2. *d*-vector

A d-vector [13] is a bottleneck feature vector extracted from a pre-trained DNN-based speaker recognition model, which is the *continuous* representation of speaker identity. The model is trained to predict speaker identity from a given acoustic feature sequence by minimizing the softmax cross-entropy defined as follows:

$$L_{\mathrm{SCE}}\left(\boldsymbol{c}, \hat{\boldsymbol{c}}\right) = -\sum_{n=1}^{N_{\mathrm{s}}} c\left(n\right) \log \hat{c}\left(n\right), \tag{2}$$

where $\hat{\boldsymbol{c}} = [\hat{c}(1), \cdots, \hat{c}(n), \cdots, \hat{c}(N_{\mathrm{s}})]^\top$ is an output vector of the DNNs. The N_{d}-dimensional d-vector $\boldsymbol{d} = [d(1), \cdots, d(N_{\mathrm{d}})]^\top$ is extracted from a bottleneck layer of the DNNs. The one before the output layer is often used. Typically, N_{d} is smaller than N_{s} and the d-vector enables us to use the lower-dimensional speaker representation. We can define the identity of the ith speaker $\boldsymbol{d}_i$ as a d-vector averaged over all d-vectors generated from acoustic features of the ith speaker. Although we can embed speakers in the continuous space defined by d-vectors and can deal with open speakers [14], it is still difficult for users to interpret what the speaker embedding means because its coordinates do not correlate with the subjective inter-speaker similarity.

3. Proposed speaker embedding

Here, we propose two algorithms for learning speaker embedding that is highly correlated with the subjective inter-speaker similarity.

3.1. Subjective inter-speaker similarity matrix

We define a subjective inter-speaker similarity matrix that represents the speaker-pair similarity perceived by listeners. Let $\mathbf{S} = [\boldsymbol{s}_1, \cdots, \boldsymbol{s}_i, \cdots, \boldsymbol{s}_{N_{\mathrm{s}}}]$ be an N_{s}-by-N_{s} symmetric similarity matrix and $\boldsymbol{s}_i = [s_{i,1}, \cdots, s_{i,j}, \cdots, s_{i,N_{\mathrm{s}}}]^\top$ be an N_{s}-dimensional similarity vector of the ith speaker. Each element $s_{i,j}$ takes a value between $-v$ and v, which represents the perceptual similarity of the ith and jth speakers. Namely, $s_{i,j}$ stores the average score of the subjective evaluation asking "To what degree do the ith speaker's voice and the jth speaker's one sound similar?" We assume that the diagonal elements, i.e., intra-speaker similarity, take the maximum value of the similarity. Figures 1(a) and (b) show the similarity matrix of 153 Japanese female speakers and its sub-matrix, respectively. Section 4.1.1 describes details of the subjective scoring for obtaining the score matrix, and Section 4.2 presents analysis of the scores.

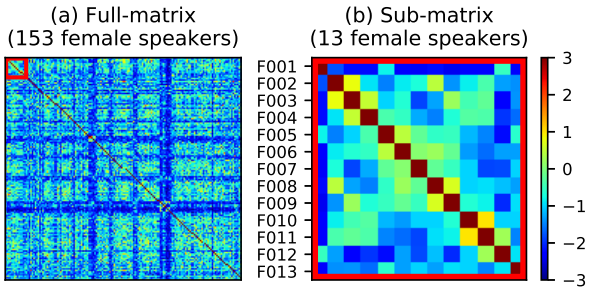

Figure 1: *(a) Similarity matrix of 153 Japanese female speakers and (b) its sub-matrix obtained by large-scale subjective scoring.*

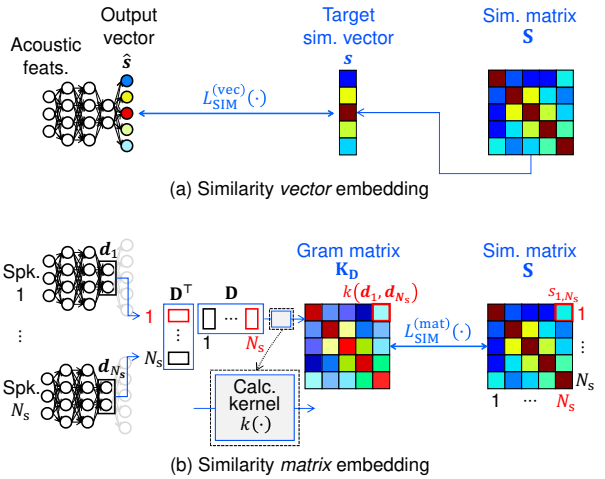

Figure 2: *Calculation of loss functions in proposed algorithms based on (a) similarity vector embedding and (b) similarity matrix embedding.*

3.2. Training based on similarity vector embedding

The first proposed algorithm uses the similarity vector as a target to be predicted by the DNNs, instead of the conventional speaker code. The loss function for the training is defined as follows:

$$L_{\mathrm{SIM}}^{(\mathrm{vec})}\left(\boldsymbol{s}, \hat{\boldsymbol{s}}\right) = \frac{1}{N_{\mathrm{s}}} \left(\hat{\boldsymbol{s}} - \boldsymbol{s}\right)^\top \left(\hat{\boldsymbol{s}} - \boldsymbol{s}\right), \tag{3}$$

where $\boldsymbol{s} \in \mathbf{S}$ and $\hat{\boldsymbol{s}}$ denote the target similarity vector and output vector of the DNNs, respectively. Figure 2(a) shows the computation procedure of $L_{\mathrm{SIM}}^{(\mathrm{vec})}(\cdot)$.

3.3. Training based on similarity matrix embedding

The second proposed algorithm directly uses the similarity matrix $\mathbf{S}$ as a constraint on coordinates of d-vectors. Let $\mathbf{D} = [\boldsymbol{d}_1, \cdots, \boldsymbol{d}_i, \cdots, \boldsymbol{d}_{N_{\mathrm{s}}}]$ be an N_{s}-by-N_{s} matrix including d-vectors extracted from all closed speakers. The loss function for the training is defined as follows:

$$L_{\mathrm{SIM}}^{(\mathrm{mat})}\left(\mathbf{D}, \mathbf{S}\right) = \frac{2}{\left\|\mathbf{1}_{N_{\mathrm{s}}} - \mathbf{I}_{N_{\mathrm{s}}}\right\|_F^2} \left\|\widetilde{\mathbf{K}}_{\mathbf{D}} - \widetilde{\mathbf{S}}\right\|_F^2, \tag{4}$$

$$\widetilde{\mathbf{K}}_{\mathbf{D}} = \mathbf{K}_{\mathbf{D}} - \left(\mathbf{K}_{\mathbf{D}} \odot \mathbf{I}_{N_{\mathrm{s}}}\right), \tag{5}$$

$$\widetilde{\mathbf{S}} = \mathbf{S} - s\mathbf{I}_{N_{\mathrm{s}}}, \tag{6}$$

where $\|\cdot\|_F^2$, $\odot$, $\mathbf{1}_{N_{\mathrm{s}}}$, and $\mathbf{I}_{N_{\mathrm{s}}}$ denote the squared Frobenius norm of the given matrix, Hadamard product, N_{s}-by-N_{s} ma-

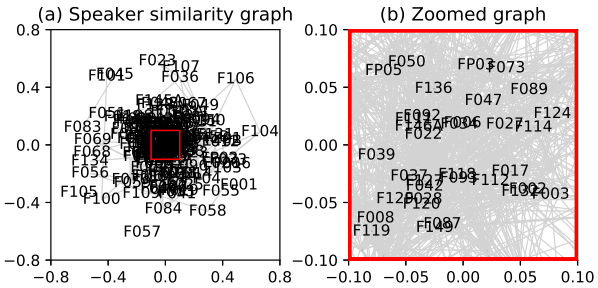

Figure 3: *(a) Speaker similarity graph defined by similarity matrix shown in Fig. 1(a) and (b) its zoomed version.*

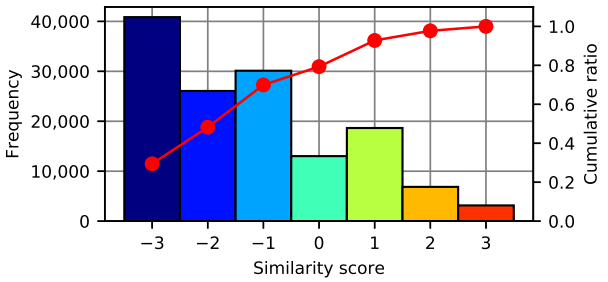

Figure 4: *Histogram of similarity scores with its cumulative ratio denoted by red line.*

trix whose all components are 1, and N_s-by-N_s identity matrix, respectively. $2/\|\mathbf{1}_{N_\mathrm{s}} - \mathbf{I}_{N_\mathrm{s}}\|_F^2$ is a normalization coefficient corresponding to the degrees of freedom of the matrix $\widetilde{\mathbf{K}}_\mathbf{D} - \widetilde{\mathbf{S}}$. $\mathbf{K}_\mathbf{D}$ is the Gram matrix of d-vectors defined as:

$$\mathbf{K}_\mathbf{D} = \begin{bmatrix} k(\boldsymbol{d}_1, \boldsymbol{d}_1) & \cdots & k(\boldsymbol{d}_1, \boldsymbol{d}_{N_\mathrm{s}}) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{d}_{N_\mathrm{s}}, \boldsymbol{d}_1) & \cdots & k(\boldsymbol{d}_{N_\mathrm{s}} \boldsymbol{d}_{N_\mathrm{s}}) \end{bmatrix}, \quad (7)$$

where $k(\boldsymbol{d}_i, \boldsymbol{d}_j)$ is a kernel function calculated by using $\boldsymbol{d}_i$ and $\boldsymbol{d}_j$, i.e., the speaker similarity derived from the d-vectors. This proposed algorithm can directly learn speaker embeddings correlated with the subjective inter-speaker similarity. Figure 2(b) shows the computation procedure of $L_\mathrm{SIM}^{(\mathrm{mat})}(\cdot)$.

The minimization of Eq. (4) means the speaker pairs are embedded with the consideration of both their perceptual similarity and dissimilarity. For the actual use of controllable DNN-based SPSS, at least the consideration of the similar speaker pairs should be satisfied. Therefore, we can relax Eq. (4) to satisfy this as follows:

$$L_\mathrm{SIM}^{(\mathrm{mat-re})}(\mathbf{D}, \mathbf{S}) = \frac{2}{\|\mathbf{W} - \mathbf{I}_\mathrm{s}\|_F^2} \left\| \mathbf{W} \odot \left(\widetilde{\mathbf{K}}_\mathbf{D} - \widetilde{\mathbf{S}} \right) \right\|_F^2, \quad (8)$$

where $\mathbf{W} = [w_{i,j}]_{1 \le i,j \le N_\mathrm{s}}$ is defined as $w_{i,j} = 1$ if $s_{i,j} > 0$ otherwise 0 and $2/\|\mathbf{W} - \mathbf{I}_\mathrm{s}\|_F^2$ is a normalization coefficient corresponding to the degrees of freedom of the matrix $\mathbf{W}$. In this formulation, perceptually dissimilar speaker pairs are not considered in training.

3.4. Discussion

The similarity matrix used in the proposed algorithms offers better understanding of the relationships among speakers by visualizing them as a graph defined by the matrix. Figures 3(a) and (b) show the speaker similarity graph defined by the similarity matrix shown in Fig. 1(a) and its zoomed version, respectively. The adjacency matrix of the graph was the same as the matrix $\mathbf{W}$ of Eq. (8). The positions of the speakers in Fig. 3 were determined by using multidimensional scaling with the similarity matrix. The minimization of Eq. (8) is similar to learning the graph from acoustic features. Therefore, we expect to further introduce graph signal processing [16] and graph embedding [17] to DNN-based speaker embedding and multi-speaker modeling in SPSS.

Regarding prior works, Tachibana et al. [18] and Ohta et al. [19] proposed controllable SPSS in the hidden Markov model (HMM) and Gaussian mixture model (GMM) era. They modeled the characteristics of a specific speaker with subjective impressions such as "warm – cold" and "clear – hoarse" so that the latent variables of the HMMs or GMMs were related to the word pairs. Our algorithms extend these ideas to make the DNNs model the *pair-wise* speaker similarity as the embedding vectors rather than the conventional *point-wise* impressions of one speaker. Furthermore, the relationship between the speakers' intention and the listeners' perception (e.g., the differences in emotion perception [20]) can be modeled by our algorithms.

The proposed algorithm based on similarity matrix embedding can directly learn inter-speaker relationships by making the Gram matrix of d-vectors close to the similarity matrix. We can choose an arbitrary kernel function to construct the embedding space. When the inner product is used as the kernel function, Eq. (4) is equivalent to deep clustering [21] (except for subtracting the diagonal components). Not only such a simple kernel but also a more complicated one can be utilized in our method.

4. Experimental Evaluation

4.1. Experimental conditions

4.1.1. Conditions for large-scale subjective scoring

We conducted large-scale subjective scoring for obtaining the similarity matrix $\mathbf{S}$ by using our crowdsourced evaluation systems. We used 153 Japanese female speakers included in the JNAS corpus [22]. Each speaker utters at least 150 reading-style utterances (totally about 44 hours). We extracted five non-parallel utterances per speaker for scoring the text-independent inter-speaker similarity of the 153 speakers. Each listener scored the similarity of 34 randomly selected speaker pairs extracted from all of the possible 11,628 different speaker pairs. The score was an integer between -3 (completely different) and $+3$ (very similar). The similarity of one of the 11,628 speaker pairs was scored by at least 10 different listeners. Finally, 4,060 listeners participated in the scoring, and 138,040 answers were obtained.

4.1.2. Conditions for DNN-based speaker embedding

The JNAS corpus was also used for training DNN-based speaker embedding model. Ninety percent of the utterances and the remainder were used for training and evaluation, respectively. The five utterances used for the subjective scoring were omitted from both the training and evaluation data. The number of utterances per speaker was balanced among the speakers. The number of closed speakers used for training, N_s, was set to 140, except for 13 speakers shown in Fig. 1(b). The 13 open speakers were used for objective and subjective evaluations in Sections 4.3 and 4.4. Although each element in the similarity matrix was ranged in $[-3, +3]$, we normalized the values to be in $[-1, +1]$ during the training. Accordingly, the sigmoid kernel $k(\boldsymbol{d}_i, \boldsymbol{d}_j) = \tanh(\boldsymbol{d}_i^\top \boldsymbol{d}_j)$ was used for the proposed

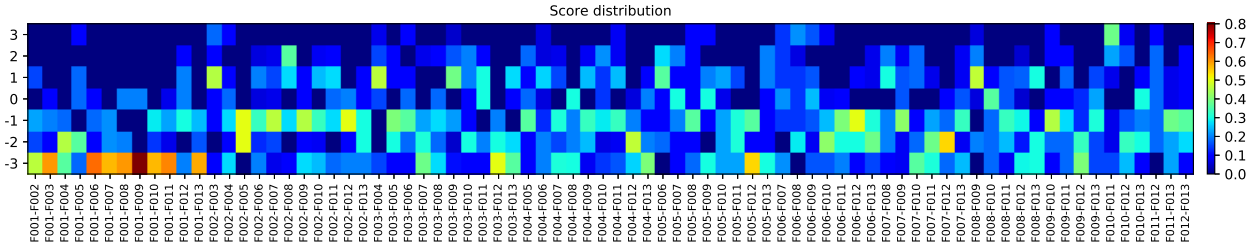

Figure 5: *Histogram of speaker-pair-wise similarity scores. The speaker pair includes the 13 speakers shown in Fig. 1(b).*

similarity matrix embedding described in Section 3.3. The d-vector of a specific speaker was estimated as the average value of d-vectors extracted from acoustic features in the voiced region. The voiced/unvoiced decision was obtained from an F_0 sequence extracted by STRAIGHT vocoder systems [23].

The DNN architecture for speaker embedding model was Feed-Forward networks that included 4 hidden layers with the tanh activation function. The number of hidden units at the 1st-through-3rd layers and the 4th layer for extracting d-vectors were 256 and 8, respectively. The softmax activation function was used for the output layer of the embedding model in the algorithms described in Sections 2.2 and 3.3. Since the values of the similarity matrix $\mathbf{S}$ were normalized in $[-1, +1]$, the tanh activation function was applied to the output layer of the embedding model in the proposed similarity vector embedding described in 3.2. The input of the embedding model was the joint vectors of the 1st-through-39th mel-cepstral coefficients and their dynamic features, and they were normalized to have zero-mean unit-variance during the training. The mel-cepstral coefficients were extracted by using STRAIGHT vocoder systems [23]. AdaGrad [24] was used as the optimization algorithm, setting its learning rate to 0.01. All training algorithms described in Sections 2.2, 3.2, and 3.3 were performed with 100 epochs.

4.1.3. Conditions for VAE-based SPSS

We constructed the VAE-based SPSS [14] that incorporated DNN-based speech recognition and speaker embedding models into speech synthesis for achieving high-quality multi-speaker modeling. The DNN architecture for the recognition model was Feed-Forward networks that included 4 hidden layers with the tanh activation function. The number of hidden units was 1,024. The recognition model was trained to output 43-dimensional Japanese phonetic posteriorgrams (PPGs) [25] from the same input vector as the speaker embedding model, i.e., the 78-dimensional acoustic feature vector including static-dynamic mel-cepstral coefficients. The recognition model training was performed with 100 epochs. About 50 utterances per one in the 140 closed speakers were used for the recognition model training. The embedding model was the same as the DNNs described in Section 4.1.2. The DNN architecture for the VAEs was Feed-Forward networks that consisted of encoder and decoder networks. The encoder had two hidden layers with the rectified linear unit (ReLU) [26] activation function and extracted the 64-dimensional latent variables from a joint vector of the static-dynamic mel-cepstral coefficients and PPGs. The first and second hidden layers had 256 and 128 hidden units, respectively. The decoder reconstructed the input static-dynamic mel-cepstral coefficients from a joint vector of the latent variables, PPGs, and the 8-dimensional speaker embedding vector. The DNN architecture for the decoder was symmetric about that

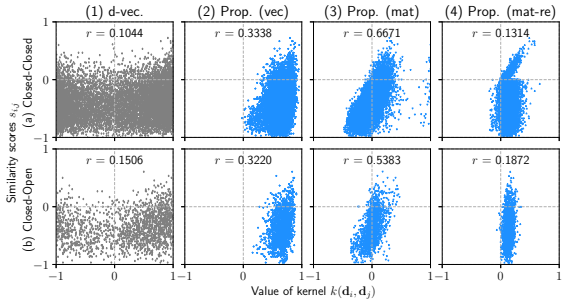

Figure 6: *Scatter plots of similarity scores $s_{i,j}$ and values of kernel $k(\boldsymbol{d}_i, \boldsymbol{d}_j)$ with their correlation coefficient r. These plots were made by all speaker pairs.*

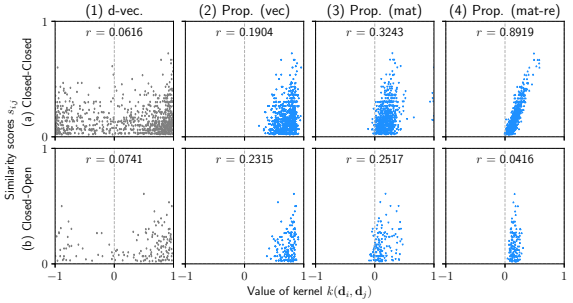

Figure 7: *Scatter plots of similarity scores $s_{i,j}$ and values of kernel $k(\boldsymbol{d}_i, \boldsymbol{d}_j)$ with their correlation coefficient r. These plots were made by speaker pairs whose similarity scores were greater than 0.*

for the encoder. The VAEs were trained to maximize the variational lower bound of the log likelihood [15] with 25 epochs using the same training data as in the embedding model training. The maximum likelihood parameter generation [27] was performed to generate static mel-cepstral coefficients considering their temporal dependencies. The generated mel-cepstral coefficients, input F0, and 5 band-aperiodicity [28, 29] were used for synthesizing speech waveform based on the STRAIGHT vocoder systems [23].

4.2. Analysis of crowdsourced similarity scores

We analyzed the crowdsourced similarity scores that made the similarity matrix shown in Fig. 1(a). The histogram of the all crowdsourced scores is shown in Fig. 4. From this figure, we found that about 70% of the scores were smaller than zero. Also, Fig. 5 plots a histogram of speaker-pair-wise scores. From this figure, we observed that most of the listeners scored -3 for more dissimilar speaker pairs (e.g., "F001-F009"). On the other hand, listeners scored more various values for more similar speaker pairs (e.g., "F010-F011"). These results suggested that listeners could easily find the dissimilar speakers rather

Table 1: *Preference scores on naturalness (left: conventional d-vector, right: proposed methods)*

Speaker	Prop. (vec)	Prop. (mat)	Prop. (mat-re)
F001	0.408 - **0.592**	0.456 - **0.544**	0.448 - **0.552**
F002	0.456 - **0.544**	0.456 - **0.544**	0.504 - 0.496
F003	0.416 - **0.584**	0.444 - **0.556**	0.448 - **0.552**
F004	0.452 - **0.548**	0.460 - 0.540	0.432 - **0.568**
F005	0.380 - **0.620**	0.484 - 0.516	0.484 - 0.516
F006	0.400 - **0.600**	0.452 - **0.548**	0.424 - **0.576**
F007	0.424 - **0.576**	0.484 - 0.516	0.492 - 0.508
F008	0.436 - **0.564**	0.384 - **0.616**	0.436 - **0.564**
F009	0.428 - **0.572**	0.492 - 0.508	0.460 - 0.540
F010	0.436 - **0.564**	0.464 - 0.536	0.452 - **0.548**
F011	0.460 - 0.540	0.428 - **0.572**	0.452 - **0.548**
F012	0.436 - **0.564**	0.460 - 0.540	0.524 - 0.476
F013	0.428 - **0.572**	0.436 - **0.564**	0.412 - **0.588**
Avg.	0.428 - **0.572**	0.454 - **0.546**	0.459 - 0.541

than similar ones.

4.3. Objective evaluation of speaker embedding

To investigate the correlation between the subjective similarity score and speaker embedding, we computed the Pearson correlation coefficient between the similarity scores $s_{i,j}$ and the values of the kernel function $k(\boldsymbol{d}_i, \boldsymbol{d}_j)$. We compared the following four algorithms:

d-vec. : Eq. (2)

Prop. (vec) : Eq. (3)

Prop. (mat) : Eq. (4)

Prop. (mat-re) : Eq. (8)

The results with their scatter plots are shown in Fig. 6. We found that the conventional d-vectors had a weak correlation with the similarity scores. Meanwhile, the three proposed algorithms trained the speaker embedding models so that the embedding vectors had strong correlations with the inter-speaker similarity, which demonstrated that the algorithms could learn speaker embedding that is highly correlated with the subjective scores compared with the conventional d-vectors. Focusing on the speaker pairs whose similarity scores were greater than 0 (shown in Fig. 7), "Prop. (mat-re)" scored the strongest correlations between the similarity scores and speaker embeddings of the closed speakers among the four algorithms. However, it did not work well in the case of the "Closed-Open" speaker pairs. One of the causes might be the data sparsity problem, since the number of the similar speaker pairs was significantly smaller than that of the dissimilar speaker pairs, as shown in Fig. 4.

4.4. Subjective evaluation of VAE-based SPSS

To investigate the effectiveness of the proposed speaker embedding in the VAE-based SPSS, we conducted subjective evaluations on the naturalness and speaker similarity of the synthetic speech of the 13 open speakers. We generated speech samples using mel-cepstral coefficients predicted by the VAEs trained with the 4 different speaker embedding models. Fifty utterances of the speakers were used for estimating the speaker embedding fed into the decoder of the VAE and evaluating the synthetic speech quality. We conducted a series of preference tests (AB tests) on the naturalness of the synthetic speech that compared the conventional algorithm with the three proposed algorithms. Twenty-five listeners participated in each of the following evaluations by using our crowd-sourced evaluation systems, and each listener evaluated 10 speech samples randomly extracted from the 50 utterances. Similarly, we conducted a series of

Table 2: *Preference scores on speaker similarity (left: conventional d-vector, right: proposed methods)*

Speaker	Prop. (vec)	Prop. (mat)	Prop. (mat-re)
F001	0.436 - **0.564**	0.488 - 0.512	0.528 - 0.472
F002	0.468 - 0.532	0.496 - 0.504	0.488 - 0.512
F003	0.432 - **0.568**	0.504 - 0.496	**0.604** - 0.396
F004	0.380 - **0.620**	0.404 - **0.596**	0.488 - 0.512
F005	0.428 - **0.572**	**0.616** - 0.384	**0.596** - 0.404
F006	0.428 - **0.572**	0.444 - **0.556**	0.464 - 0.536
F007	0.492 - 0.508	**0.568** - 0.432	**0.548** - 0.452
F008	0.424 - **0.576**	0.500 - 0.500	0.504 - 0.496
F009	0.400 - **0.600**	0.500 - 0.500	0.448 - **0.552**
F010	0.432 - **0.568**	0.404 - **0.596**	0.496 - 0.504
F011	0.348 - **0.652**	0.444 - **0.556**	0.536 - 0.464
F012	0.492 - 0.508	**0.544** - 0.456	**0.564** - 0.436
F013	0.372 - **0.628**	**0.564** - 0.436	0.452 - **0.548**
Avg.	0.426 - **0.574**	0.498 - 0.502	0.517 - 0.483

XAB tests on the speaker similarity of the synthetic speech using the natural speech of the speaker as the reference speech samples. The total number of task sets was 2 (AB or XAB) $\times$ 3 (proposed embedding algorithms) $\times$ 13 (open speakers) $\times$ 25 (listeners per one task) $=$ 1,950.

The preference scores on the naturalness and speaker similarity are shown in Tables 1 and 2, respectively. The bold values denote that there is a significant difference between the two scores ($p < 0.05$). The row "Avg." means the scores averaged over all speakers. From the results, we found that "Prop. (vec)" always improved both the naturalness and speaker similarity of the synthetic speech, which indicated that the proposed speaker embedding considering the subjective inter-speaker similarity was effective for multi-speaker modeling in DNN-based SPSS. We observed that "Prop. (mat)" also improved the naturalness; however, it significantly degraded the speaker similarity in some cases (e.g., "F005" and "F012"). Similar tendencies were observed in the scores of "Prop. (mat-re)." To investigate the reason, we calculated the degree of a vertex of the speaker similarity graph shown in Fig. 3(a), i.e., the number of similar speakers of each speaker, and found that those of "F005" and "F012" were 7 and 1, respectively. Therefore, we inferred that the proposed similarity matrix embedding might not work well when the number of similar speakers was small. Also, the lower generalization towards the open speakers of "Prop. (mat-re)" might cause the degradation of the speaker similarity, as shown in Fig. 7(b)(4).

5. Conclusion

This paper proposed novel algorithms for incorporating subjective inter-speaker similarity perceived by listeners into the training a speaker embedding model based on deep neural networks (DNNs). The algorithms used an inter-speaker similarity matrix obtained from large-scale subjective scoring as a constraint on training the model. Two approaches for the training were investigated. One is similarity *vector* embedding, which trains the model to predict a vector of the similarity matrix. The other is similarity *matrix* embedding, which trains the model to minimize the squared Frobenius norm between the similarity matrix and Gram matrix of speaker embeddings. For obtaining the similarity matrix, we conducted large-scale subjective scoring in terms of inter-speaker similarity. The experimental results of the DNN-based speaker embedding using the scores demonstrated that the proposed algorithms learned speaker embedding that is highly correlated with the subjective similarity compared with the conventional d-vectors. We also investigated the effectiveness of our speaker embedding for multi-speaker modeling

in DNN-based speech synthesis, and found that the proposed similarity vector embedding improved naturalness and speaker similarity of the synthetic speech. In the future, we will investigate the effects of the kernel function and the parameterization of the similarity score (e.g., using the interval $[0, 1]$, where 1 means "similar" while 0 means "dissimilar") in the proposed similarity matrix embedding, and improve the speaker similarity of the algorithm by introducing more sophisticated techniques of graph signal processing such as graph convolutional networks [30] to the training.

6. Acknowledgements

Part of this work was supported by the SECOM Science and Technology Foundation, JSPS KAKENHI Grant Number 18J22090 and 17H06101, the Ministry of Internal Affairs and Communications, and the GAP foundation program of the University of Tokyo.

7. References

[1] H. Zen, K. Tokuda, and A. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, Nov. 2009.

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

[3] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. NIPS*, Montreal, Canada, Dec. 2014, pp. 2672–2680.

[4] Y. Saito, S. Takamichi, and H. Saruwatari, "Statistical parametric speech synthesis incorporating generative adversarial networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 1, pp. 84–96, Jan. 2018.

[5] Y. Saito, S. Takamichi, and H. Saruwatari, "Text-to-speech synthesis using STFT spectra based on low-/multi-resolution generative adversarial networks," in *Proc. ICASSP*, Alberta, Canada, Apr. 2018, pp. 5299–5303.

[6] Y. Zhao, S. Takaki, H.-T. Luong, J. Yamagishi, D. Saito, and N. Minematsu, "Wasserstein GAN and waveform loss-based acoustic model training for multi-speaker text-to-speech synthesis systems using a WaveNet vocoder," *IEEE Access*, vol. 6, pp. 60478–60488, Sep. 2018.

[7] Y. Wang, RJ Skerry-Ryan, D. Stanton, Y. Wu, R.-J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R.-A. Saurous, "Tacotron: Towards end-to-end speech synthesis," *arXiv*, vol. abs/1703.10135, 2017.

[8] A. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv*, vol. abs/1609.03499, 2016.

[9] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, "Speaker-dependent WaveNet vocoder," in *Proc. INTERSPEECH*, Stockholm, Sweden, Aug. 2017, pp. 1118–1122.

[10] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, RJ Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning WaveNet on mel spectrogram predictions," in *Proc. ICASSP*, Calgary, Canada, Apr. 2018, pp. 4779–4783.

[11] N. Hojo, Y. Ijima, and H. Mizuno, "DNN-based speech synthesis using speaker codes," *IEICE Transactions on Information and Systems*, vol. E101-D, no. 2, pp. 462–472, Feb. 2018.

[12] H.-T. Luong, S. Takaki, G. E. Henter, and J. Yamagishi, "Adapting and controlling DNN-based speech synthesis using input codes," in *Proc. ICASSP*, New Orleans, U.S.A., May 2017, pp. 1905–1909.

[13] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. ICASSP*, Florence, Italy, May 2014, pp. 4080–4084.

[14] Y. Saito, Y. Ijima, K. Nishida, and S. Takamichi, "Non-parallel voice conversion using variational autoencoders conditioned by phonetic posteriorgrams and d-vectors," in *Proc. ICASSP*, Alberta, Canada, Apr. 2018, pp. 5274–5278.

[15] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *arXiv*, vol. abs/1312.6114, 2013.

[16] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[17] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowledge-Based Systems*, vol. 151, pp. 78–94, Jul. 2018.

[18] M. Tachibana, T. Nose, J. Yamagishi, and T. Kobayashi, "A technique for controlling voice quality of synthetic speech using multiple regression HSMM," in *Proc. ICSLP*, Pittsburgh, U.S.A., Sep. 2006, pp. 2438–2441.

[19] K. Ohta, Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, "Regression approaches to voice quality control based on one-to-many eigenvoice conversion," in *Proc. ICSLP*, Bonn, Germany, Aug. 2007, pp. 101–106.

[20] J. Lorenzo-Trueba, G. E. Henter, S. Takaki, J. Yamagishi, Y. Morino, and Y. Ochiai, "Investigating different representations for modeling and controlling multiple emotions in DNN-based speech synthesis," *Speech Communication*, vol. 99, pp. 135–143, May 2018.

[21] J. R. Hershey, Z. Chen, J. L. Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. ICASSP*, Shanghai, China, Mar. 2016, pp. 31–35.

[22] K. Itou, M. Yamamoto, K.Takeda, T. Takezawa, T. Matsuoka, T. Kobayashi, K. Shikano, and S. Itahashi, "JNAS: Japanese speech corpus for large vocabulary continuous speech recognition research," *Journal of the Acoustical Society of Japan (E)*, vol. 20, no. 3, pp. 199–206, May 1999.

[23] H. Kawahara, I. Masuda-Katsuse, and A. D. Cheveigne, "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, Apr. 1999.

[24] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, Jul. 2011.

[25] L. Sun, K. Li, H. Wang, S. Kang, and H. Meng, "Phonetic posteriorgrams for many-to-one voice conversion without parallel data training," in *Proc. ICME*, Seattle, U.S.A., Jul. 2016.

[26] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. AISTATS*, Lauderdale, U.S.A., Apr. 2011, pp. 315–323.

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

[28] H. Kawahara, Jo Estill, and O. Fujimura, "Aperiodicity extraction and control using mixed mode excitation and group delay manipulation for a high quality speech analysis, modification and synthesis system STRAIGHT," in *MAVEBA 2001*, Florence, Italy, Sep. 2001, pp. 1–6.

[29] Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, "Maximum likelihood voice conversion based on GMM with STRAIGHT mixed excitation," in *Proc. INTERSPEECH*, Pittsburgh, U.S.A., Sep. 2006, pp. 2266–2269.

[30] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv*, vol. abs/1609.02907, 2016.