INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Sequence-to-Sequence Voice Conversion with Similarity Metric Learned Using Generative Adversarial Networks

Takuhiro Kaneko, Hirokazu Kameoka, Kaoru Hiramatsu, Kunio Kashino

NTT Communication Science Laboratories, NTT Corporation, Japan

{kaneko.takuhiro,kameoka.hirokazu,hiramatsu.kaoru,kashino.kunio}@lab.ntt.co.jp

Abstract

We propose a training framework for sequence-to-sequence voice conversion (SVC). A well-known problem regarding a conventional VC framework is that acoustic-feature sequences generated from a converter tend to be over-smoothed, resulting in buzzy-sounding speech. This is because a particular form of similarity metric or distribution for parameter training of the acoustic model is assumed so that the generated feature sequence that averagely fits the training target example is considered optimal. This over-smoothing occurs as long as a manually constructed similarity metric is used. To overcome this limitation, our proposed SVC framework uses a similarity metric implicitly derived from a generative adversarial network, enabling the measurement of the distance in the high-level abstract space. This would enable the model to mitigate the over-smoothing problem caused in the low-level data space. Furthermore, we use convolutional neural networks to model the long-range context-dependencies. This also enables the similarity metric to have a shift-invariant property; thus, making the model robust against misalignment errors involved in the parallel data. We tested our framework on a non-native-to-native VC task. The experimental results revealed that the use of the proposed framework had a certain effect in improving naturalness, clarity, and speaker individuality.

Index Terms: voice conversion, deep neural network, generative adversarial network, similarity metric learning

1. Introduction

In this paper, we address the problem of converting the voice characteristics of speech including speaker identity and pronunciation quality. We are particularly interested in developing a real-time (or low-latency) pronunciation-conversion system, which converts non-native speech into native-like intelligible speech in an online manner. The problem of converting non/para-linguistic information of speech while preserving linguistic information is called voice conversion (VC). Since pronunciation quality is part of a speaker's identity in a wide sense, the pronunciation-conversion problem can be seen as a special class of the VC problem.

One successful VC framework involves statistical methods based on Gaussian mixture models (GMMs) [1, 2, 3]. This framework typically consists of collecting parallel data, namely a pair of time-aligned feature sequences of source and target speech and training a mapping function between the acoustic features of the source and target speech using the parallel data. At test time, the feature sequence of input speech is converted using the trained mapping function. This GMM-based framework uses a GMM to represent the conditional distribution of the target feature given the source feature. Recently, a neural network (NN)-based framework based on restricted Boltzmann machines [4] and feed-forward deep NNs [5, 6] and an exemplar-based framework based on non-negative matrix factorization (NMF) [7, 8] have also been proposed with notable success.

A well-known problem regarding the conventional VC frameworks is that acoustic-feature sequences generated from a converter tend to be over-smoothed, resulting in buzzy-sounding speech. This is caused by a side effect of assuming a particular form of similarity metric (e.g., mean squared error (MSE)) or distribution (e.g., Gaussian) for parameter training of the acoustic model so that the generated feature sequence that averagely fits the training target example is considered optimal. This over-smoothing occurs as long as a manually constructed similarity metric is used. Postfiltering methods based on the global variance [9, 10] or modulation spectrum [11] have been proposed with the aim of reconstructing the original spectro-temporal fine texture. However, these methods rely on heuristics since it is usually difficult to model the exact probability density of the features of real speech. Recently, a direct waveform-modification method using the spectrum differential feature has been proposed [12]. This method avoids generating over-smoothed spectra by transplanting the spectro-temporal fine texture of the source speech into the generated speech.

Most conventional VC frameworks are designed to convert acoustic features frame-by-frame. When it comes to a pronunciation-conversion task, we must consider the contextual or temporal dependencies of the conversion rules since the way speakers pronounce each phoneme may differ depending on the preceding, current, and succeeding contexts. To model long-term dependencies, some attempts have been made to model the mapping function using recurrent NNs (RNNs) [13] including the long short-term memory networks [14]. However, RNNs are not well suited to parallel implementations; thus, the conversion process becomes computationally demanding.

Except for some parallel-data-free frameworks [15, 16, 17], many VC frameworks require accurately aligned parallel data of source and target speech. When there is a large acoustic gap between source and target speech, collecting parallel speech data can be a painstaking process since automatic time alignment between the pair of acoustic-feature sequences becomes relatively difficult. Since many frameworks are weak against misalignment involved in parallel data, careful pre-screening and manual correction may be required to make these frameworks reliable. It is difficult to completely avoid misalignment errors particularly in a pronunciation-conversion task where the acoustic characteristics of each phoneme pronounced by source and target speakers can be very dissimilar.

In this paper, we propose a VC framework that (1) avoids over-smoothing of converted feature sequences, (2) provides a reasonable way of modeling time-dependent conversion rules, and (3) is robust against misalignment errors involved in the parallel data. We use a similarity metric implicitly obtained with a generative adversarial network (GAN) [18] that has been suc-

http://dx.doi.org/10.21437/Interspeech.2017-970

cessful in image generation [19, 20] and has begun to be applied in speech synthesis [21, 22, 23]. A GAN offers a framework for training a generator in such a way that it can deceive a discriminator that attempts to distinguish real data from the samples drawn from the generator. Thus, we expect that measuring the difference in the units in a particular layer in the GAN discriminator between the generated and target speech will enable the measurement of the similarity in high-level abstract space and mitigate the over-smoothing problem caused in the low-level data space. To model time-dependent conversion rules, we use a segmental feature (a sequence of acoustic features with a duration of hundreds of frames) as the input/output. For the generator, converter, and discriminator networks, we use a gated convolutional NN (CNN) [24], so that this network structure not only allows for modeling long-term dependencies by deepening the layers but also offers robustness against misalignment errors involved in the parallel data due to the nature of the CNN, known as the shift invariance property.

We previously proposed a learning-based postfilter using a GAN that adds a high-fidelty spectro-temporal texture to the feature sequence obtained with parametric speech synthesis so that the synthesized speech becomes as indistinguishable as possible from real speech [21]. The proposed framework combines the goals of VC and postfiltering, namely converting input speech as close as possible to target speech and making the output speech as indistinguishable as possible from real speech. We tested our framework on a non-native-to-native VC task. The experimental results showed that our proposed framework can be used to improve naturalness, clarity, and speaker individuality.

This paper is organized as follows. In Section 2, we explain the proposed sequence-to-sequence VC (SVC) framework with a GAN and gated CNN. We present the experimental results in Section 3 and summarize our findings in Section 4.

2. Sequence-to-sequence voice conversion

2.1. Framework overview

In a conventional VC framework, the goal is to convert voice characteristics, and a frame-by-frame approach, as shown in Figure 1(a), has been widely used for this conversion. In contrast, our goal is not only to convert voice characteristics but also convert pronunciation, e.g., to convert non-native speech to native-like speech. To achieve this goal, we need to model the temporal dependencies of a speech sequence. To achieve this, the conventional frame-by-frame framework is difficult to apply since it is limited to modeling the mapping function (namely, the converter (C)) in a short time (e.g., 5 ms) and is not able to model the temporal dependencies. Our proposed SVC framework was developed to overcome this limitation, as shown in Figure 1(b), which converts speech on a sequence-to-sequence basis using a C that models long-range context dependencies.

To obtain a suitable C, we need to find a suitable answers to the following two questions: "How is the C optimized?" and "How is the sequence structure modeled?". We answer the first question in Section 2.2 and the second question in Section 2.3.

2.2. Optimization with learned similarity metric and GAN

Sequence-to-sequence VC is considered a matching problem. Given a source acoustic-feature sequence $\boldsymbol{x} = (x_1, ..., x_T)$ and target acoustic-feature sequence $\boldsymbol{y} = (y_1, ..., y_T)$ as training data, the C that maps $\boldsymbol{x}$ to $\boldsymbol{y}$ can be optimized by minimizing

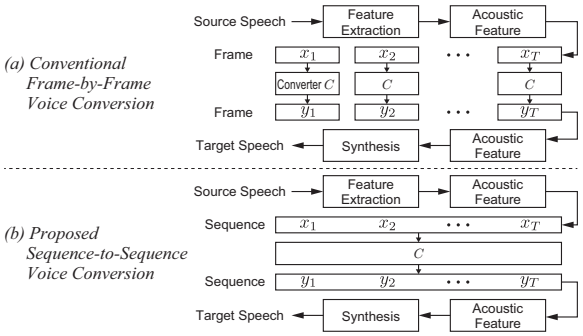


Figure 1: *Comparison of voice-conversion frameworks*

the following objective function:

$$\mathcal{L}_{\mathrm{SVC}} = S(\boldsymbol{y}, C(\boldsymbol{x})), \tag{1}$$

where $S(\boldsymbol{x}_1, \boldsymbol{x}_2)$ is a similarity metric.

In conventional VC frameworks, the similarity between the source and target acoustic features is measured in the data space using a particular form of similarity metric, e.g., MSE:

$$\mathcal{L}_{\mathrm{SVC}}^{\mathrm{Data}} = \frac{1}{M_0}||\boldsymbol{y} - C(\boldsymbol{x})||^2, \tag{2}$$

where M_0 is the dimensions of $\boldsymbol{y}$. The weak point of these frameworks is that they are sensitive to the slight translation in the data space. For example, this metric imposes a large error when a sharp peak is shifted by only one frame. This causes over-smoothing, resulting in buzzy-sounding speech.

To solve this problem, we use a similarity metric learned using a GAN [18]. In the following sub-sections, we first give a brief overview of a GAN then explain our proposed framework with the similarity metric learned using a GAN.

2.2.1. Background: GAN

A GAN [18] is a framework for training a deep generative model using a minimax game. The goal is to learn a generator distribution $P_G(\boldsymbol{y})$ that matches the true data distribution $P_{\mathrm{Data}}(\boldsymbol{y})$. It consists of two networks: a generator G that transforms noise variables $\boldsymbol{z} \sim P_{\mathrm{Noise}}(\boldsymbol{z})$ to data space $\boldsymbol{y} = G(\boldsymbol{z})$ and a discriminator D that assigns probability $p = D(\boldsymbol{y})$ when $\boldsymbol{y}$ is a sample from the $P_{\mathrm{Data}}(\boldsymbol{y})$ and assigns probability $1 - p$ when $\boldsymbol{y}$ is a sample from the $P_G(\boldsymbol{y})$. In a GAN, the following binary cross entropy is maximized and minimized with respect to the D and G, respectively.

$$\mathcal{L}_{\mathrm{GAN}} = \log D(\boldsymbol{y}) + \log(1 - D(G(\boldsymbol{z}))). \tag{3}$$

This enables the D to find the binary classifier that provides the best possible discrimination between true and generated data and simultaneously enables the G to fit $P_{\mathrm{Data}}(\boldsymbol{y})$. Both G and D can be trained using back-propagation.

2.2.2. Similarity metric learning with GAN

In a GAN, its D's hidden feature space is optimized to determine the key difference between true and generated data to discriminate them. This means that a more abstract structure is determined in the D's hidden feature space than that in the data space. We use this metric to measure similarity. This technique is also used in computer vision [25], in which this similarity metric enables a high-fidelity image to be generated. In the term

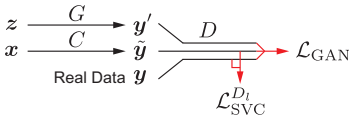


Figure 2: *Flow for training C with similarity metric learned with GAN. Red lines indicate terms in objective.*

$D_l(\boldsymbol{y})$, which denotes the l-th layer hidden feature in the D, a new similarity metric is written as

$$\mathcal{L}_{\mathrm{SVC}}^{D_l} = \frac{1}{M_l}||D_l(\boldsymbol{y}) - D_l(C(\boldsymbol{x}))||^2, \tag{4}$$

where M_l is the dimension of $D_l(\boldsymbol{y})$.

Following a previous study [25], we train our model with the joint objectives of the GAN objective with Equation 3 and SVC objective with Equation 4:

$$\mathcal{L} = \mathcal{L}_{\mathrm{SVC}}^{D_l} + \mathcal{L}_{\mathrm{GAN}}. \tag{5}$$

Figure 2 gives an overview of the training procedure. In practice, we obtain better results when using not only a sample from the G but also a sample from the C as the discriminator input. This is because similar positive and negative samples are more useful for the D to determine the decision boundary. In this setting, the GAN objective function is rewritten as

$$\begin{aligned}\mathcal{L}_{\mathrm{GAN}} = &\log D(\boldsymbol{y}) + \log(1 - D(G(\boldsymbol{z}))) \\ &+ \log(1 - D(C(\boldsymbol{x}))).\end{aligned} \tag{6}$$

We optimize the G by minimizing $\mathcal{L}_{\mathrm{GAN}}$, optimize the C by minimizing $\gamma\mathcal{L}_{\mathrm{SVC}}^{D_l} + \mathcal{L}_{\mathrm{GAN}}$, and optimize the D by minimizing $-\mathcal{L}_{\mathrm{GAN}}$. Here, γ is the trade-off parameter between $\mathcal{L}_{\mathrm{SVC}}^{D_l}$ and $\mathcal{L}_{\mathrm{GAN}}$, which we set to 1 in the experiments.

2.3. Sequence modeling with gated CNN

To achieve SVC, we need to model the long-range context-dependencies. To do so with a reasonably small number of parameters, we use a CNN. Figure 3 shows the network architectures of the G, C, and D. We consider converting the $F \times T$ time-sequential acoustic features (e.g., mel-cepstrum), where F is the feature dimension and T is the frame length. We carefully design the network architectures considering the objective of each network. To convert the source acoustic features to the target ones considering the temporal dependency in the overall features, we design the C using a one-dimensional (1D) CNN [26, 27]. In this model, we regard the feature dimension as a channel (i.e., RGB in the case of an image) and convolve the feature using a $(1, k_T)$ kernel, where k_T is the kernel width in the time direction. In the G and D, we focus on the 2D texture (i.e., spectro-temporal structure) of acoustic features. Based on this intuition, we treat them as image-like data [19, 20]. We design the G to generate the 2D texture from K-dimensional random noise using an up-sampling 2D CNN and design the D to discriminate the 2D texture using a down-sampling 2D CNN.

Based on the observation that acoustic features have region dependency (e.g., they have different structures in voiced segments and unvoiced segments), we design all the CNNs using a gated CNN [24]. In a gated CNN, gated linear units (GLUs) are used as an activation function instead of regular rectified linear units (ReLUs) [28] or Tanh activations. A GLU is a data-driven activation function, and the l-th-layer output $\boldsymbol{H}_l$ is calculated using the $(l-1)$-th-layer output $\boldsymbol{H}_{l-1}$ and model parameters $\boldsymbol{W}$, $\boldsymbol{V}$, $\boldsymbol{b}$, and $\boldsymbol{c}$.

$$\boldsymbol{H}_l = (\boldsymbol{H}_{l-1} * \boldsymbol{W} + \boldsymbol{b}) \otimes \sigma(\boldsymbol{H}_{l-1} * \boldsymbol{V} + \boldsymbol{c}), \tag{7}$$

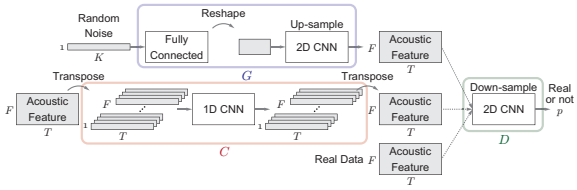


Figure 3: *Network architectures of G, C, and D*

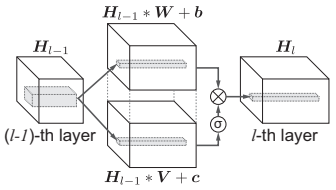


Figure 4: *Architectures of gated CNN*

where σ is the sigmoid function and $\otimes$ is the element-wise product between matrices. We show the gated CNN architecture in Figure 4. This gated mechanism enables the information to be propagated depending on the previous layer states and is well suited to represent region dependency.

3. Experiments

3.1. Experimental conditions

In our VC experiments, we used an Indian male speaker as the source (hereafter, the **SRC**) and an American male speaker as the target (the **TGT**). The data consisted of 842 pair utterances (roughly 1 hour). We used 83 utterances for evaluation and the others for training. The data were sampled at 16 kHz, then 25 mel-cepstral coefficients (MCEPs), logarithmic fundamental frequency (log F_0), and 5-band aperiodicities were extracted every 5 ms by using the STRAIGHT analysis system [29]. To obtain parallel utterances, we used dynamic time warping (DTW) to align MCEP sequences of the source and target speakers [1]. The MCEPs were converted using the C described in Section 2. Log F_0 was converted using a logarithm Gaussian-normalized transformation [30], i.e., linear conversion based on the mean and standard deviation of the source and target speech, which is widely used in VC. Aperiodicities were directly used without modification, since a previous study showed that converting aperiodicites does not provide a statistically significant difference on synthesized speech [31].

Table 1 details the network architectures of the G, C, and D. The symbols $\downarrow$ and $\uparrow$ indicate down-sampling and up-sampling, respectively. To upscale and downscale, we respectively used convolutions and backward convolutions with stride 2. The terms LReLU and BNorm denote leaky ReLU [32, 33] and batch normalization [34], respectively. We set the dimensions of input and output as $K = 100$, $F = 25$, and $T = 200$ (this means that a temporal dependency in 200 frames $\times$ 5 ms = 1 sec. was captured). In training, we fixed the frame length of the converter input, but we designed it as a fully convolutional network (FCN) [35], so we could take inputs of arbitrary length in synthesizing. During preprocessing, we normalized the source and target MCEPs to zero-mean and unit-variance for each dimension using their training sets, respectively. We trained the G, C, and D using the Adam optimizer [36] with a minibatch of size 16. The learning rate was set to 0.0002 for the C, 0.0001 for the G, and 0.0002 for the D, respectively; the momentum term β_1 was set to 0.5.

To clarify the characteristics of our framework with a learned similarity metric (the **LSM**), we implemented two

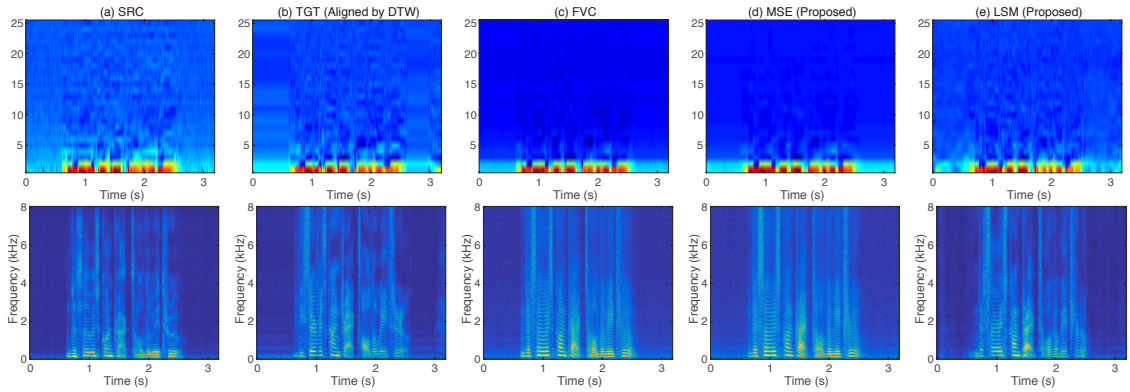


Figure 5: *Comparison of MCEPs (upper row) and STFT spectrograms (lower row) of SRC, TGT, FVC, MSE, and LSM*

Table 1: *Details of network architectures of G, C, and D*

G (**Input:** K Random noise, **Output:** $F \times T$ MCEP)
1024 fully connected, BNorm, ReLU
$[F/4] \cdot [T/4] \cdot 128$ fully connected, BNorm, ReLU
4×4 128 conv. $\uparrow$, BNorm, GLU
4×4 2 conv. $\uparrow$, GLU
C (**Input:** $F \times T$ MCEP, **Output:** $F \times T$ MCEP)
$(1 \times 11$ 1024 conv., GLU $) \times 5$ stacks
1×11 $2 \cdot F$ conv., GLU
D (**Input:** $F \times T$ MCEP, **Output:** 1 Probability)
4×4 128 conv. $\downarrow$, GLU
4×4 256 conv. $\downarrow$, BNorm, GLU
1024 fully connected, BNorm, LReLU
1 fully connected, sigmoid

frameworks for comparison. One is a state-of-the art frame-by-frame VC framework (the **FVC**) [6] that uses a deep autoencoder to obtain a compact representation of MCEPs. It has 7 hidden layers, and the numbers of units in input, hidden, and output layers are $[25, 100, 40, 15, 150, 15, 40, 100, 25]$. The other framework has the same converter as the LSM but it is optimized using MSE (the **MSE**).

3.2. Objective evaluation

Figure 5 shows the comparison of the MCEPs and STFT spectrograms of the SRC, TGT, FVC, MSE, and LSM. In the FVC and MSE, the spectral textures were over-smoothed both in the mel-cepstral and spectral domains. In the LSM, the spectral textures similar to the target ones were reproduced not only in the mel-cepstral domain but also in the spectral domain, although the spectral conversion is performed in the mel-cepstral domain.

3.3. Subjective evaluation

We conducted listening tests by referring to the Voice Conversion Challenge 2016 (VCC 2016) [37] at which naturalness and similarity were used as evaluation metrics. We also evaluated for clarity to assess the effectiveness in pronunciation conversion. We conducted an AB test on naturalness and clarity: participants were asked to select the preferred one or to opt for neutral if they did not have any preference. To measure the similarity of the VC samples, the same/different paradigm was used: participants were given two samples and asked whether the samples were produced by the same speaker. For evaluation, ten sample pairs were selected from the test data. In the AB test, all pairs were presented in both orders (AB and BA) to eliminate bias in the order of stimuli. There were seven participants who were well-educated English speakers.

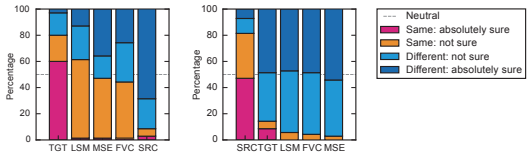
Tables 2 and 3 list the average preference scores for naturalness and clarity, respectively. These results indicate that the LSM outperformed the other two. In particular, the difference was marked regarding clarity, indicating that the LSM is effective against the over-smoothing problem. Figures 6 shows the similarity to the TGT and SRC with the three frameworks. All frameworks were competitive regarding dissimilarity to the SRC, but the LSM outperformed the MSE and FVC and exceeded the neutral rate regarding similarity to the TGT. Hence, we can conclude that the proposed framework can improve the quality of VC.

Table 2: *Average preference scores (%) for naturalness with 95% confidence intervals. Bold font indicates top scores.*

	Former	Latter	Neutral
FVC vs. LSM	32.9 ± 7.9	**54.3** ± 8.4	12.9 ± 5.6
MSE vs. LSM	27.1 ± 7.5	**65.0** ± 8.0	7.9 ± 4.5

Table 3: *Average preference scores (%) for clarity with 95% confidence intervals. Bold font indicates top scores.*

	Former	Latter	Neutral
FVC vs. LSM	17.1 ± 6.3	**72.9** ± 7.5	10.0 ± 5.0
MSE vs. LSM	10.0 ± 5.0	**84.3** ± 6.1	5.7 ± 3.9



(a) Similarity to TGT *(b) Similarity to SRC*

Figure 6: *Similarity to TGT and SRC with VC frameworks*

4. Conclusions

We explored sequence-to-sequence VC. To measure the similarity for complex sequential data, we proposed a VC framework that uses a similarity metric learned using a GAN. To model the sequence structure, we introduced a gated CNN into our model. In the experiments, we applied our framework to non-native-to-native VC tasks. The results showed that our proposed framework outperformed the state-of-the-art framework, as well as that optimized using a conventional similarity metric, in terms of naturalness, clarity, and speaker individuality. Our proposed framework using a learned similarity metric is not limited to a particular task, so we plan to apply it to other tasks such as TTS.

Acknowledgements: We thank Mr. Keisuke Oyamada (Univ. of Tsukuba), who kindly provided us with the source code of the state-of-the-art VC framework [6]. This work was supported by JSPS KAKENHI Grant Numbers JP26730100 and JP26280060.

5. References

[1] Y. Stylianou, O. Cappe, and E. Moulines, "Statistical methods for voice quality transformation," in *Proc. Eurospeech*, 1995, pp. 447–450.

[2] Y. Stylianou and O. Cappe, "A system for voice conversion based on probabilistic classification and a harmonic plus noise model," in *Proc. ICASSP*, 1998, pp. 281–284.

[3] T. Toda, A. W. Black, and K. Tokuda, "Voice conversion based on maximum-likelihood estimation of spectral parameter trajectory," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 15, no. 8, pp. 2222–2235, 2007.

[4] T. Nakashika, T. Takiguchi, and Y. Ariki, "Voice conversion based on speaker-dependent restricted Boltzmann machines," *IEICE Trans. Inf. Syst.*, vol. 97, no. 6, pp. 1403–1410, 2014.

[5] L.-H. Chen, Z.-H. Ling, L.-J. Liu, and L.-R. Dai, "Voice conversion using deep neural networks with layer-wise generative training," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 22, no. 12, pp. 1859–1872, 2014.

[6] S. H. Mohammadi and A. Kain, "Voice conversion using deep neural networks with speaker-independent pre-training," in *Proc. SLT*, 2014, pp. 19–23.

[7] R. Takashima, T. Takiguchi, and Y. Ariki, "Exampler-based voice conversion using sparse representation in noisy environments," *IEICE Trans. Inf. Syst.*, vol. E96-A, no. 10, pp. 1946–1953, 2013.

[8] Z. Wu, T. Virtanen, E. S. Chng, and H. Li, "Exemplar-based sparse representation with residual compensation for voice conversion," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 22, no. 10, pp. 1506–1521, 2014.

[9] T. Toda and K. Tokuda, "A speech parameter generation algorithm considering global variance for HMM-based speech synthesis," *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 5, pp. 816–824, 2007.

[10] H. Silén, E. Helander, J. Nurminen, and M. Gabbouj, "Ways to implement global variance in statistical speech synthesis," in *Proc. Interspeech*, 2012, pp. 1436–1439.

[11] S. Takamichi, T. Toda, G. Neubig, S. Sakti, and S. Nakamura, "A postfilter to modify the modulation spectrum in HMM-based speech synthesis," in *Proc. ICASSP*, 2014, pp. 290–294.

[12] K. Kobayashi, T. Toda, G. Neubig, S. Sakti, and S. Nakamura, "Statistical singing voice conversion with direct waveform modification based on the spectrum differential." in *Proc. Interspeech*, 2014, pp. 2514–2518.

[13] T. Nakashika, T. Takiguchi, and Y. Ariki, "High-order sequence modeling using speaker-dependent recurrent temporal restricted Boltzmann machines for voice conversion." in *Proc. Interspeech*, 2014, pp. 2278–2282.

[14] L. Sun, S. Kang, K. Li, and H. Meng, "Voice conversion using deep bidirectional long short-term memory based recurrent neural networks," in *Proc. ICASSP*, 2015, pp. 4869–4873.

[15] D. Erro, A. Moreno, and A. Bonafonte, "INCA algorithm for training voice conversion systems from nonparallel corpora," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 18, no. 5, pp. 944–953, 2010.

[16] T. Nakashika, T. Takiguchi, and Y. Ariki, "Parallel-data-free many-to-many voice conversion using an adaptive restricted Boltzmann machine," in *Proc. MLSP*, 2016.

[17] T. Kinnunen, L. Juvela, P. Alku, and J. Yamagishi, "Non-parallel voice conversion using i-vector PLDA: Towards unifying speaker verification and transformation," in *Proc. ICASSP*, 2017, pp. 5535–5539.

[18] 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*, 2014, pp. 2672–2680.

[19] E. Denton, S. Chintala, S. A., and R. Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Proc. NIPS*, 2015, pp. 1486–1494.

[20] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *Proc. ICLR*, 2016.

[21] T. Kaneko, H. Kameoka, N. Hojo, Y. Ijima, K. Hiramatsu, and K. Kashino, "Generative adversarial network-based postfilter for statistical parametric speech synthesis," in *Proc. ICASSP*, 2017, pp. 4910–4914.

[22] Y. Saito, S. Takamichi, and H. Saruwatari, "Training algorithm to deceive anti-spoofing verification for DNN-based speech synthesis," in *Proc. ICASSP*, 2017, pp. 4900–4904.

[23] T. Kaneko, S. Takaki, H. Kameoka, and J. Yamagishi, "Generative adversarial network-based postfilter for STFT spectrograms," in *Proc. Interspeech*, 2017.

[24] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *arXiv preprint arXiv:1612.08083*, 2016.

[25] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Proc. ICML*, 2016, pp. 1558–1566.

[26] Y. Kim, "Convolutional neural networks for sentence classification," in *arXiv preprint arXiv:1408.5882*, 2014.

[27] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. NIPS*, 2015, pp. 649–657.

[28] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, 2010, pp. 807–814.

[29] H. Kawahara, J. 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 *Proc. MAVEBA*, 2001, pp. 59–64.

[30] K. Liu, J. Zhang, and Y. Yan, "High quality voice conversion through phoneme-based linear mapping functions with straight for mandarin," in *Proc. FSKD*, 2007, pp. 410–414.

[31] Y. Ohtani, T. Toda, H. Saruwatari, and K. Shikano, "Maximum likelihood voice conversion based on GMM with STRAIGHT mixed excitation," in *Proc. Interspeech*, 2006, pp. 2266–2269.

[32] A. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. ICML Workshop*, 2013.

[33] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," in *Proc. ICML Workshop*, 2015.

[34] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, 2015, pp. 448–456.

[35] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. CVPR*, 2015, pp. 3431–3440.

[36] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[37] T. Toda, L.-H. Chen, D. Saito, F. Villavicencio, M. Wester, Z. Wu, and J. Yamagishi, "The voice conversion challenge 2016," in *Proc. Interspeech*, 2016, pp. 1632–1636.