GENERATIVE ADVERSARIAL NETWORK-BASED POSTFILTER FOR STATISTICAL PARAMETRIC SPEECH SYNTHESIS

Takuhiro Kaneko[†], *Hirokazu Kameoka*[†], *Nobukatsu Hojo*[‡]
Yusuke Ijima[‡], *Kaoru Hiramatsu*[†], *Kunio Kashino*[†]

[†]NTT Communication Science Laboratories, NTT Corporation, Japan
[‡]NTT Media Intelligence Laboratories, NTT Corporation, Japan

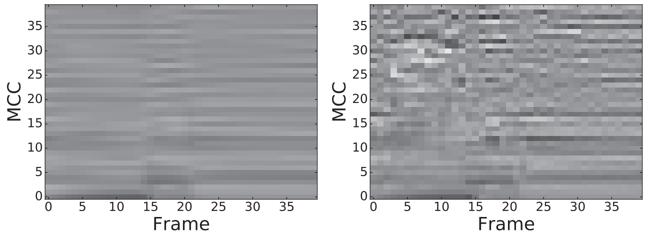
ABSTRACT

We propose a postfilter based on a generative adversarial network (GAN) to compensate for the differences between natural speech and speech synthesized by statistical parametric speech synthesis. In particular, we focus on the differences caused by over-smoothing, which makes the sounds muffled. Over-smoothing occurs in the time and frequency directions and is highly correlated in both directions, and conventional methods based on heuristics are too limited to cover all the factors (e.g., global variance was designed only to recover the dynamic range). To solve this problem, we focus on "spectral texture", i.e., the details of the time-frequency representation, and propose a learning-based postfilter that captures the structures directly from the data. To estimate the true distribution, we utilize a GAN composed of a generator and a discriminator. This optimizes the generator to produce samples imitating the dataset according to the adversarial discriminator. This adversarial process encourages the generator to fit the true data distribution, i.e., to generate realistic spectral texture. Objective evaluation of experimental results shows that the GAN-based postfilter can compensate for detailed spectral structures including modulation spectrum, and subjective evaluation shows that its generated speech is comparable to natural speech.

Index Terms— Statistical parametric speech synthesis, postfilter, deep neural network, generative adversarial network

1. INTRODUCTION

In the field of speech synthesis, statistical parametric speech synthesis based on hidden Markov models (HMMs) [1] or deep neural networks (DNNs) [2] has become popular since it provides various advantages over concatenative speech synthesis [3], such as the flexibility to control its voice characteristics [4, 5] and its compact footprint [6]. However, the quality of its synthesized speech is limited. There are three major factors behind the quality degradation [7]: vocoding, accuracy of acoustic models, and over-smoothing. In this paper, we focus on the over-smoothing problem.



(a) SYN (b) NAT

Fig. 1. Examples of spectral textures in Mel-cepstral domain. Our goal is to reconstruct the natural spectral texture (b) from the synthesized spectral texture (a).[1]

In synthesized speech, over-smoothing occurs in both the time and frequency directions. There have been several attempts to alleviate the problem in the respective directions. For example, in the frequency direction, a postfilter, which is common in speech coding [8, 9], is used to emphasize formants after generating parameters. In the time direction, a postfilter [10] is used to enhance spectral peaks, global variance (GV) [11] and variance scaling (VS) [12] are used to enhance variation of a spectral feature trajectory, and a postfilter [13] is used to enhance the modulation spectrum (MS). Although these methods help improve the quality of synthesized speech, they are based on empirical findings about acoustic differences between natural and synthesized speech. Therefore, each method is too limited to cover all the factors causing the differences, and there is still a large quality gap between natural and synthesized speech.

Considering this background, we propose a learning-based postfilter that learns the acoustic differences directly from the data. In particular, we focus on reconstructing detailed spectral structures in both the time and frequency directions simultaneously to handle over-smoothing that occurs in both directions. In other words, our goal is to reproduce a "spectral texture" like the natural one from the synthesized one, as shown in Fig. 1.

To achieve this goal, we need to represent the complex

[1] For ease of viewing, spectral textures are normalized for each dimension of Mel-cepstrum using the mean and standard deviation of synthesized data in training sets. Brighter color indicates a larger value.

978-1-5090-4117-6/17/$31.00 ©2017 IEEE

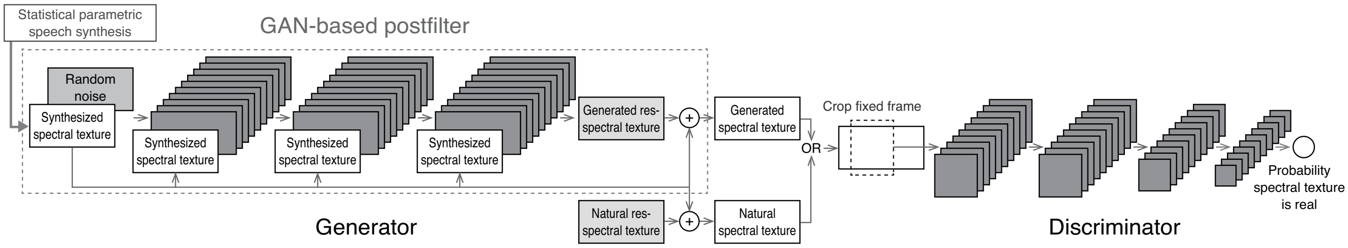

Statistical parametric speech synthesis
GAN-based postfilter
Random noise
Synthesized spectral texture
Synthesized spectral texture
Synthesized spectral texture
Synthesized spectral texture
Generated res-spectral texture
Generated spectral texture
Crop fixed frame
OR
Natural res-spectral texture
Natural spectral texture
Probability spectral texture is real
Generator
Discriminator

Fig. 2. System overview of proposed GAN-based postfilter.

distribution of the "spectral texture". A previous work [14, 15, 16] tried to represent it using a learning-based approach similar to ours, but they assumed its distribution as Gaussian (Chen *et al.* [14, 15] used RBM and Hashimoto *et al.* [16] used mean squared error (MSE) as a loss function), which still resulted in statistical averaging. To solve this problem, we exploit a generative adversarial network (GAN) [17] composed of a generator and a discriminator. This optimizes the generator to produce samples imitating the dataset according to the adversarial discriminator, which enables the generator to fit the true distribution, i.e., to generate realistic spectral texture. In computer vision, it has been shown that this contributes to the generation of sharp and fine images [18, 19, 20]. Objective evaluation of experimental results shows that the GAN-based postfilter outperforms the conventional postfilters [12, 13] as well as the DNN-based postfilter using MSE, and it can reconstruct detailed spectral structures including modulation spectra. Moreover, subjective evaluation shows that the quality of its generated speech is comparable to that of natural speech.

In Section 2 of this paper, we briefly review a GAN and explain the formulation to use it for postfiltering. In Section 3, we report experimental results. Section 4 provides a discussion and concludes the paper.

2. GAN-BASED POSTFILTER

2.1. Generative adversarial networks

A GAN [17] is a framework for estimating a generative model by an adversarial process. It consists of two networks: a *generator* network G that maps noise variable $z \sim P(z)$ to data space $x = G(z)$, and a *discriminator* network D that assigns a probability $p = D(x) \in [0, 1]$ when x is a real training sample and assigns $1 - p$ when x is generated by the generator. $P(z)$ is a prior on z and typically a uniform $[-1, 1]$ distribution is chosen. The GAN objective is to encourage D to find the binary classifier that provides the best possible discrimination between real and generated data and simultaneously to encourage G to fit the true data distribution. D and G play a two-player minmax game with the following binary cross entropy:

$$\begin{aligned}\min_G \max_D \; & \mathbb{E}_{x \sim P_{\mathrm{Data}}(x)}[\log D(x)] \\ & + \mathbb{E}_{z \sim P_{\mathrm{Noise}}(z)}[\log(1 - D(G(z)))]. \end{aligned} \tag{1}$$

Both G and D can be trained with backpropagation.

2.2. GANs for postfiltering

To utilize a GAN for postfiltering, we make three changes to the naive GAN architectures. The system overview is summarized in Fig. 2.

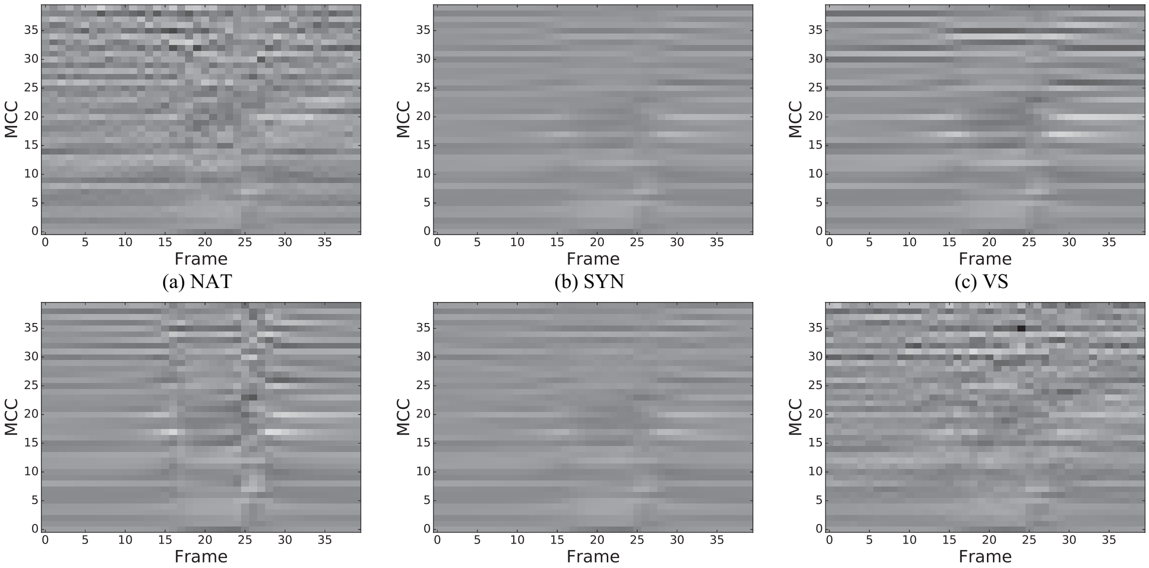
Conditional generative adversarial network: Our goal is to reconstruct a "spectral texture" from the synthesized one. To achieve this, we use a conditional generative adversarial network (CGAN) [18, 21], which is an extension of the GAN where G and D receive an additional vector y as input. In CGAN, the loss function can be rewritten as

$$\begin{aligned}\min_G \max_D \; & \mathbb{E}_{x,y \sim P_{\mathrm{Data}}(x,y)}[\log D(x, y)] \\ & + \mathbb{E}_{z \sim P_{\mathrm{Noise}}(z), y \sim P_y(y)}[\log(1 - D(G(z, y), y))]. \end{aligned} \tag{2}$$

We use y as a synthesized spectral texture. This model enables the generator to generate a realistic spectral texture conditioned on the given synthesized one. Here, z represents stochastic fluctuation in reconstructing a natural spectral texture from a synthesized one. As shown in Fig. 4, natural spectral vibrates irregularly, and z covers this phenomena.

Residual representation: In our task, the generator is expected to maintain a global structure and simultaneously to reconstruct small vibrations similar to a natural spectral texture. On the basis of intuition, we design the generator to produce the residual spectral texture instead of the raw one. This encourages the generator to learn the detailed differences between natural and synthesized spectral textures. This idea is inspired by the success of residual representation in computer vision [18, 22].

Convolutional architecture: To obtain the flexibility to allow temporal shift and contraction in the spectral structure, we use a convolutional neural network (CNN) that can capture the structure in both the time and frequency directions with a reasonably small number of parameters. In particular, we design the generator as a fully convolutional network (FCN) [23] to take spectral texture of arbitrary length. This solves the problem of continuity between frames. Note that a recurrent neural network (RNN) can be used as an alternative. The selection of CNN and RNN has been discussed in various areas and the comparison will be included in future work.



(a) NAT (b) SYN (c) VS

(d) MS (e) MSE (f) GAN

Fig. 3. Comparison of spectral textures generated by different methods.[1]

3. EXPERIMENTS

3.1. Experimental conditions

In the experiments, we used Japanese speech data from a professional female narrator. The data consisted of about 7,000 utterances (roughly 8 hours). We used 500 utterances for evaluation and the others for training. The data was sampled at 22.05 kHz, and 40 Mel-cepstral coefficients (MCCs), logarithmic fundamental frequency (log F_0), and 5-band aperiodicities were then extracted every 5 ms by the STRAIGHT analysis system [24]. In the extracted spectral features, we chose Mel-cepstrum as a target and applied postfilters to it, i.e., our goal is to reconstruct spectral textures in the Mel-cepstral domain. We used DNN-based statistical parametric speech synthesis [2] as the baseline, with 506 dimensional linguistic features as the input and 40 MCCs, log F_0, 5-band aperiodicities, their delta and delta-delta features, and a voiced/unvoiced binary value as the output. There were five hidden layers and 1,024 units per layer.

As explained in Section 2.2, we used the convolutional architectures (detailed in Table 1) for the generator and discriminator. As the input of the generator, we used the $D \times T$ Mel-cepstrum and the same-sized noise with dimension of MCCs $D = 40$ and frame length T. We design the generator as a fully convolutional network, so we can take inputs of arbitrary length. In the discriminator, we use a fully connected architecture at the last layer, so we used the fixed-size $D \times T_c$ Mel-cepstrum as the input with $T_c = 40$. The symbol $\downarrow$ represents down sampling. To downscale, we used convolutions with stride 2. The terms *ReLU* and *LReLU* indicate rectified activation [25] and leaky rectified activation [26, 27], respectively. During pre-processing, we normalized natural and synthesized Mel-cepstrum to zero-mean and unit-variance for each dimension using their training sets, respectively. We trained our model using the Adam optimizer [28] using an initial learning rate of 0.0001 and a batch size of 128.

Table 1. Network architectures for GAN-based postfilter.

Generator (**Input:** $D \times T$ Mel-cepstrum + $D \times T$ noise)
5×5 128 conv., ReLU + input Mel-cepstrum
5×5 256 conv., ReLU + input Mel-cepstrum
5×5 128 conv., ReLU + input Mel-cepstrum
5×5 1 conv.
Discriminator (**Input:** $D \times T_c$ Mel-cepstrum)
5×5 64 conv., LReLU
5×5 128 conv. $\downarrow$, LReLU
3×3 256 conv. $\downarrow$, LReLU
3×3 128 conv. $\downarrow$, LReLU
1 fully connected, sigmoid

To clarify the characteristics of our GAN-based postfilter (**GAN**), we compare it with various methods. **NAT** is extracted from a natural speech by STRAIGHT [24] (goal), and **SYN** is generated by DNN-based statistical parametric speech synthesis (baseline). We applied the following postfilters to SYN. **VS**: variance scaling-based postfilter [12]. **MS**: modulation spectrum-based postfilter [13], where DFT length for MS was set to 4,096 and α defining the amount of shift from synthetic to natural MS was set to 0.85 on the basis of the findings in [13]. **MSE**: DNN-based postfilter, where the generator has the same architecture as GAN but mean squared error is used as the loss function instead of the discriminator. **GANv**: applying the GAN-based postfilter only in voiced segments. In the boundary between voiced and unvoiced segments, we decreased the effect of the GAN-based postfilter linearly. We considered this method because the pre-experiments showed the quality of speech generated by GAN is often degraded in the unvoiced segments.

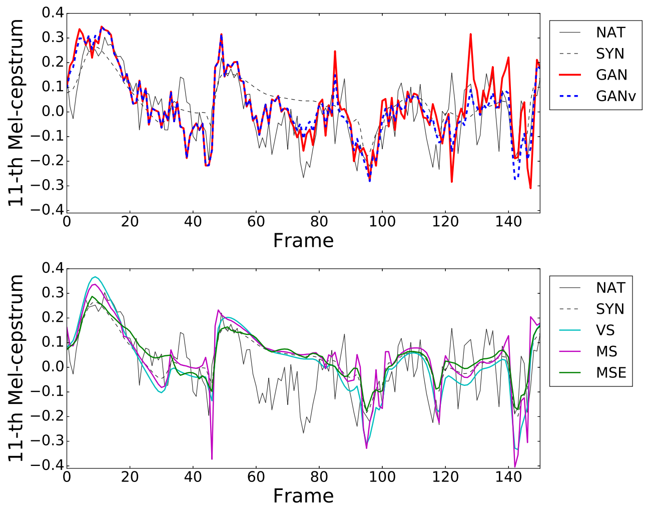


Fig. 4. Comparison of Mel-cepstral trajectories generated by different methods.[2]

3.2. Objective evaluation

Fig. 3 shows the example spectral textures generated by the different methods. We can clearly see that the proposed GAN-based postfilter is more successful in reconstructing spectral texture similar to the natural one. The spectral textures generated by VS and MSE are more similar to the synthesized one. The former is because VS only changes the scale in each dimension, and the latter is because statistical averaging still occurs in the model. MS partially emphasizes modulation, but its spectral texture is far from the natural one.

The Mel-cepstral trajectories predicted by the different methods are plotted in Fig. 4. As shown, the trajectories of SYN, VS, MS, and MSE are too smooth, but GAN and GANv can predict the trajectory that has a similar complexity to the natural one.

Mel-cepstral distortion is a popular measure to evaluate synthesized speech, but it is not suitable for our case because it uses squared error internally and cannot measure the fidelity of spectral texture. The difficulty of measuring such fidelity has been discussed in other areas [20, 29]. As an alternative, we calculated the differences in modulation spectra with respect to natural speech. Fig. 5 shows the difference averaged over utterances and all modulation frequencies for each Mel-cepstral coefficient. This result indicates that the GAN-based postfilter can also capture modulation characteristics. We expect this is because we use the convolutional architecture that can capture the relationship among consecutive frames.

3.3. Subjective evaluation

To evaluate the performance of the GAN-based postfilter, we conducted a subjective preference listening test (AB test). For comparison, we considered the following four sets: SYN vs. GAN, GAN vs. GANv, GAN vs. NAT, and GANv vs. NAT. For each set, 20 sample pairs are selected from test data. All pairs were presented in both orders (AB and BA) to eliminate bias in the order of stimuli. The number of listeners participating in the test was ten. For each sample pair, the participants were asked to choose the preferred one or to opt for neutral if they did not have any preference. To focus on the evaluation of the spectral quality, we used log F_0, aperiodicities, and voiced/unvoiced binary values extracted from natural speech.

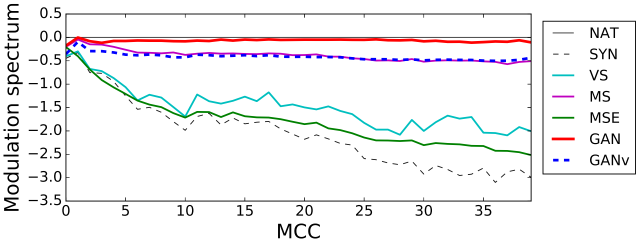


Fig. 5. Averaging difference in modulation spectrum per Mel-cepstral coefficient for different methods compared to natural speech.

Table 2 shows the experimental results. The results for GAN vs. SYN show that the GAN-based postfilter significantly improves the synthesized speech. GAN vs. GANv indicates that the GAN-based postfilter is effective particularly in voiced segments. In the results for GANv vs. NAT, all the scores are over 30% and there is no significant difference in 95% confidence intervals. This result indicates that GANv is comparable to NAT.

Table 2. Average preference score (%) with 95% confidence intervals. Bold font indicates the number is over 30%.

	Former	Latter	Neutral
GAN vs. SYN	**56.5** ± 4.9	22.0 ± 4.1	21.5 ± 4.0
GAN vs. GANv	11.3 ± 3.1	**37.3** ± 4.8	**51.5** ± 4.9
GAN vs. NAT	16.8 ± 3.7	**53.5** ± 4.9	29.8 ± 4.5
GANv vs. NAT	**30.3** ± 4.5	**34.5** ± 4.7	**35.3** ± 4.7

4. CONCLUSION

We examined a learning-based postfilter to reconstruct “spectral texture” from the over-smoothed synthesized spectral feature. To achieve this, we proposed a novel postfilter based on a generative adversarial network. Objective evaluation of the results showed that the proposed method can reproduce the detailed spectral structure including modulation spectra. Furthermore, subjective evaluation showed that the quality of speech generated by the proposed methods is comparable to that of natural speech.

Future work includes applying the GAN-based postfilter in a higher dimensional spectral domain instead of the Mel-cepstral domain, extending our methods to different network architectures such as RNNs, which are one of the popular methods in TTS [30], and optimizing the neural networks in end-to-end.

[2] For ease of viewing, the Mel-cepstral trajectories generated by our proposed methods are shown in the upper graph, while those generated by comparative methods are shown in the lower graph.

5. REFERENCES

[1] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, “Simultaneous modeling of spectrum, pitch and duration in HMM-based speech synthesis,” in *Proc. Eurospeech*, 1999, pp. 2347–2350.

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

[3] A. Hunt and A. Black, “Unit selection in a concatenative speech synthesis system using a large speech database,” in *Proc. ICASSP*, 1996, pp. 373–376.

[4] M. Tamura, T. Masuko, K. Tokuda, and T. Kobayashi, “Adaptation of pitch and spectrum for HMM-based speech synthesis using MLLR,” in *Proc. ICASSP*, 2001, pp. 805–808.

[5] T. Nose, J. Yamagishi, T. Masuko, and T. Kobayashi, “A style control technique for HMM-based expressive speech synthesis,” *IEICE Trans. Inf. Syst.*, vol. E90-D, no. 2, pp. 533–543, 2007.

[6] A. Gutkin, X. Gonzalvo, S. Breuer, and P. Taylor, “Quantized HMMs for low footprint text-to-speech synthesis,” in *Proc. Interspeech*, 2010.

[7] H. Zen, K. Tokuda, and A. Black, “Statistical parametric speech synthesis,” *Speech Commn.*, vol. 51, no. 11, pp. 1039–1064, 2009.

[8] K. Koishida, K. Tokuda, T. Kobayashi, and S. Imai, “CELP coding based on mel-cepstral analysis,” in *Proc. ICASSP*, 1995, pp. 33–36.

[9] T. Yoshimura, K. Tokuda, T. Masuko, T. Kobayashi, and T. Kitamura, “Incorporating a mixed excitation model and postfilter into HMM-based text-to-speech synthesis,” *Syst. Comput. Jpn.*, vol. 36, no. 12, pp. 43–50, 2005.

[10] Z. Ling, Y. Wu, Y. Wang, L. Qin, and R. Wang, “USTC system for Blizzard Challenge 2006 an improved HMM-based speech synthesis method,” in *Proc. Blizzard Challenge Workshop*, 2006.

[11] 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.

[12] 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.

[13] 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.

[14] L.-H. Chen, T. Raitio, C. Valentini-Botinhao, J. Yamagishi, and Z.-H. Ling, “DNN-based stochastic postfilter for HMM-based speech synthesis,” in *Proc. Interspeech*, 2014, pp. 1954–1958.

[15] L.-H. Chen, T. Raitio, C. Valentini-Botinhao, Z.-H. Ling, and J. Yamagishi, “A deep generative architecture for postfiltering in statistical parametric speech synthesis,” *IEEE Trans. Audio Speech and Lang. Process.*, vol. 23, no. 11, pp. 2003–2014, 2015.

[16] K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, “The effect of neural networks in statistical parametric speech synthesis,” in *Proc. ICASSP*, 2015, pp. 4455–4459.

[17] 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.

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

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

[20] A. B. L. Larsen, S. K. Sønderby, and O. Winther, “Autoencoding beyond pixels using a learned similarity metric,” in *Proc. ICML*, 2016.

[21] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” in *arXiv preprint arXiv:1411.1784*. 2014.

[22] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016.

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

[24] 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 Commn.*, vol. 27, no. 3, pp. 187–207, 1999.

[25] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proc. ICML*, 2010, pp. 807–814.

[26] A. Maas, A. Y. Hannun, and A. Y. Ng, “Rectifier nonlinearities improve neural network acoustic models,” in *Proc. ICML*, 2013.

[27] B. Xu, N. Wang, T. Chen, and M. Li, “Empirical evaluation of rectified activations in convolutional network,” in *arXiv preprint arXiv:1505.00853*. 2015.

[28] D. P. Kingma and M. Welling, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, 2015.

[29] L. Theis, A. Oord, and M. Bethge, “A note on the evaluation of generative models,” in *Proc. ICLR*, 2016.

[30] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, “TTS synthesis with bidirectional LSTM based recurrent neural networks,” in *Proc. Interspeech*, 2014, pp. 1964–1968.