WG-WaveNet: Real-Time High-Fidelity Speech Synthesis without GPU

Po-chun Hsu[12]*, Hung-yi Lee*[12]

[1]College of Electrical Engineering and Computer Science, National Taiwan University
[2]Graduate Institute of Communication Engineering, National Taiwan University

r07942095@ntu.edu.tw, hungyilee@ntu.edu.tw

Abstract

In this paper, we propose WG-WaveNet, a fast, lightweight, and high-quality waveform generation model. WG-WaveNet is composed of a compact flow-based model and a post-filter. The two components are jointly trained by maximizing the likelihood of the training data and optimizing loss functions on the frequency domains. As we design a flow-based model that is heavily compressed, the proposed model requires much less computational resources compared to other waveform generation models during both training and inference time; even though the model is highly compressed, the post-filter maintains the quality of generated waveform. Our PyTorch implementation can be trained using less than 8 GB GPU memory and generates audio samples at a rate of more than 960 kHz on an NVIDIA 1080Ti GPU. Furthermore, even if synthesizing on a CPU, we show that the proposed method is capable of generating 44.1 kHz speech waveform 1.2 times faster than real-time. Experiments also show that the quality of generated audio is comparable to those of other methods. Audio samples are publicly available online.

Index Terms: neural vocoder, raw waveform synthesis, text-to-speech

1. Introduction

Recently, neural network-based models have achieved state-of-the-art performance in speech tasks such as text-to-speech and voice conversion [1, 2, 3, 4]. These models are typically composed of two parts. The first model conducts the speech tasks and generates a spectrogram [1, 4], F0 frequencies, or other acoustic features [5]. The second part, referred to as a vocoder, is a generative model or a heuristic method transforming acoustic features into audio samples.

WaveNet [5] is first used as a neural vocoder to produce close-to-human natural speech [1]. The autoregressive architecture makes WaveNet capable of generating high-quality audio; however, it also leads to notably slow speed at inference time. To address this problem, several methods are proposed. One is modifying the architecture of the autoregressive model and applying a more compact framework to reduce the computing time of generating each audio sample [6, 7]. Without changing the nature of the autoregression, highly optimizing or weight pruning is still required to achieve real-time generation.

Another approach to improve the autoregressive model is based on the teacher-student framework. By applying knowledge distillation methods, a student model can learn from a well-trained teacher model to generate audio waveform in parallel. Although this framework has achieved remarkable real-time synthesis performance in [8] and [9], requirements of well-trained teacher models, highly-optimized distillation methods, and well-designed architectures remain problems for implementation.

WaveGlow [10] is a non-autoregressive model that can generate high-quality audio samples in parallel. This flow-based neural vocoder is trained only to maximize the likelihood of the training data. The simple network and single cost function make it straightforward to implement and to train. The model is fast enough to real-time synthesize audio waveform. However, since WaveGlow is deep and contains a large number of parameters, it consumes huge computational resources during training and inference.

In this paper, we aim to design an efficient, high-quality, and small footprint waveform generation model. We first apply the weight-sharing method to compress a WaveGlow and significantly reduce the size of the flow-based vocoder. A WaveNet-based post-filter is then applied to avoid the compression harming the speech quality. It is trained using loss functions on the frequency domains. Since the post-filter only needs to amend the output of the compressed WaveGlow, a small WaveNet is competent, keeping the overall model fast and lightweight. The proposed model, which we refer to as WG-WaveNet, possesses the advantage of simplicity in network architecture and loss function. Besides, compared with other neural vocoding methods, it requires much less computational cost during both training and inference.

The contributions of this work are summarized as follow:

- We propose a hybrid neural vocoder model, which is composed of a highly compressed WaveGlow model and a WaveNet-based post-filter. The proposed model, WG-WaveNet, is efficient and economical during training. WG-WaveNet can be trained on an NVIDIA 1080Ti GPU (using less than 8 GB GPU memory) in 4 days, while 8 NVIDIA GV100 GPUs were reported to use in the original WaveGlow paper [10].
- The proposed methods significantly improve the generating efficiency. In particular, the inference speed of the proposed WG-WaveNet is higher than 960 kHz using an NVIDIA 1080Ti GPU and 1.5 times faster than real-time even only using a CPU.
- For speech quality, perceptual experiments show that the proposed model can generate speech with a similar quality compared with WaveNet, WaveGlow, SqueezeWave [11], and Parallel WaveGAN [12].
- We also study the quality of 44.1 kHz audio waveform (which we call high-fidelity audio) generated from neural vocoders. We explore the performances of recordings with various sampling rates and the effects of different parameters of short-time Fourier transform for training vocoders. The proposed method not only makes it possible to synthesize 44.1 kHz audio samples on a single CPU 1.2 times faster than real-time but also achieves a score of 4.01 in the MOS test, which even betters 16 kHz recordings.

This work was supported by NVIDIA and Taiwan AI Labs.

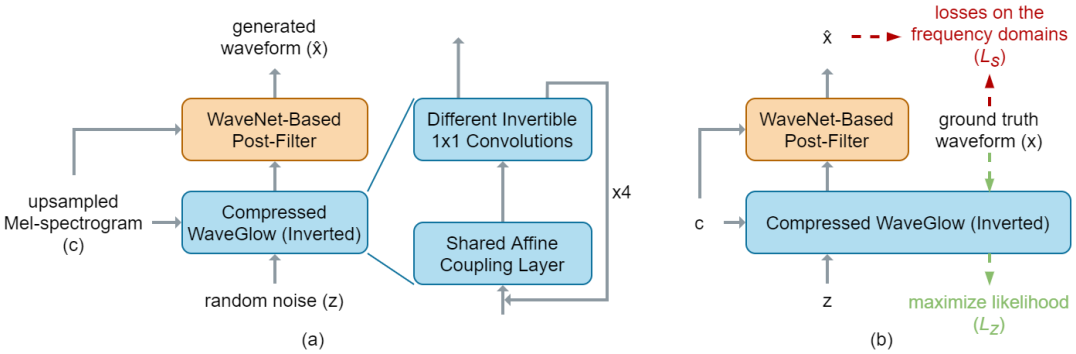

Figure 1: *(a) The architecture of WG-WaveNet. (b) Training of WG-WaveNet.*

2. Proposed Model

The proposed WG-WaveNet is composed of two components, shown in Figure 1(a). The first part is a highly compressed WaveGlow model, which will be introduced in Section 2.1. In Section 2.2, to further improve the sound quality, we employ a WaveNet-based post-filter trained with loss functions on the frequency domains.

2.1. Highly Compressed WaveGlow

WaveGlow [10] is a reversible network trained for modeling the distribution of the real-world speech data. During training, the model takes audio samples as input and Mel-spectrograms as the condition. It learns to transform the distribution of the audio samples in the training dataset to a zero-mean spherical Gaussian. The training objective is to maximize the likelihood of the training data. During inference time, an inverted WaveGlow takes a randomly-sampled Gaussian noise as input and generates speech waveform conditioned on a Mel-spectrogram.

The WaveGlow model consists of several transformations to progressively map speech data to Gaussian space. A transformation is composed of an affine coupling layer [13] and an invertible 1x1 convolution layer [14]. Each affine coupling layer in WaveGlow adopts a WaveNet-like module. The overall model is huge and hard to train.

We apply cross-layer parameter sharing to reduce parameters and make the model more compact. The cross-layer parameter sharing has shown to be helpful in NLP task pre-training [15] and source separation [16]. As shown in Figure 1(a), transformations in the compressed WaveGlow share the same affine coupling layer [1]. This approach keeps the model from drastically growing in size when it gets deeper. Considering that these transformations are processes of gradually mapping data from one distribution to another, invertible 1x1 convolution layers remain different across transformations to keep variability. We found that this improved the quality of generated speech in preliminary experiments. We also change the upsampling method from deconvolution to layers of duplication and convolution to further reduce parameters.

The training process is the same as mentioned in [10], shown by the green path in Figure 1(b). The loss function, denoted as L_z, is the negative log-likelihood of the training data. The proposed compression approach reduces the number of parameters in the WaveGlow and considerably cuts down the requirements of GPU memory. In the following section, we propose to use a post-filter to further speed up the convergence and improve the performance of the compressed WaveGlow.

[1]To make the affine coupling layer shareable here, we remove the early-output mechanism used in the original WaveGlow to keep the output shape the same across layers.

2.2. WaveNet-Based Post-Filter

A random noise z is sampled from a Gaussian as the input of the inverted compressed WaveGlow. The output of WaveGlow is then used as the input of the WaveNet-based post-filter to generates $\hat{x}$ in parallel [17, 18] conditioned on an upsampled Mel-spectrogram. The WaveNet-based post-filter is trained by minimizing the loss function $L_s(x, \hat{x})$, in which x is the ground truth samples, while $\hat{x}$ is the output of the post-filter. The WaveNet-based post-filter and the inverted compressed WaveGlow are jointly learned to minimize $L_s(x, \hat{x})$[2]. Since the WaveNet here synthesizes audio samples based on the output of the inverted compressed WaveGlow, its parameters can also be highly reduced.

For L_s, we utilize loss functions on the different frequency domains. Spectral losses have been shown effective for training waveform generation models in [19], [20], and [12]. We modify the multi-resolution Short-time Fourier transform (STFT) auxiliary loss in [12] as follows:

$$L_s(x, \hat{x}) = \frac{1}{M}\sum_{i=1}^{M}(L_{sc}^{i}(x, \hat{x}) + L_{mag}^{i}(x, \hat{x}) + L_{mel}^{i}(x, \hat{x})), \tag{1}$$

where M is the number of different parameter sets of STFT; L_{sc} and L_{mag} are the spectral convergence loss and the log STFT-magnitude loss from [21]:

$$L_{sc}(x, \hat{x}) = \frac{\big\||STFT(x)| - |STFT(\hat{x})|\big\|_F}{\big\||STFT(x)|\big\|_F} \tag{2}$$

$$L_{mag}(x, \hat{x}) = \frac{1}{N_{mag}}\left\|log\,|STFT(x)| - log\,|STFT(\hat{x})|\right\|_1, \tag{3}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\|\cdot\|_1$ is the L_1 norm, $|STFT(\cdot)|$ is the STFT magnitude, and N_{mag} is the number of elements in the magnitude. To make L_s more representative to human perception, we add a Mel-scale STFT-magnitude loss:

$$L_{mel}(x, \hat{x}) = \frac{1}{N_{mel}}\left\|log\,|MEL(x)| - log\,|MEL(\hat{x})|\right\|_1, \tag{4}$$

where $|MEL(\cdot)|$ and N_{mel} denote the Mel-scaled STFT magnitude and the number of elements in the magnitude, respectively. The number of Mel bands differs in different STFT parameter sets.

The WaveNet-based post-filter is trained jointly with the inverted compressed WaveGlow, as shown by the red path in Figure 1(b). The loss function for training WG-WaveNet is a linear combination of L_z and L_s:

[2]Since the WaveNet-based post-filter is irreversible, it can not be trained jointly by maximizing the likelihood as WaveGlow.

$$L_{total} = \lambda L_z + L_s, \quad (5)$$

where λ is a scalar to balance the loss terms. In practice, L_s is calculated every n iterations.

Eventually, the overall WG-WaveNet (compressed WaveGlow plus WaveNet postfilter) is *one thirty-fifth* of the original WaveGlow in model size. Model details will be discussed in Section 3.2.

3. Experiments

3.1. Datasets

Two datasets were used in the experiments. One was the LJ Speech Dataset [22]. This English dataset consists of 13100 clean audio clips (about 24 hours) of a female speaker. The sampling rate is 22050. The other was an internal Mandarin corpus, which contains 9004 utterances (about 6.8 hours) from a female speaker. The recordings were sampled at 44.1 kHz. 100 utterances were selected from each dataset for evaluation.

We used the 80-band Mel-spectrogram as the condition to synthesize audio. For WG-WaveNet, the FFT size, hop size and window size for STFT are 2048, 200, and 800, respectively.

3.2. Model Details

The numbers of parameters of different models are listed in Table 2. The WaveNet-based post-filter in the proposed WG-WaveNet is composed of 7 layers of dilated convolution blocks with 64 channels. The original WaveGlow has 12 transformations. With the help of the post-filter, the compressed WaveGlow consists of only 4 transformations. The WaveNet-like module in the shared affine coupling layer has 7 layers with 128 channels. Both the WaveNet-based post-filter and the compressed WaveGlow have fewer numbers of layers and channels than the original WaveNet and WaveGlow, making WG-WaveNet much more compact. WaveNet has 24.7 M parameters, and WaveGlow has 87.9 M parameters. WG-WaveNet, on the other hand, has only 2.5 M parameters, which is 1/10 and 1/35 of those in WaveNet and WaveGlow, respectively.

We compared our method with four different baseline models, WaveNet, WaveGlow, SqueezeWave, and Parallel WaveGAN. To ensure that the models were consistent compared to the original models, for the first three, we used pre-trained models from public implementations[3][4][5]. Note that the pre-trained models of WaveGlow and SqueezeWave were released by the official. We followed the setup in [12] to train the Parallel WaveGAN. The WG-WaveNet model was trained for 1 M steps using the Adam optimizer [23] with a batch size of 8. The learning rate was $4e^{-4}$ and reduced by half every 200 K steps. We set $\lambda = 1$ and $n = 3$ based on preliminary experiments. The parameters for calculating L_s in Section 2.2 are listed in Table 1.

Table 1: *The parameters for calculating L_s.*

FFT size	4096, 2048, 1024, 512, 256
hop size	400, 200, 100, 50, 25
window size	1600, 800, 400, 200, 100
# of Mel bands	640, 320, 160, 80, 40

3.3. Speed and Computational Cost

We evaluated the speed and memory usage of different models during training and inference. Parallel WaveGAN and WG-WaveNet were trained on the same server using an Nvidia V100 16GB RAM GPU to fairly evaluate the computational cost at the training stage. The testing environment was a personal computer with an Intel i7-6700K CPU and an Nvidia 1080Ti GPU. Since the computational cost of parallel synthesis methods might be affected by the output length at the inference stage, we tested the models using utterances with various lengths uniformly distributed from 2 to 9 seconds.

Table 2: *Comparison of computational cost and speed during training and inference time. The units of memory, time, and speed are GB, day, and kHz, respectively. Details of training and inference are described in Section 3.3.*

Model	**Size**	**Training Mem. / Time**	**Infer. Speed CPU / GPU**
WaveNet	24.7 M	-	0.1 / 0.12
WaveGlow	87.9 M	-	10 / 279
SqueezeWave	23.7 M	-	330 / 4486
Parallel WaveGAN	1.3 M	14.4 / 2.7	18 / 841
WG-WaveNet (ours)	2.5 M	7.7 / 3.5	33 / 967
WG-WaveNet (g-20)	3.1 M	**5.2 / 2.5**	**53 / 1634**

The results are shown in Table 2. Though the training time of WG-WaveNet is slightly longer than that of Parallel WaveGAN, the training memory is 47% less. The inference speed of WG-WaveNet is at a rate of 967 kHz with GPU and 1.5 times faster than real-time without GPU. Moreover, we trained a faster version of WG-WaveNet, denoted as g-20. In WaveGlow, the input is reshaped to groups of 8 samples [10]. Inspired by [11], the input of g-20 is reshaped to groups of 20 samples. This faster WG-WaveNet can be optimized with much less computational resources and generate 22 kHz speech 2.4 times faster than real-time without GPU.

3.4. Audio Quality Comparison

We conducted Mean Opinion Score (MOS) tests[6] as a subjective evaluation (higher is better) and calculated Mel Cepstral Distortion (MCD) [24] as an objective evaluation (lower is better). In the MOS test, raters were asked to score utterances on a five-point scale according to the quality. Each utterance was randomly selected from the evaluation set and scored by at least 20 raters.

The evaluation results are shown in Table 3. To assess the effects of L_z and L_s on model performance, we trained WG-WaveNet with different λ and n. Figure 2 shows the trade-off between audio quality and inference speed.

The observations based on Table 3 and Figure 2 are concluded as follows: (1) WaveNet has the highest MOS, which is close to that of the ground truth data, yet there is a gap between the performance of parallel and autoregressive synthesis methods. (2) MCD is not strongly related to human perception. Training a model using L_s ($\lambda = 0, n = 1$) leads to the lowest MCD but not the best MOS, while WaveNet has the highest MOS and MCD. A similar contradictory result was also found in [6]. (3) Though we used the official release models to synthesize utterances, WaveGlow and SqueezeWave did not perform well. Subjects reported there were noise and reverberation effects in the generated speech. (4) The ablation study shows that both L_z and L_s are crucial for training WG-WaveNet. We found that training using only L_s ($\lambda = 0, n = 1$) led to good quality at voiced part of speech but significant high-frequency glitch at unvoiced part. (5) The MOS decreases rapidly when

[3] https://github.com/r9y9/wavenet_vocoder

[4] https://github.com/NVIDIA/waveglow

[5] https://github.com/tianrengao/SqueezeWave

[6] Audio samples are publicly available at https://bogihsu.github.io/WG-WaveNet/

Table 3: *MOS and MCD results compared with other models. Mel-spectrograms were extracted from the ground truth. The MOS results are reported with 95% confidence intervals.*

Model	MOS	MCD
WaveNet	**4.49±0.101**	4.619
WaveGlow	3.71±0.159	4.393
SqueezeWave	2.96±0.121	3.608
Parallel WaveGAN	4.24±0.108	4.026
WG-WaveNet (ours)		
$\lambda = 1, n = 3$	4.08±0.118	3.783
$\lambda = 1, n = 1$	3.23±0.159	2.948
$\lambda = 0, n = 1$	3.65±0.164	**2.407**
g-20	3.75±0.124	3.848
Ground Truth	4.61±0.096	-

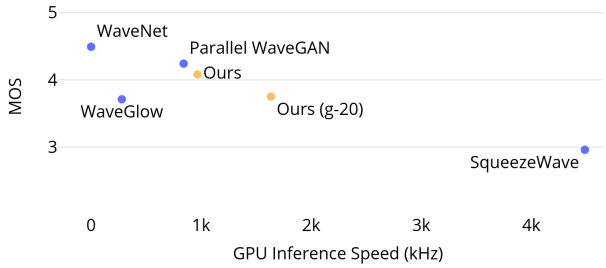

Figure 2: *Trade-off between MOS and GPU inference speed.*

the generating efficiency improves. WG-WaveNet, however, has the faster speed and a MOS of 4.08, which is close to that of Parallel WaveGAN. This indicates the proposed WG-WaveNet can greatly increase the synthesis speed while preserving a comparable performance.

3.5. High-Fidelity Audio Generation

Due to the fast inference speed of WG-WaveNet as shown in Sections 3.3 and 3.4, we show that WG-WaveNet can generate high-fidelity audio (44.1 kHz) in this subsection. To evaluate the performance, we trained WG-WaveNet and Parallel WaveGAN on the 44.1 kHz speech dataset mentioned in Section 3.1. We only compared WG-WaveNet with Parallel WaveGAN here because only Parallel WaveGAN and SqueezeWave are efficient enough to synthesize 44.1 kHz audio, and the audio quality of SqueezeWave is not comparable with Parallel WaveGAN. MOS tests with the same setups as those in Section 3.4 were conducted on the generated waveform and ground truth data with different sampling rates.

The results are shown in Table 4. "w800" denotes that the window size for extracting Mel-spectrograms is set to 800. The FFT size, hop size, and the number of Mel bands are also the same as mentioned in 3.1. "w1600" denotes that the window size is doubled to 1600, and the other parameters are also doubled. Since the sampling rate is changed from 22050 to 44100, doubling STFT parameters (w1600) makes the temporal resolution of extracted features the same as in Section 3.4, while the temporal resolution is doubled in "w800". Similarly, parameters for calculating L_s in "w800" are the same as in Table 1, while they are doubled in "w1600". We first found that the sampling rates of the ground truth samples significantly affect their perceptual scores. The raters considered the ground truths with higher sampling rates are better. Experiments reveal that when the temporal resolution of acoustic features is fixed (w1600), it is harder to generate 44.1 kHz speech than to generate the 22 kHz one. We observed that Mel-spectrograms with higher temporal resolution (w800) helped improve the performance of WG-WaveNet (w800). WG-WaveNet outperformed Parallel WaveGAN in both "w800" and "w1600" cases. Eventually, the faster WG-WaveNet reached 4.01 MOS, which is even better than that of 16 kHz ground truth speech.

Table 4: *MOS results of high-fidelity audio generation with 95% confidence intervals. Mel-spectrograms were extracted from the ground truth sampled at 44.1 kHz.*

Model	MOS
Parallel WaveGAN	
w1600	3.12±0.134
w800	3.04±0.126
WG-WaveNet (ours)	
w1600	3.15±0.148
w800	3.71±0.131
w800 (g-20)	**4.01±0.110**
Ground Truth (16 kHz)	3.72±0.147
Ground Truth (22 kHz)	4.15±0.127
Ground Truth (44.1 kHz)	4.44±0.105

Table 5: *MOS results and GPU inference speed (in kHz) compared with other models. Mel-spectrograms were generated by the Tacotron 2 model. The MOS results are reported with 95% confidence intervals.*

Model	MOS	Infer. Speed
Tacotron 2+GL	2.11±0.139	-
Tacotron 2+WaveNet	**3.96±0.116**	0.12
Tacotron 2+Parallel WaveGAN	3.72±0.127	841
Tacotron 2+WG-WaveNet (ours)	3.68±0.133	**967**
Ground Truth	4.36±0.108	-

3.6. Text-to-Speech

We combined WG-WaveNet with a Tacotron 2 model to evaluate the proposed method as a vocoder. The Tacotron 2 was built following [1]. Data preprocessing for training the TTS model and vocoders were set to the same as mentioned in Section 3.1.

The results of MOS tests and GPU inference speed of vocoders are reported in Table 5. Note that the ground truth inherently has better prosody and quality than those of the speech generated by Tacotron 2. We found that the performance gap between WaveNet and the parallel synthesis methods narrowed. WG-WaveNet has the MOS comparable to that of Parallel WaveGAN, and the inference speed is faster than other methods, which shows the advantage of WG-WaveNet as a vocoder for fast high-quality speech synthesis.

4. Conclusion

We proposed WG-WaveNet, a fast, lightweight, and high-quality waveform generation model. Combining with a highly compressed WaveGlow and a WaveNet-based post-filter, WG-WaveNet requires much less computational resources compared to other parallel synthesis methods during both training and inference time. The experimental results show that WG-WaveNet is capable of generating high-quality 22 kHz and 44.1 kHz audio samples faster than real time without GPU.

5. References

[1] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.

[2] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural speech synthesis with transformer network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6706–6713.

[3] J.-c. Chou, C.-c. Yeh, H.-y. Lee, and L.-s. Lee, "Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations," *arXiv preprint arXiv:1804.02812*, 2018.

[4] K. Qian, Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson, "Autovc: Zero-shot voice style transfer with only autoencoder loss," 2019.

[5] A. v. d. 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 preprint arXiv:1609.03499*, 2016.

[6] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "Fftnet: A real-time speaker-dependent neural vocoder," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2251–2255.

[7] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," *arXiv preprint arXiv:1802.08435*, 2018.

[8] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, "Parallel wavenet: Fast high-fidelity speech synthesis," *arXiv preprint arXiv:1711.10433*, 2017.

[9] W. Ping, K. Peng, and J. Chen, "Clarinet: Parallel wave generation in end-to-end text-to-speech," *arXiv preprint arXiv:1807.07281*, 2018.

[10] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3617–3621.

[11] B. Zhai, T. Gao, F. Xue, D. Rothchild, B. Wu, J. E. Gonzalez, and K. Keutzer, "Squeezewave: Extremely lightweight vocoders for on-device speech synthesis," *arXiv preprint arXiv:2001.05685*, 2020.

[12] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6199–6203.

[13] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real nvp," *arXiv preprint arXiv:1605.08803*, 2016.

[14] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 215–10 224.

[15] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[16] C.-I. Tuan, Y.-K. Wu, H.-y. Lee, and Y. Tsao, "Mitas: A compressed time-domain audio separation network with parameter sharing," *arXiv preprint arXiv:1912.03884*, 2019.

[17] F. Lluís, J. Pons, and X. Serra, "End-to-end music source separation: is it possible in the waveform domain?" *arXiv preprint arXiv:1810.12187*, 2018.

[18] D. Rethage, J. Pons, and X. Serra, "A wavenet for speech denoising," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5069–5073.

[19] S. Takaki, T. Nakashika, X. Wang, and J. Yamagishi, "Stft spectral loss for training a neural speech waveform model," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 7065–7069.

[20] S. Takaki, H. Kameoka, and J. Yamagishi, "Training a neural speech waveform model using spectral losses of short-time fourier transform and continuous wavelet transform," *arXiv preprint arXiv:1903.12392*, 2019.

[21] S. Ö. Arık, H. Jun, and G. Diamos, "Fast spectrogram inversion using multi-head convolutional neural networks," *IEEE Signal Processing Letters*, vol. 26, no. 1, pp. 94–98, 2018.

[22] K. Ito, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.

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

[24] R. Kubichek, "Mel-cepstral distance measure for objective speech quality assessment," in *Proceedings of IEEE Pacific Rim Conference on Communications Computers and Signal Processing*, vol. 1. IEEE, 1993, pp. 125–128.