# HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis

**Jungil Kong**
Kakao Enterprise
henry.k@kakaoenterprise.com

**Jaehyeon Kim**
Kakao Enterprise
jay.xyz@kakaoenterprise.com

**Jaekyoung Bae**
Kakao Enterprise
storm.b@kakaoenterprise.com

## Abstract

Several recent work on speech synthesis have employed generative adversarial networks (GANs) to produce raw waveforms. Although such methods improve the sampling efficiency and memory usage, their sample quality has not yet reached that of autoregressive and flow-based generative models. In this work, we propose HiFi-GAN, which achieves both efficient and high-fidelity speech synthesis. As speech audio consists of sinusoidal signals with various periods, we demonstrate that modeling periodic patterns of an audio is crucial for enhancing sample quality. A subjective human evaluation (mean opinion score, MOS) of a single speaker dataset indicates that our proposed method demonstrates similarity to human quality while generating 22.05 kHz high-fidelity audio 167.9 times faster than real-time on a single V100 GPU. We further show the generality of HiFi-GAN to the mel-spectrogram inversion of unseen speakers and end-to-end speech synthesis. Finally, a small footprint version of HiFi-GAN generates samples 13.4 times faster than real-time on CPU with comparable quality to an autoregressive counterpart.

## 1 Introduction

Voice is one of the most frequent and naturally used communication interfaces for humans. With recent developments in technology, voice is being used as a main interface in artificial intelligence (AI) voice assistant services such as Amazon Alexa, and it is also widely used in automobiles, smart homes and so forth. Accordingly, with the increase in demand for people to converse with machines, technology that synthesizes natural speech like human speech is being actively studied.

Recently, with the development of neural networks, speech synthesis technology has made a rapid progress. Most neural speech synthesis models use a two-stage pipeline: 1) predicting a low resolution intermediate representation such as mel-spectrograms (Shen et al., 2018, Ping et al., 2017, Li et al., 2019) or linguistic features (Oord et al., 2016) from text, and 2) synthesizing raw waveform audio from the intermediate representation (Oord et al., 2016, 2018, Prenger et al., 2019, Kumar et al., 2019). The first stage is to model low-level representations of human speech from text, whereas the second stage model synthesizes raw waveforms with up to 24,000 samples per second and up to 16 bit fidelity. In this work, we focus on designing a second stage model that efficiently synthesizes high fidelity waveforms from mel-spectrograms.

Various work have been conducted to improve the audio synthesis quality and efficiency of second stage models. WaveNet (Oord et al., 2016) is an autoregressive (AR) convolutional neural network that demonstrates the ability of neural network based methods to surpass conventional methods in quality.

However, because of the AR structure, WaveNet generates one sample at each forward operation; it is prohibitively slow in synthesizing high temporal resolution audio. Flow-based generative models are proposed to address this problem. Because of their ability to model raw waveforms by transforming noise sequences of the same size in parallel, flow-based generative models fully utilize modern parallel computing processors to speed up sampling. Parallel WaveNet (Oord et al., 2018) is an inverse autoregressive flow (IAF) that is trained to minimize its Kullback-Leibler divergence from a pre-trained WaveNet called a teacher. Compared to the teacher model, it improves the synthesis speed to 1,000 times or more, without quality degradation. WaveGlow (Prenger et al., 2019) eliminates the need for distilling a teacher model, and simplifies the learning process through maximum likelihood estimation by employing efficient bijective flows based on Glow (Kingma and Dhariwal, 2018). It also produces high-quality audio compared to WaveNet. However, it requires many parameters for its deep architecture with over 90 layers.

Generative adversarial networks (GANs) (Goodfellow et al., 2014), which are one of the most dominant deep generative models, have also been applied to speech synthesis. Kumar et al. (2019) proposed a multi-scale architecture for discriminators operating on multiple scales of raw waveforms. With sophisticated architectural consideration, the MelGAN generator is compact enough to enable real-time synthesis on CPU. Yamamoto et al. (2020) proposed multi-resolution STFT loss function to improve and stabilize GAN training and achieved better parameter efficiency and less training time than an IAF model, ClariNet (Ping et al., 2018). Instead of mel-spectrograms, GAN-TTS (Bińkowski et al., 2019) successfully generates high quality raw audio waveforms from linguistic features through multiple discriminators operating on different window sizes. The model also shows fewer FLOPs compared to Parallel WaveNet. Despite the advantages, there is still a gap in sample quality between the GAN models and AR or flow-based models.

We propose HiFi-GAN, which achieves both higher computational efficiency and sample quality than AR or flow-based models. As speech audio consists of sinusoidal signals with various periods, modeling the periodic patterns matters to generate realistic speech audio. Therefore, we propose a discriminator which consists of small sub-discriminators, each of which obtains only a specific periodic parts of raw waveforms. This architecture is the very ground of our model successfully synthesizing realistic speech audio. As we extract different parts of audio for the discriminator, we also design a module that places multiple residual blocks each of which observes patterns of various lengths in parallel, and apply it to the generator.

HiFi-GAN achieves a higher MOS score than the best publicly available models, WaveNet and WaveGlow. It synthesizes human-quality speech audio at speed of 3.7 MHz on a single V100 GPU. We further show the generality of HiFi-GAN to the mel-spectrogram inversion of unseen speakers and end-to-end speech synthesis. Finally, the tiny footprint version of HiFi-GAN requires only 0.92M parameters while outperforming the best publicly available models and the fastest version of HiFi-GAN samples 13.44 times faster than real-time on CPU and 1,186 times faster than real-time on single V100 GPU with comparable quality to an autoregressive counterpart.

Our audio samples are available on the demo web-site[1], and we provide the implementation as open source for reproducibility and future work.[2]

## 2 HiFi-GAN

### 2.1 Overview

HiFi-GAN consists of one generator and two discriminators: multi-scale and multi-period discriminators. The generator and discriminators are trained adversarially, along with two additional losses for improving training stability and model performance.

### 2.2 Generator

The generator is a fully convolutional neural network. It uses a mel-spectrogram as input and upsamples it through transposed convolutions until the length of the output sequence matches the temporal resolution of raw waveforms. Every transposed convolution is followed by a multi-receptive

---

[1] https://jik876.github.io/hifi-gan-demo/
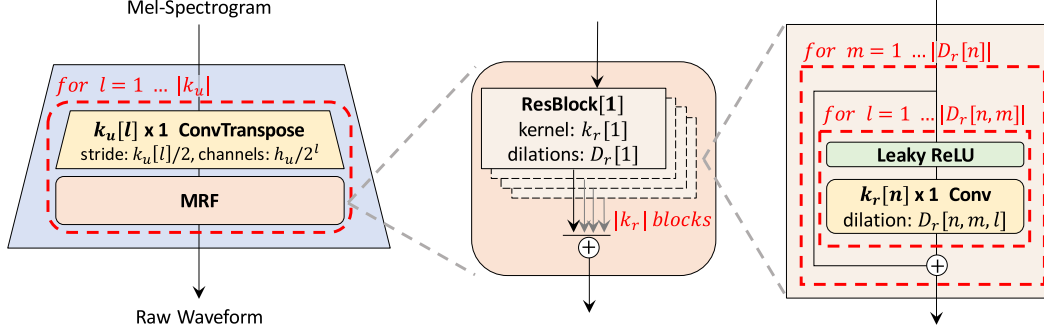[2] https://github.com/jik876/hifi-gan

Figure 1: The generator upsamples mel-spectrograms up to $|k_u|$ times to match the temporal resolution of raw waveforms. A MRF module adds features from $|k_r|$ residual blocks of different kernel sizes and dilation rates. Lastly, the $n$-th residual block with kernel size $k_r[n]$ and dilation rates $D_r[n]$ in a MRF module is depicted.

field fusion (MRF) module, which we describe in the next paragraph. Figure 1 shows the architecture of the generator. As in previous work (Mathieu et al., 2015, Isola et al., 2017, Kumar et al., 2019), noise is not given to the generator as an additional input.

**Multi-Receptive Field Fusion** We design the multi-receptive field fusion (MRF) module for our generator, which observes patterns of various lengths in parallel. Specifically, MRF module returns the sum of outputs from multiple residual blocks. Different kernel sizes and dilation rates are selected for each residual block to form diverse receptive field patterns. The architectures of MRF module and a residual block are shown in Figure 1. We left some adjustable parameters in the generator; the hidden dimension $h_u$, kernel sizes $k_u$ of the transposed convolutions, kernel sizes $k_r$, and dilation rates $D_r$ of MRF modules can be regulated to match one's own requirement in a trade-off between synthesis efficiency and sample quality.

## 2.3 Discriminator

Identifying long-term dependencies is the key for modeling realistic speech audio. For example, a phoneme duration can be longer than 100 ms, resulting in high correlation between more than 2,200 adjacent samples in the raw waveform. This problem has been addressed in the previous work (Donahue et al., 2018) by increasing receptive fields of the generator and discriminator. We focus on another crucial problem that has yet been resolved; as speech audio consists of sinusoidal signals with various periods, the diverse periodic patterns underlying in the audio data need to be identified.

To this end, we propose the multi-period discriminator (MPD) consisting of several sub-discriminators each handling a portion of periodic signals of input audio. Additionally, to capture consecutive patterns and long-term dependencies, we use the multi-scale discriminator (MSD) proposed in MelGAN (Kumar et al., 2019), which consecutively evaluates audio samples at different levels. We conducted simple experiments to show the ability of MPD and MSD to capture periodic patterns, and the results can be found in Appendix B.

**Multi-Period Discriminator** MPD is a mixture of sub-discriminators, each of which only accepts equally spaced samples of an input audio; the space is given as period $p$. The sub-discriminators are designed to capture different implicit structures from each other by looking at different parts of an input audio. We set the periods to [2, 3, 5, 7, 11] to avoid overlaps as much as possible. As shown in Figure 2b, we first reshape 1D raw audio of length $T$ into 2D data of height $T/p$ and width $p$ and then apply 2D convolutions to the reshaped data. In every convolutional layer of MPD, we restrict the kernel size in the width axis to be 1 to process the periodic samples independently. Each sub-discriminator is a stack of strided convolutional layers with leaky rectified linear unit (ReLU) activation. Subsequently, weight normalization (Salimans and Kingma, 2016) is applied to MPD. By reshaping the input audio into 2D data instead of sampling periodic signals of audio, gradients from MPD can be delivered to all time steps of the input audio.
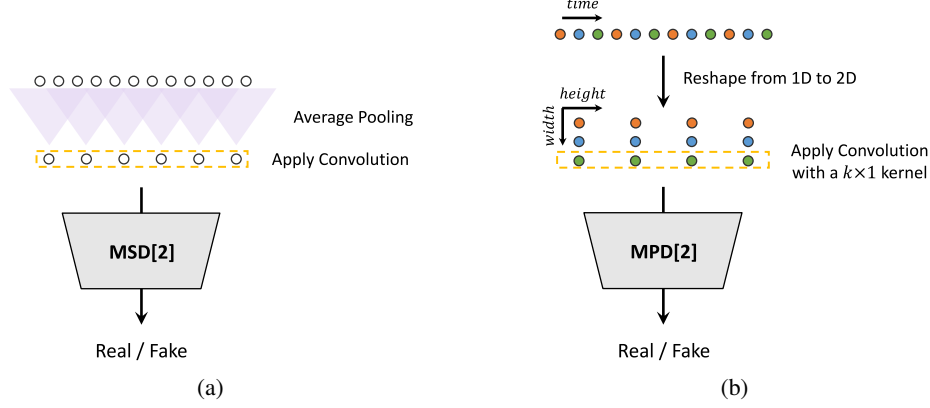
Figure 2: (a) The second sub-discriminator of MSD. (b) The second sub-discriminator of MPD with period 3.

**Multi-Scale Discriminator** Because each sub-discriminator in MPD only accepts disjoint samples, we add MSD to consecutively evaluate the audio sequence. The architecture of MSD is drawn from that of MelGAN (Kumar et al., 2019). MSD is a mixture of three sub-discriminators operating on different input scales: raw audio, ×2 average-pooled audio, and ×4 average-pooled audio, as shown in Figure 2a. Each of the sub-discriminators in MSD is a stack of strided and grouped convolutional layers with leaky ReLU activation. The discriminator size is increased by reducing stride and adding more layers. Weight normalization is applied except for the first sub-discriminator, which operates on raw audio. Instead, spectral normalization (Miyato et al., 2018) is applied and stabilizes training as it reported.

Note that MPD operates on disjoint samples of raw waveforms, whereas MSD operates on smoothed waveforms.

For previous work using multi-discriminator architecture such as MPD and MSD, Bińkowski et al. (2019)'s work can also be referred. The discriminator architecture proposed in the work has resemblance to MPD and MSD in that it is a mixture of discriminators, but MPD and MSD are Markovian window-based fully unconditional discriminator, whereas it averages the output and has conditional discriminators. Also, the resemblance between MPD and RWD (Bińkowski et al., 2019) can be considered in the part of reshaping input audio, but MPD uses periods set to prime numbers to discriminate data of as many periods as possible, whereas RWD uses reshape factors of overlapped periods and does not handle each channel of the reshaped data separately, which are different from what MPD is aimed for. A variation of RWD can perform a similar operation to MPD, but it is also not the same as MPD in terms of parameter sharing and strided convolution to adjacent signals. More details of the architectural difference can be found in Appendix C.

## 2.4 Training Loss Terms

**GAN Loss** For brevity, we describe our discriminators, MSD and MPD, as one discriminator throughout Section 2.4. For the generator and discriminator, the training objectives follow LS-GAN (Mao et al., 2017), which replace the binary cross-entropy terms of the original GAN objectives (Goodfellow et al., 2014) with least squares loss functions for non-vanishing gradient flows. The discriminator is trained to classify ground truth samples to 1, and the samples synthesized from the generator to 0. The generator is trained to fake the discriminator by updating the sample quality to be classified to a value almost equal to 1. GAN losses for the generator $G$ and the discriminator $D$ are defined as

$$\mathcal{L}_{Adv}(D; G) = \mathbb{E}_{(x,s)}\left[ (D(x) - 1)^2 + (D(G(s)))^2 \right] \tag{1}$$

$$\mathcal{L}_{Adv}(G; D) = \mathbb{E}_{s}\left[ (D(G(s)) - 1)^2 \right] \tag{2}$$

4

, where $x$ denotes the ground truth audio and $s$ denotes the input condition, the mel-spectrogram of the ground truth audio.

**Mel-Spectrogram Loss**    In addition to the GAN objective, we add a mel-spectrogram loss to improve the training efficiency of the generator and the fidelity of the generated audio. Referring to previous work (Isola et al., 2017), applying a reconstruction loss to GAN model helps to generate realistic results, and in Yamamoto et al. (2020)'s work, time-frequency distribution is effectively captured by jointly optimizing multi-resolution spectrogram and adversarial loss functions. We used mel-spectrogram loss according to the input conditions, which can also be expected to have the effect of focusing more on improving the perceptual quality due to the characteristics of the human auditory system. The mel-spectrogram loss is the L1 distance between the mel-spectrogram of a waveform synthesized by the generator and that of a ground truth waveform. It is defined as

$$\mathcal{L}_{Mel}(G) = \mathbb{E}_{(x,s)}\left[||\phi(x) - \phi(G(s))||_1\right] \tag{3}$$

, where $\phi$ is the function that transforms a waveform into the corresponding mel-spectrogram. The mel-spectrogram loss helps the generator to synthesize a realistic waveform corresponding to an input condition, and also stabilizes the adversarial training process from the early stages.

**Feature Matching Loss**    The feature matching loss is a learned similarity metric measured by the difference in features of the discriminator between a ground truth sample and a generated sample (Larsen et al., 2016, Kumar et al., 2019). As it was successfully adopted to speech synthesis (Kumar et al., 2019), we use it as an additional loss to train the generator. Every intermediate feature of the discriminator is extracted, and the L1 distance between a ground truth sample and a conditionally generated sample in each feature space is calculated. The feature matching loss is defined as

$$\mathcal{L}_{FM}(G; D) = \mathbb{E}_{(x,s)}\left[\sum_{i=1}^{T} \frac{1}{N_i}||D^i(x) - D^i(G(s))||_1\right] \tag{4}$$

, where $T$ denotes the number of layers in the discriminator; $D^i$ and $N_i$ denote the features and the number of features in the $i$-th layer of the discriminator, respectively.

**Final Loss**    Our final objectives for the generator and discriminator are as

$$\mathcal{L}_G = \mathcal{L}_{Adv}(G; D) + \lambda_{fm}\mathcal{L}_{FM}(G; D) + \lambda_{mel}\mathcal{L}_{Mel}(G) \tag{5}$$
$$\mathcal{L}_D = \mathcal{L}_{Adv}(D; G) \tag{6}$$

, where we set $\lambda_{fm} = 2$ and $\lambda_{mel} = 45$. Because our discriminator is a set of sub-discriminators of MPD and MSD, Equations 5 and 6 can be converted with respect to the sub-discriminators as

$$\mathcal{L}_G = \sum_{k=1}^{K}\left[\mathcal{L}_{Adv}(G; D_k) + \lambda_{fm}\mathcal{L}_{FM}(G; D_k)\right] + \lambda_{mel}\mathcal{L}_{Mel}(G) \tag{7}$$

$$\mathcal{L}_D = \sum_{k=1}^{K}\mathcal{L}_{Adv}(D_k; G) \tag{8}$$

, where $D_k$ denotes the $k$-th sub-discriminator in MPD and MSD.

## 3    Experiments

For fair and reproducible comparison with other models, we used the LJSpeech dataset (Ito, 2017) in which many speech synthesis models are trained. LJSpeech consists of 13,100 short audio clips of a single speaker with a total length of approximately 24 hours. The audio format is 16-bit PCM with a sample rate of 22 kHz; it was used without any manipulation. HiFi-GAN was compared against the best publicly available models: the popular mixture of logistics (MoL) WaveNet (Oord et al., 2018)

implementation (Yamamoto, 2018) [3] and the official implementation of WaveGlow (Valle, 2018b) and MelGAN (Kumar, 2019). We used the provided pretrained weights for all the models.

To evaluate the generality of HiFi-GAN to the mel-spectrogram inversion of unseen speakers, we used the VCTK multi-speaker dataset (Veaux et al., 2017), which consists of approximately 44,200 short audio clips uttered by 109 native English speakers with various accents. The total length of the audio clips is approximately 44 hours. The audio format is 16-bit PCM with a sample rate of 44 kHz. We reduced the sample rate to 22 kHz. We randomly selected nine speakers and excluded all their audio clips from the training set. We then trained MoL WaveNet, WaveGlow, and MelGAN with the same data settings; all the models were trained until 2.5M steps.

To evaluate the audio quality, we crowd-sourced 5-scale MOS tests via Amazon Mechanical Turk. The MOS scores were recorded with 95% confidence intervals (CI). Raters listened to the test samples randomly, where they were allowed to evaluate each audio sample once. All audio clips were normalized to prevent the influence of audio volume differences on the raters. All quality assessments in Section 4 were conducted in this manner, and were not sourced from other papers.

The synthesis speed was measured on GPU and CPU environments according to the recent research trends regarding the efficiency of neural networks (Kumar et al., 2019, Zhai et al., 2020, Tan et al., 2020). The devices used are a single NVIDIA V100 GPU and a MacBook Pro laptop (Intel i7 CPU 2.6GHz). Additionally, we used 32-bit floating point operations for all the models without any optimization methods.

To confirm the trade-off between synthesis efficiency and sample quality, we conducted experiments based on the three variations of the generator, $V1$, $V2$, and $V3$ while maintaining the same discriminator configuration. For $V1$, we set $h_u = 512$, $k_u = [16, 16, 4, 4]$, $k_r = [3, 7, 11]$, and $D_r = [[1, 1], [3, 1], [5, 1]] \times 3]$. $V2$ is simply a smaller version of $V1$, which has a smaller hidden dimension $h_u = 128$ but with exactly the same receptive fields. To further reduce the number of layers while maintaining receptive fields wide, the kernel sizes and dilation rates of $V3$ were selected carefully. The detailed configurations of the models are listed in Appendix A.1. We used 80 bands mel-spectrograms as input conditions. The FFT, window, and hop size were set to 1024, 1024, and 256, respectively. The networks were trained using the AdamW optimizer (Loshchilov and Hutter, 2017) with $\beta_1 = 0.8$, $\beta_2 = 0.99$, and weight decay $\lambda = 0.01$. The learning rate decay was scheduled by a 0.999 factor in every epoch with an initial learning rate of $2 \times 10^{-4}$.

# 4 Results

## 4.1 Audio Quality and Synthesis Speed

To evaluate the performance of our models in terms of both quality and speed, we performed the MOS test for spectrogram inversion, and the speed measurement. For the MOS test, we randomly selected 50 utterances from the LJSpeech dataset and used the ground truth spectrograms of the utterances which were excluded from training as input conditions.

For easy comparison of audio quality, synthesis speed and model size, the results are compiled and presented in Table1. Remarkably, all variations of HiFi-GAN scored higher than the other models. $V1$ has 13.92M parameters and achieves the highest MOS with a gap of 0.09 compared to the ground truth audio; this implies that the synthesized audio is nearly indistinguishable from the human voice. In terms of synthesis speed, $V1$ is faster than WaveGlow and MoL WaveNet. $V2$ also demonstrates similarity to human quality with a MOS of 4.23 while significantly reducing the memory requirement and inference time, compared to $V1$. It only requires 0.92M parameters. Despite having the lowest MOS among our models, $V3$ can synthesize speech 13.44 times faster than real-time on CPU and 1,186 times faster than real-time on single V100 GPU while showing similar perceptual quality with MoL WaveNet. As $V3$ efficiently synthesizes speech on CPU, it can be well suited for on-device applications.

---

[3]In the original paper (Oord et al., 2018), an input condition called linguistic features was used, but this implementation uses mel-spectrogram as the input condition as in the later paper (Shen et al., 2018).

Table 1: Comparison of the MOS and the synthesis speed. Speed of $n$ kHz means that the model can generate $n \times 1000$ raw audio samples per second. The numbers in () mean the speed compared to real-time.

| Model | MOS (CI) | Speed on CPU (kHz) | | Speed on GPU (kHz) | | # Param (M) |
|---|---|---|---|---|---|---|
| Ground Truth | 4.45 ($\pm$0.06) | $-$ | | $-$ | | $-$ |
| WaveNet (MoL) | 4.02 ($\pm$0.08) | $-$ | | 0.07 | ($\times$0.003) | 24.73 |
| WaveGlow | 3.81 ($\pm$0.08) | 4.72 | ($\times$0.21) | 501 | ($\times$22.75) | 87.73 |
| MelGAN | 3.79 ($\pm$0.09) | 145.52 | ($\times$6.59) | 14,238 | ($\times$645.73) | 4.26 |
| HiFi-GAN $V1$ | **4.36** ($\pm$0.07) | 31.74 | ($\times$1.43) | 3,701 | ($\times$167.86) | 13.92 |
| HiFi-GAN $V2$ | 4.23 ($\pm$0.07) | 214.97 | ($\times$9.74) | 16,863 | ($\times$764.80) | **0.92** |
| HiFi-GAN $V3$ | 4.05 ($\pm$0.08) | **296.38** ($\times$**13.44**) | | **26,169** ($\times$**1,186.80**) | | 1.46 |

## 4.2 Ablation Study

We performed an ablation study of MPD, MRF, and mel-spectrogram loss to verify the effect of each HiFi-GAN component on the quality of the synthesized audio. $V3$ that has the smallest expressive power among the three generator variations was used as a generator for the ablation study, and the network parameters were updated up to 500k steps for each configuration.

The results of the MOS evaluation are shown in Table 2, which show all three components contribute to the performance. Removing **MPD** causes a significant decrease in perceptual quality, whereas the absence of **MSD** shows a relatively small but noticeable degradation. To investigate the effect of **MRF**, one residual block with the widest receptive field was retained in each MRF module. The result is also worse than the baseline. The experiment on the **mel-spectrogram loss** shows that it helps improve the quality, and we observed that the quality improves more stably when the loss is applied.

To verify the effect of MPD in the settings of other GAN models, we introduced MPD in MelGAN. MelGAN trained with MPD outperforms the original one by a gap of 0.47 MOS, which shows statistically significant improvement.

We experimented with periods of powers of 2 to verify the effect of periods set to prime numbers. While the period 2 allows signals to be processed closely, it results in statistically significant degradation with a difference of 0.20 MOS from the baseline.

Table 2: Ablation study results. Comparison of the effect of each component on the synthesis quality.

| Model | MOS (CI) |
|---|---|
| Ground Truth | 4.57 ($\pm$0.04) |
| Baseline (HiFi-GAN $V3$) | 4.10 ($\pm$0.05) |
| w/o MPD | 2.28 ($\pm$0.09) |
| w/o MSD | 3.74 ($\pm$0.05) |
| w/o MRF | 3.92 ($\pm$0.05) |
| w/o Mel-Spectrogram Loss | 3.25 ($\pm$0.05) |
| MPD $p$=[2,4,8,16,32] | 3.90 ($\pm$0.05) |
| MelGAN | 2.88 ($\pm$0.08) |
| MelGAN with MPD | 3.35 ($\pm$0.07) |

## 4.3 Generalization to Unseen Speakers

We used 50 randomly selected utterances of nine unseen speakers in the VCTK dataset that were excluded from the training set for the MOS test. Table 3 shows the experimental results for the mel-spectrogram inversion of the unseen speakers. The three generator variations scored 3.77, 3.69, and 3.61. They were all better than AR and flow-based models, indicating that the proposed models

generalize well to unseen speakers. Additionally, the tendency of difference in MOS scores of the proposed models is similar with the result shown in Section 4.1, which exhibits generalization across different datasets.

Table 3: Quality comparison of synthesized utterances for unseen speakers.

| Model | MOS (CI) |
|---|---|
| Ground Truth | 3.79 ($\pm$0.07) |
| WaveNet (MoL) | 3.52 ($\pm$0.08) |
| WaveGlow | 3.52 ($\pm$0.08) |
| MelGAN | 3.50 ($\pm$0.08) |
| HiFi-GAN $V1$ | **3.77** ($\pm$0.07) |
| HiFi-GAN $V2$ | 3.69 ($\pm$0.07) |
| HiFi-GAN $V3$ | 3.61 ($\pm$0.07) |

Table 4: Quality comparison for end-to-end speech synthesis.

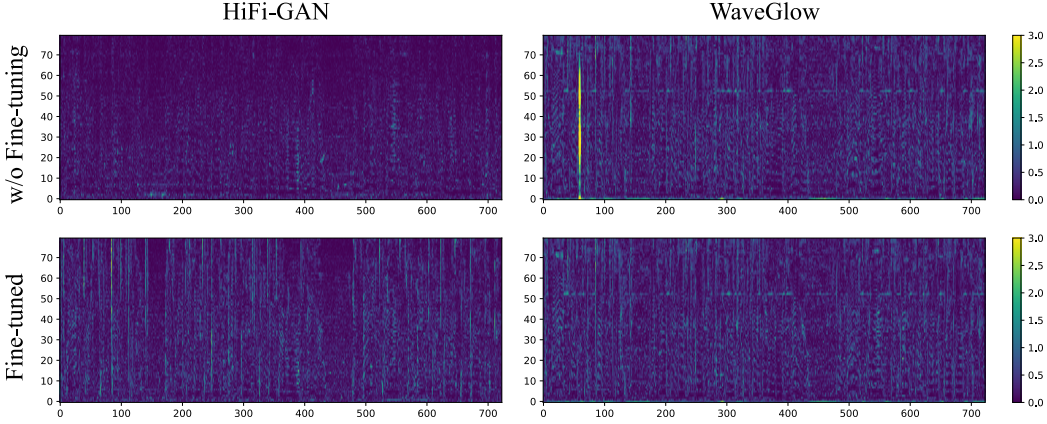| Model | MOS (CI) |
|---|---|
| Ground Truth | 4.23 ($\pm$0.07) |
| WaveGlow (w/o fine-tuning) | 3.69 ($\pm$0.08) |
| HiFi-GAN $V1$ (w/o fine-tuning) | 3.91 ($\pm$0.08) |
| HiFi-GAN $V2$ (w/o fine-tuning) | 3.88 ($\pm$0.08) |
| HiFi-GAN $V3$ (w/o fine-tuning) | 3.89 ($\pm$0.08) |
| WaveGlow (find-tuned) | 3.66 ($\pm$0.08) |
| HiFi-GAN $V1$ (find-tuned) | **4.18** ($\pm$0.08) |
| HiFi-GAN $V2$ (find-tuned) | 4.12 ($\pm$0.07) |
| HiFi-GAN $V3$ (find-tuned) | 4.02 ($\pm$0.08) |



Figure 3: Pixel-wise difference in the mel-spectrogram domain between generated waveforms and a mel-spectrogram from Tacotron2. Before fine-tuning, HiFi-GAN generates waveforms corresponding to input conditions accurately. After fine-tuning, the error of the mel-spectrogram level increased, but the perceptual quality increased.

## 4.4 End-to-End Speech Synthesis

We conducted an additional experiment to examine the effectiveness of the proposed models when applied to an end-to-end speech synthesis pipeline, which consists of *text to mel-spectrogram* and *mel-spectrogram to waveform* synthesis modules. We herein used Tacotron2 (Shen et al., 2018) to generate mel-spectrograms from text. Without any modification, we synthesized the mel-spectrograms using the most popular implementation of Tacotron2 (Valle, 2018a) with the provided pre-trained weights. We then fed them as input conditions into second stage models, including our models and WaveGlow used in Section 4.1.[4]

The MOS scores are listed in Table 4. The results without fine-tuning show that all the proposed models outperform WaveGlow in the end-to-end setting, while the audio quality of all models are unsatisfactory compared to the ground truth audio. However, when the pixel-wise difference in the mel-spectrogram domain between generated waveforms and a mel-spectrogram from Tacotron2 are

---

[4]MelGAN and MoL WaveNet were excluded from comparison for their differences in pre-processing such as frequency clipping. Mismatch in the input representation led to producing low quality audio when combined with Tacotron2.

investigated as demonstrated in Figure 3, we found that the difference is insignificant, which means that the predicted mel-spectrogram from Tacotron2 was already noisy. To improve the audio quality in the end-to-end setting, we applied fine-tuning with predicted mel-spectrograms of Tacotron2 in teacher-forcing mode (Shen et al., 2018) to all the models up to 100k steps. MOS scores of all the fine-tuned proposed models over 4, whereas fine-tuned WaveGlow did not show quality improvement. We conclude that HiFi-GAN adapts well on the end-to-end setting with fine-tuning.

## 5   Conclusion

In this work, we introduced HiFi-GAN, which can efficiently synthesize high quality speech audio. Above all, our proposed model outperforms the best performing publicly available models in terms of synthesis quality, even comparable to human level. Moreover, it shows a significant improvement in terms of synthesis speed. We took inspiration from the characteristic of speech audio that consists of patterns with various periods and applied it to neural networks, and verified that the existence of the proposed discriminator greatly influences the quality of speech synthesis through the ablation study. Additionally, this work presents several experiments that are significant in speech synthesis applications. HiFi-GAN shows ability to generalize unseen speakers and synthesize speech audio comparable to human quality from noisy inputs in an end-to-end setting. In addition, our small footprint model demonstrates comparable sample quality with the best publicly available autoregressive counterpart, while generating samples in an order-of-magnitude faster than real-time on CPU. This shows progress towards on-device natural speech synthesis, which requires low latency and memory footprint. Finally, our experiments show that the generators of various configurations can be trained with the same discriminators and learning mechanism, which indicates the possibility of flexibly selecting a generator configuration according to the target specifications without the need for a time-consuming hyper-parameter search for the discriminators.

We release HiFi-GAN as open source. We envisage that our work will serve as a basis for future speech synthesis studies.

## Acknowledgments

# References

Mikołaj Bińkowski, Jeff Donahue, Sander Dieleman, Aidan Clark, Erich Elsen, Norman Casagrande, Luis C Cobo, and Karen Simonyan. High fidelity speech synthesis with adversarial networks. *arXiv preprint arXiv:1909.11646*, 2019.

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. *arXiv preprint arXiv:1802.04208*, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

Keith Ito. The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`, 2017.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018.

Kundan Kumar, Rithesh Kumar, Thibault de Boissiere, Lucas Gestin, Wei Zhen Teoh, Jose Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C Courville. Melgan: Generative adversarial networks for conditional waveform synthesis. In *Advances in Neural Information Processing Systems 32*, pages 14910–14921, 2019.

Rithesh Kumar. descriptinc/melgan-neurips. `https://github.com/descriptinc/melgan-neurips`, 2019.

Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *International conference on machine learning*, pages 1558–1566. PMLR, 2016.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6706–6713, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

Michael Mathieu, Camille Couprie, and Yann LeCun. Deep multi-scale video prediction beyond mean square error. *arXiv preprint arXiv:1511.05440*, 2015.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*, 2018.

Aaron Oord, Yazhe Li, Igor Babuschkin, Karen Simonyan, Oriol Vinyals, Koray Kavukcuoglu, George Driessche, Edward Lockhart, Luis Cobo, Florian Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR, 2018.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Wei Ping, Kainan Peng, Andrew Gibiansky, Sercan O Arik, Ajay Kannan, Sharan Narang, Jonathan Raiman, and John Miller. Deep voice 3: Scaling text-to-speech with convolutional sequence learning. *arXiv preprint arXiv:1710.07654*, 2017.

Wei Ping, Kainan Peng, and Jitong Chen. Clarinet: Parallel wave generation in end-to-end text-to-speech. *arXiv preprint arXiv:1807.07281*, 2018.

Ryan Prenger, Rafael Valle, and Bryan Catanzaro. Waveglow: A flow-based generative network for speech synthesis. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3617–3621. IEEE, 2019.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016.

Jonathan Shen, Ruoming Pang, Ron J Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, Rj 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)*, pages 4779–4783. IEEE, 2018.

Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10781–10790, 2020.

Rafael Valle. Nvidia/tacotron2. `https://github.com/NVIDIA/tacotron2`, 2018a.

Rafael Valle. Nvidia/waveglow. `https://github.com/NVIDIA/waveglow`, 2018b.

Christophe Veaux, Junichi Yamagishi, Kirsten MacDonald, et al. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2017.

Ryuichi Yamamoto. wavenet vocoder. `https://github.com/r9y9/wavenet_vocoder/`, 2018.

Ryuichi Yamamoto, Eunwoo Song, and Jae-Min 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)*, pages 6199–6203. IEEE, 2020.

Bohan Zhai, Tianren Gao, Flora Xue, Daniel Rothchild, Bichen Wu, Joseph E Gonzalez, and Kurt Keutzer. Squeezewave: Extremely lightweight vocoders for on-device speech synthesis. *arXiv preprint arXiv:2001.05685*, 2020.

# Appendix A

## A.1. Details of the Model Architecture

The detailed architecture of the generator and MPD is depicted in Figure 4. The configuration of three variants of the generator is listed in Table 5.
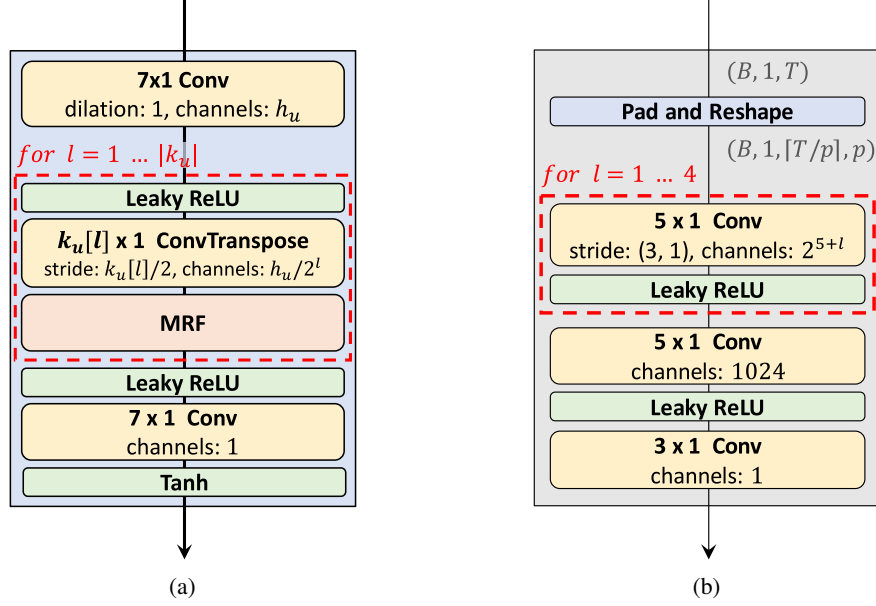


Figure 4: (a) The generator. (b) The sub-discriminator of MPD with period $p$.

Table 5: Hyper-parameters of three generator $V1$, $V2$, and $V3$.

| Model | $h_u$ | $k_u$ | $k_r$ | $D_r$ |
|---|---|---|---|---|
| | | | Hyper-Parameters | |
| $V1$ | 512 | [16, 16, 4, 4] | [3, 7, 11] | [[1, 1], [3, 1], [5, 1]] $\times 3$ |
| $V2$ | 128 | [16, 16, 4, 4] | [3, 7, 11] | [[1, 1], [3, 1], [5, 1]] $\times 3$ |
| $V3$ | 256 | [16, 16, 8] | [3, 5, 7] | [[1], [2]], [[2], [6]], [[3], [12]] |

Each ResBlock in the generators is a structure in which multiple convolution layers and residual connections are stacked. In the ResBlock of V1 and V2, 2 convolution layers and 1 residual connection are stacked 3 times. In the Resblock of V3, 1 convolution layer and 1 residual connection are stacked 2 times. Therefore, V3 consists of a much smaller number of layers than V1 and V2.

# Appendix B

## B.1. Periodic signal discrimination experiments

We conducted additional experiments similar to training a discriminator using a simple dataset to verify the ability of MPD to discriminate periodic signals. We have assumed that most of the signals generated by the generator are well made, with a very small amount of error signals. We generated 40,000 sinusoidal data of randomly selected frequency in the range of 1~8,000Hz, random phase and random energy. We experimented with 3 ratios to see the performance change according to the difference between the ratio of true and false label. We gave true label [99, 99.5, 99.9]% of the frequencies and false label the remaining frequencies. We added a projection layer to MPD and MSD, trained with these datasets, and compared the classification results of 8,000 unseen data. As this is binary classification, we set the ratio of true label to false label of validation data to 50:50 to facilitate comparison. We repeated this experiment 5 times to get the average, and the results are listed in Table 6. The results show that MPD is superior in discriminating periodic signals than MSD. And remarkably MPD can discriminate 0.1% of false frequencies with 85% accuracy.

Table 6: Comparison of classification accuracy for periodic signals.

| Model | True Label Ratio | | |
|---|---|---|---|
|  | 99.0% | 99.5% | 99.9% |
| MSD | 80.76% | 59.01% | 50.42% |
| MPD | 97.01% | 96.38% | 85.33% |

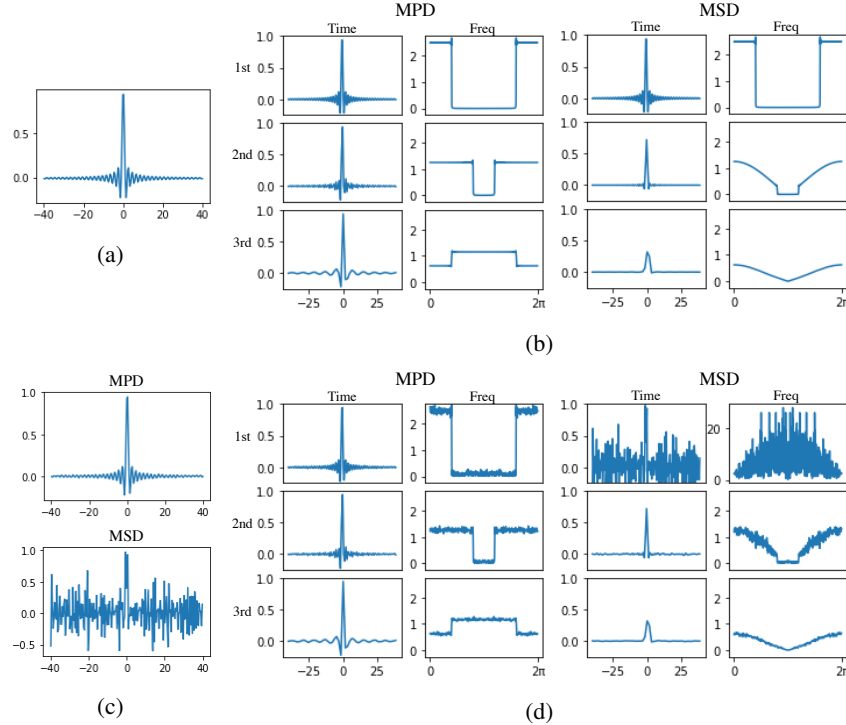## B.2. Frequency response analysis of input signals of discriminators



Figure 5: (a) The ground-truth sinc function. (b) The down-sampled input signals for sub-discriminators and their frequency responses. (c) The synthesized signals. (b) The down-sampled synthesized signals for sub-discriminators and their frequency responses.

We investigated the importance of modeling discriminators to capture periodic patterns through a toy example. In this example, we train two generators to mimic the sinc function each with MPD and MSD, respectively. we set the domain of sinc function to evenly spaced 1,000 numbers over an interval [-200, 200], and set the function values as the ground-truth signal, which is visualized in Figure 5a. To maintain the equal structure of MSD and MPD, we set the periods of MPD to [1, 2, 4]. All sub-discriminators of MPD and MSD are designed as feed-forward networks with the same structure. The generator is simply modeled with 1,000 parameters each corresponded to one point of the domain of the ground-truth signal.

Figure 5b shows input signals of sub-discriminators and the magnitude of their frequency responses. In the case of MPD, the frequency responses of input signals are not distorted except for aliasing. On the other hand, the input signals of MSD are getting smoother whenever down-sampling. It is due to the low-pass filtering of average pooling, which results into diminishing amplitudes in high frequencies.

When comparing the outputs of learned generators, the difference is more evident. Figure 5c shows the outputs of generators each of which is trained with MSD and MPD, respectively. In the case of MPD, the synthesized signal is similar to the ground truth, while in the case of MSD, the synthesized signal is noisy.

The reason of the sub-optimal outcomes of MSD can be found in average pooling of the down-sampling procedure. Figure 5d shows the down-sampled synthesized signals for sub-discriminators and the magnitude of their frequency responses. When it comes to down-sampling synthesized signals for the second and third sub-discriminators of MSD, the result seems similar to that of the ground-truth signal. It indicates that even though there is huge difference between the ground-truth and generated signals, the average-pooled input signals of some sub-discriminators of MSD can be similar. On the other hand, MPD shows similar results to the ground truth regardless of down-sampling factors. In conclusion, we can see that MPD captures more periodic patterns of an input signal than MSD, and capturing periodic patterns is important to model signals.

## Appendix C

### C.1. Details of architectural difference between RWD and MPD

When the first layer of RWD is the grouped 1d convolution, the convolution layer of RWD becomes similar to the first 2d convolution with k×1 kernels of MPD, but there is still a difference as RWD would not share weights across each group. Moreover, RWD operates on an entire input sequence regardless of input reshaping, while MPD only operates on down-sampled, or evenly spaced, samples from the input sequence. In that regard, it can be seen that RWD uses additional parameters to mix representations of down-sampled samples at each layer, in other words, MPD shares more parameters at each layer than RWD.