Interspeech 2022
18-22 September 2022, Incheon, Korea

A Compact Transformer-Based GAN Vocoder

Chenfeng Miao, Ting Chen*, Minchuan Chen, Jun Ma, Shaojun Wang, Jing Xiao*

Ping An Technology

miao_chenfeng@126.com, tingc95@outlook.com

Abstract

Recent work has shown that self-attention module in Transformer architecture is an effective way of modeling natural languages and images. In this study, we propose a novel approach for waveform synthesis utilizing Self-Attention Network (SAN). To the best of our knowledge, Transformer architectures or SANs have not yet been successfully applied in high-fidelity waveform generation. The main challenge in applying SANs to waveform generation tasks lies in its quadratic growth of the computational complexity with respect to the input sequence length, which makes it unsuitable for high sample-rate waveform generation. To solve this problem, we apply dilated sliding window to vanilla SAN. This technique enables our model to have a large receptive field while maintaining linear computational complexity and a small footprint. We experimentally show that the proposed model achieves smaller model size, while producing audio samples with comparable speech quality in comparison with the best publicly available model. In particular, our small footprint model has only 0.57M parameters and can generate 22.05kHz high-fidelity audio 113× faster than real-time on a NVIDIA V100 GPU without engineered inference kernels.

Index Terms: Transformer, neural vocoder

1. Introduction

As an important part of human-machine interface, text-to-speech (TTS) technology is currently a research hotspot and has a wide range of applications in the industry. The main pipeline of the TTS system needs to train two models separately, including an acoustic model that generate acoustic features (e.g. Mel-spectrograms [1]) from text, and a vocoder that convert acoustic features into raw waveform. Recently, several fully end-to-end TTS models [2, 3, 4, 5] are proposed to generate waveform directly from text, but they either generate lower quality samples or take more time to train than separately training method.

Vocoders can be roughly divided into two categories: the vocoders used in statistical parametric speech synthesis (SPSS)[6, 7], and the neural network-based vocoders. SPSS vocoders have dominated waveform generation for decades, until neural vocoders made a huge breakthrough in synthesizing human-like audio [8] and have been widely used since then. According to the model they use, neural vocoders can be classified into autoregressive vocoders, flow-based vocoders, GAN-based vocoders, VAE-base vocoders and Diffusion-based vocoders. Autoregressive models [8, 9] are the first and most expressive model in waveform generation, but they suffer from slow inference speed due to their recurrent architecture. Later some works introduce flow-based methods to accelerate inference. Parallel WaveNet [10] and Clarinet [11] introduce inverse autoregressive flow (IAF) to enable parallel synthesis, but they require teacher distillation from autoregressive models and still requires large computation. WaveGlow [12] and FloWaveNet [13] avoid distillation and can generate high-fidelity waveforms in real-time by incorporating non-causal WaveNet layers with normalizing flows. However, they require a significant amount of GPU memory as well as a great amount of inference latency. There are also several works of VAE-based vocoders and diffusion-base vocoders. However, among all neural vocoders, GAN-based ones [14, 15, 16, 17] show remarkable achievements, by its ability to generate high quality voice while synthesizing orders of magnitude faster.

Most GAN-based vocoders use dilated convolution to increase the receptive field to model the long-dependency in waveform sequence. This idea was firstly used in signal processing and image segmentation, and then adopted by audio generation. However, many recent works have shown that Transformer [18] and its variants have great potential to replace convolution networks as backbone architecture in many fields, including NLP [19, 20] and CV [21, 22] models. Despite the huge success in these areas, there are relatively few applications of using Transformers to model audio data. TransformerTTS [23] uses Transformer architecture for mel-spectrogram generation, Audiomer [24] proposes a convolutional Transformer network which achieves state-of-the-art performance in end-to-end Keyword Spotting task. However, to the best of our knowledge, there is no successful application of Transformer in raw waveform generation task.

The main difficulty of incorporating SAN for raw waveform tasks lies in its inability of scaling to long sequences. In a high-fidelity speech synthesize setting, the model needs to synthesize with up to 22,050 samples per second and up to 16-bit fidelity. However, SAN's computational complexity is quadratic with respect to input sequence length. As a result, using vanilla Transformer architecture directly on raw waveform is infeasible. In this work, we try to extend the Transformer architecture to waveform generation task, and propose a Transformer-based GAN vocoder. Our main contributions are summarized as follows:

1. We propose a convolution-free waveform generator, which is mainly built upon self-attention modules. To the best of our knowledge, we are the first to apply Transformer in high-fidelity raw waveform generation task.

2. We introduce several design choices to reduce the computational complexity of self-attention module, making the network efficient to train and synthesis.

3. The proposed model enjoys small footprint, fast synthesize speed, and produces audio samples with comparable speech quality in comparison with the best publicly available model [17].

4. We conduct several experiments to demonstrate how to trade-off between speech quality and size of model parameters. This would be an important guide for practitioners to balance between cost and performance when building their own TTS systems.

*equation contribution

10.21437/Interspeech.2022-11254

2. Related Works

2.1. GAN Vocoders

Although likelihood-based models, such as auto-regressive models [8, 9, 11], flow-based models [12], diffusion-based models [25, 26], contribute a lot to the development of audio generation tasks, it is nevertheless well-established that the implicit likelihood models, such as Generative Adversarial Networks (GANs) [27], are one of the most dominant models that can produce high-fidelity raw audio with high efficiency. There are several GAN vocoders [28, 17, 29, 30] proposed recently. Among them, [17] achieves the best audio quality, [29] outperforms other counterpart models in synthesis speed. [15] introduces a robust model that can generate waveform of multiple speakers. Despite the differences of these models in network architectures, the generators of all these models are composed of Convolution Neural Networks (CNNs). In this work, we take a challenge of using Transformer blocks to build the generator, which allows for a more parameter efficient implementation.

2.2. Transformer Architecture

The Transformer architecture, first proposed in [18], has become the de-facto standard for NLP tasks. Recently, Transformer and its variants have made a great influence in computer vision (CV) community. The success of Transformer architecture in the field of NLP and CV inspires us to apply it to audio tasks. Some prior works focus on generating high-fidelity mel-spectrograms from text sequence using Transformer architecture [23] or feed-forward transformer blocks [31, 32]. However, to the best of our knowledge, there is no prior work that using Transformer to build a neutral vocoder.

The Self-Attention Network (SAN), as the most important module in Transformer architecture, has several properties that make it good fit for audio tasks in comparison of CNNs: (1) the ability to capture long-range dependencies for high-frequency audio data with large sequence length, (2) parameter-independent scaling of receptive field size for building lightweight models, and (3) content-aware interactions as opposed to content-independent interactions of CNNs. However, the vanilla SAN connects all input positions, requiring a computational complexity of $O(n^2)$, where n is the sequence length. As such, SANs are very fast for short-sequence tasks like machine translation, but extremely slow and inefficient to train for long-sequence tasks like waveform generation. To address this limitation, some prior works propose to compute sparse attention instead of a dense one to accelerate the Transformer model [33]. Some works[34, 35, 22, 21] utilize a local window (with a window size of w) when computing dot-product attention, which in theory reduces the self-attention complexity to $O(n \cdot w)$. In this work, we investigate several techniques to improve the model efficiency. To this end, we propose a transformer-based neutral vocoder. Our model enjoys small footprint, fast synthesize speed, and produces audio samples with comparable speech quality in comparison with counterpart models.

3. Proposed architecture

Similar to MelGAN [28] and HiFi-GAN [17], our overall architecture is based on GAN and using mel-spectrogram as input to generate raw waveform. For discriminator, we choose the multi-resolution discriminating(MRD) framework whose performance is experimentally confirmed by [30]. Our implementation of discriminator is identical to HiFi-GAN [17]. However, our design of generator is different. As shown in Fig. 1, the input is firstly processed by a linear layer and a Transformer block, then fed into a stack of N similar blocks. Each block contains one upsampling layer and k Transformer blocks. Finally, a linear layer is used to project hidden vector to waveform output. The upsampling is done by a stack of transposed convolutional layers. Each Transformer blocks consists of a multi-head self-attention layer and a position-wise feed-forward linear layer. A residual connection and a LayerNorm (LN) layer are applied after each module of Transformer block. More details about the computation of multi-head self-attention are shown in the rest of this section.

3.1. Multi-head Self-Attention

Multi-head self-attention layer consists of multiple attention heads working in parallel. Each attention head applies the attention mechanism to its own input. Assuming X is the input sequence, a multi-head self-attention is computed as:

$$\mathrm{Head^i} = \mathrm{Attention}(\mathrm{XW^i_Q}, \mathrm{XW^i_K}, \mathrm{XW^i_V}) \tag{1}$$

$$\mathrm{Y} = \mathrm{Concat}(\mathrm{Head^1}, ..., \mathrm{Head^h})\mathrm{W_O} + \mathrm{X} \tag{2}$$

where $\mathrm{W_Q}, \mathrm{W_K}, \mathrm{W_V}, \mathrm{W_O}$ are learned linear transformations shared across all locations. Y is the output sequence with the sample size as input X. The score function of scaled dot-product attention can be formulated as:

$$\mathrm{Attention}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \mathrm{SoftMax}(\frac{\mathrm{Q} \cdot \mathrm{K^T}}{\sqrt{\mathrm{d_k}}}) \cdot \mathrm{V} \tag{3}$$

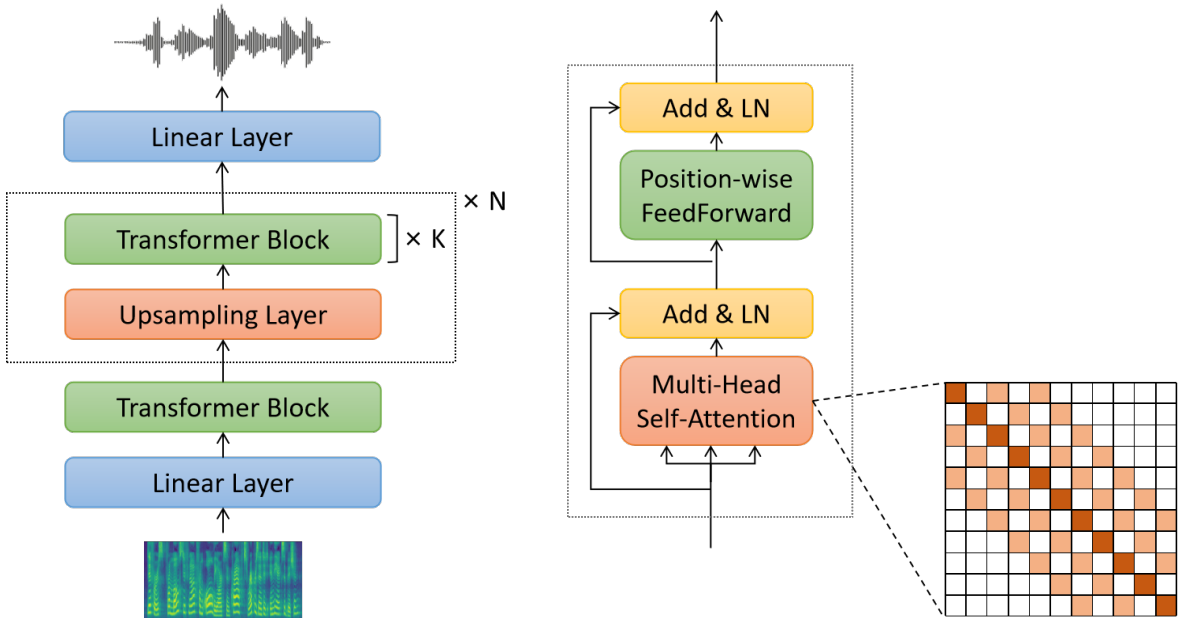
Where $\mathrm{d_k}$ is the dimensionality of K. The rest part of this subsection will discuss more details about the computation of scaled dot-product attention.

3.1.1. Sliding window

As aforementioned, the computational complexity of vanilla SAN is quadratic with respect to the sequence length, so it is inefficient when modeling long sequences. To reduce computational complexity, a natural idea is to compute attention within a fixed-size local window. Assuming w is the window size, the complexity of window-based self-attention is reduced from $O(n^2)$ to $O(n \cdot w)$, which is linear to the sequence length. There are two popular window-based attentions. One is using shifted non-overlapping windows [22] and the other is using sliding windows [35]. In image classification tasks, [22] shows that shifted non-overlapping window-based attention outperforms sliding window-based attention through their experimental result. However, to the best of our knowledge, no similar conclusions have been found in our previous audio tasks. Instead, we have come to a completely opposite conclusion through our experiments. Following equation shows the proposed windowing operation. The sliding windowing $\mathrm{SW}(\cdot)$ is applied to both the key and query vector:

$$\mathrm{Attention}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \mathrm{SoftMax}(\frac{\mathrm{Q} \cdot \mathrm{SW(K)^T}}{\sqrt{\mathrm{d_k}}}) \cdot \mathrm{SW(V)} \tag{4}$$

The window of i^{th} element of the input vector is from $(i - \frac{w}{2})$ to $(i + \frac{w}{2})$, centered around i. Note that sliding windows are used similar to that of CNNs. However, SANs can scale to large window size without increasing the number of model parameters.



(a) Model architecture *(b) Transformer block*

Figure 1: *Overall model architecture. (a) Model architecture. (b) Transformer block implemented with dilated sliding windows.*

3.1.2. Dilated Sliding window

The sampling rate of the audio data, usually between 8khz (telephone channel) and 48khz (high-fidelity audio). Differs from natural language generation tasks, waveform synthesis models need to generate longer sequences (millions of samples instead of dozens of words). It is quite challenge for neural networks to learn long-term dependencies, so window-based technique is always applied in audio-related tasks. This is quite reasonable since audio's locally invariant nature. It is well-known that vanilla SAN is good at capturing long-term dependency. But since we compute attention in sliding windows, the reception field size is limited by the window size. In prior works [8, 17], convolutions with dilated kernels are widely used to increase model's reception field. Inspired by dilated convolutions, we compute the attention inside dilated sliding windows. This technique allows SANs to increase the reception field by a constant scale factor, while not increasing the computation cost. The computational complexity and reception field of proposed methods are shown in Table 1. As can be seen, the reception field grows linearly with the dilated factor d, while the complexity remains $O(n \cdot w)$.

3.1.3. Relative position bias

Since the attention mechanism is position-insensitive, in order to make use of the order information of the sequence, some position information needs to be injected into the sequence when computing self-attention. In this work, we follow [22] by including a relative position bias to each attention head:

$$\mathrm{Attention}(\mathrm{Q}, \mathrm{K}, \mathrm{V}) = \mathrm{SoftMax}(\frac{\mathrm{Q} \cdot \mathrm{K}^{\mathrm{T}}}{\sqrt{\mathrm{d_k}}} + \mathrm{B}) \cdot \mathrm{V} \quad (5)$$

where $B \in \mathcal{R}^w$ is the relative position bias, which is a learned variable. w is the window size. We initialize B with zero vectors in our experiments. In complementary experiments, we employed sine and cosine functions proposed in vanilla Transformer [18] as the absolute positional embedding, and their performance is nearly identical. More details can be found in section 4.4.

Table 1: *Computational complexity and receptive field of different methods. n is the sequence length, w is the window size, d is the dilated factor*

Method	Complexity	Receptive field
SAN	$O(n^2)$	n
SAN + Sliding Window	$O(n \cdot w)$	w
SAN + Dilated Sliding Window	$O(n \cdot w)$	$d \cdot w$

4. Experiment

4.1. Datasets

All of our experiments are conducted on the LJ-Speech dataset [36], which consists of 13,100 audio clips of a single female speaker with a total length of approximately 24 hours. The audio format is 16-bit PCM with a sampling rate of 22.05kHz. We use an 80-band mel-spectrogram of the original audio as input to the generator. The FFT size, hop size and window size are 1024, 256, 1024 respectively.

4.2. Model configurations

We firstly use a linear layer to convert 80-band mel-spectrogram into a hidden representation of h_{init} dimension. 4 transposed convolutional layers followed by leaky-relu activations are employed to gradually upsample the temporal dimension of input sequence by factors of 8, 8, 2, 2. The number of channels per upsampling layer is reduced by half. Each upsampling layer is followed by a stack of 3 Transformer blocks with dilations of 1,

3, 5 respectively. The hidden dimension of the multi-head self-attention layer and feed-forward linear layer are set to $8 \times \frac{h}{4}$ and $2 \times h$ respectively, where h is the input dimension, 8 is the number of head. The window sizes of all sliding windows are set to 5. The final linear layer with tanh activation produces the output waveform. We provide 2 variants of our model, a large model with initial hidden dimensionality of $h_{init} = 512$ and a small model with $h_{init} = 128$. We use AdamW optimizer with $\beta_1 = 0.85, \beta_2 = 0.99$, and the weight decay of 0.01. The initial learning rate is 10^{-4} and decays by 0.999 every epoch. The batch size is set to 24 on 2 V100 GPU for the small model and 16 for the large model. The large model converges at 650k^{th} training step while the small model converges at 900k^{th} training step.

The baseline systems are two most popular GAN vocoders: HiFi-GAN [17] and MelGAN [28]. The official project repository of Hifi-GAN [1] and MelGAN [2] with default configurations are utilized. Audio samples of the proposed models and baseline models are available at `https://mcf330.github.io/IS2022audiosample/`.

4.3. Audio Quality and Synthesis Speed

We perform the Mean Opinion Score (MOS) test to evaluate speech quality. 20 utterances were randomly selected from the test set and then synthesized using the proposed models and the baseline model. The MOS ranking, model size and synthesis speed are shown in Table 3 and Table 2, respectively. It can be seen that the proposed large model achieves comparable MOS ranking to the state-of-the-art HiFi-GAN V1 model while requiring fewer parameters. The proposed small model has 0.57M parameters, and can synthesize 22.05 kHZ high-fidelity speech $113\times$ faster than real time, which is competitive to its counterparts, HiFi-GAN V2 and MelGAN. We notice that there's a decline of inference speed comparing with CNN counterparts for our models. The main reason is that we use dilated convolutional kernels to compute the attention, which leads to inefficient use of GPU memory. Therefore, we believe that our inference speed can be significantly improved by using some customized GPU kernels as reported in [33].

Table 2: *Model size and synthesis speed for different models. The model is implemented in PyTorch without any customized GPU kernels. We run synthesis on a single Tesla V100 GPU. We select 20 audio clips for synthesis speed evaluation, and synthesis each sentence 20 times to obtain the average synthesis speed. When calculating the compositing speed, we exclude the time overhead of transferring data between the CPU and GPU.*

Model family	Synthesis Speed (in kHZ)	Number of parameters (in millions)
HiFi-GAN V1 [17]	3,669	13.94
HiFi-GAN V2 [17]	16,702	0.92
MelGAN [28]	14,200	4.26
Ours (small)	2,491	**0.57**
Ours (large)	1,322	9.01

[1] `https://github.com/jik876/hifi-gan`

[2] `https://github.com/descriptinc/melgan-neurips`

Table 3: *The MOS with 95% confidence intervals for different methods on LJ-Speech data set. Higher is better.*

Model family	MOS (CI)
HiFi-GAN V1 [17]	4.25 ± 0.11
HiFi-GAN V2 [17]	4.15 ± 0.18
MelGAN [28]	3.98 ± 0.10
Ours (small)	4.13 ± 0.21
Ours (large)	4.22 ± 0.17
Ground Truth	$\mathbf{4.45 \pm 0.28}$

Table 4: *MOS with 95% confidence intervals of ablation studies on LJ-Speech dataset. Each ablation study involves removing some components of the proposed network.*

Model family	MOS (CI)
Baseline	4.05 ± 0.22
− Sliding Window	3.01 ± 0.33
− Dilation	3.94 ± 0.21
− Relative attention bias	3.92 ± 0.21

4.4. Ablations

We performed the following ablation analyses on the large model using MOS evaluation: sliding window vs shifted non-overlapping window; dilated attention kernels vs. dense attention kernels; relative position bias vs. absolute positional embedding. Ideally, we would like to train a model with vanilla self-attention layers for thorough comparison with the proposed model. However, we found that models using vanilla self-attention layers are not feasible to train due to GPU memory constraints.

we compare the performance of these ablations relative to the proposed model at 100 epochs of training, which was not enough for these models to converge, but far enough to see their relative performance differences. The results of the ablation study are shown in Table 4. We can see that all the design choices we proposed help to improve the MOS ranking. The MOS value drops significantly after removing the sliding window, Whereas the absence of dilation and relative position bias shows a slight but noticeable degradation.

5. CONCLUSION

In this work, we extend prior works on text and image, demonstrating that self-attention-based models can also operate effectively on raw waveform generation. Our model achieves comparable audio quality against convolution-based counterparts. Qualitative results show that our model is very lightweight, and very fast at inference time. We hope that our generator can be a new replacement to parameter-heavy alternatives in other audio-related tasks. Future work would focus on integrating some Transformer-based acoustic models [23], to build an end-to-end Transformer-based TTS model.

6. References

[1] T. Fukada, K. Tokuda, T. Kobayashi, and S. Imai, "An adaptive algorithm for mel-cepstral analysis of speech." in *ICASSP*, vol. 92. Citeseer, 1992, pp. 137–140.

[2] J. Donahue, S. Dieleman, M. Bińkowski, E. Elsen, and K. Simonyan, "End-to-end adversarial text-to-speech," *ICLR*, 2020.

[3] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech 2: Fast and high-quality end-to-end text to speech," *ICLR*, 2021.

[4] C. Miao, L. Shuang, Z. Liu, C. Minchuan, J. Ma, S. Wang, and J. Xiao, "Efficienttts: An efficient and high-quality text-to-speech architecture," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7700–7709.

[5] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *International Conference on Machine Learning*. PMLR, 2021, pp. 5530–5540.

[6] H. Kawahara, "Straight, exploitation of the other aspect of vocoder: Perceptually isomorphic decomposition of speech sounds," *Acoustical science and technology*, vol. 27, no. 6, pp. 349–353, 2006.

[7] M. Morise, F. Yokomori, and K. Ozawa, "World: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[8] 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," in *The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016*, 2016.

[9] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. van den Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient Neural Audio Synthesis," in *ICLR*, 2018.

[10] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart *et al.*, "Parallel WaveNet: Fast high-fidelity speech synthesis," in *Proc. PMLR*, 2018.

[11] W. Ping, K. Peng, and J. Chen, "ClariNet: Parallel Wave Generation in End-to-End Text-to-Speech," in *ICLR*, 2019.

[12] R. Prenger, R. Valle, and B. Catanzaro, "WaveGlow: A Flow-Based Generative Network For Speech Synthesis," in *ICASSP*, 2019.

[13] S. Kim, S. Lee, J. Song, J. Kim, and S. Yoon, "FloWaveNet: A generative flow for raw audio," in *ICML*, 2019.

[14] 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*. IEEE, 2020.

[15] W. Jang, D. Lim, and J. Yoon, "Universal MelGAN: A Robust Neural Vocoder for High-Fidelity Waveform Generation in Multiple Domains," in *ICASSP*, 2021.

[16] E. Song, R. Yamamoto, M.-J. Hwang, J.-S. Kim, O. Kwon, and J.-M. Kim, "Improved parallel wavegan vocoder with perceptually weighted spectrogram loss," in *SLT*, 2021.

[17] J. Kong, J. Kim, and J. Bae, "HiFi-GAN: Generative Adversarial Networks for Efficient and High Fidelity Speech Synthesis ," in *NeurIPS*, 2020.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *NACCL*, 2019.

[20] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language Models are Unsupervised Multitask Learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[21] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," *ICLR*, 2021.

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *arXiv preprint arXiv:2103.14030*, 2021.

[23] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural Speech Synthesis with Transformer Network," in *AAAI*, 2019.

[24] S. K. Sahu, S. Mitheran, J. Kamdar, and M. Gandhi, "Audiomer: A Convolutional Transformer for Keyword Spotting," *arXiv preprint arXiv:2109.10252*, 2021.

[25] N. Chen, Y. Zhang, H. Zen, R. J. Weiss, M. Norouzi, and W. Chan, "WaveGrad: Estimating Gradients for Waveform Generation," in *ICLR*, 2021.

[26] Z. Kong, W. Ping, J. Huang, K. Zhao, and B. Catanzaro, "DiffWave: A Versatile Diffusion Model for Audio Synthesis," in *ICLR*, 2021.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in *Advances in neural information processing systems*, 2014.

[28] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brebisson, Y. Bengio, and A. Courville., "MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis," in *NeurIPS*, 2019.

[29] G. Yang, S. Yang, K. Liu, P. Fang, W. Chen, and L. Xie, "Multi-band MelGAN: Faster Waveform Generation for High-Quality Text-to-Speech," in *Interspeech*, 2020.

[30] J. You, D. Kim, G. Nam, G. Hwang, and G. Chae, "GAN Vocoder: Multi-Resolution Discriminator Is All You Need," in *Interspeech*, 2021.

[31] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "FastSpeech: Fast, Robust and Controllable Text to Speech," in *Advances in Neural Information Processing Systems*, 2019.

[32] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "FastSpeech 2: Fast and High-Quality End-to-End Text to Speech ," in *International Conference on Learning Representations*, 2021.

[33] R. Child, S. Gray, A. Radford, and I. Sutskever, "Generating Long Sequences with Sparse Transformers," *arXiv preprint arXiv:1904.10509*, 2019.

[34] M. Zaheer, G. Guruganesh, K. A. Dubey, J. Ainslie, C. Alberti, S. Ontanon, P. Pham, A. Ravula, Q. Wang, L. Yang *et al.*, "Big bird: Transformers for longer sequences," in *NeurIPS*, 2020.

[35] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The Long-Document Transformer," *arXiv preprint arXiv:2004.05150*, 2020.

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