End-to-End Speech Synthesis with Generative Adversarial Networks

Vincent Liu
Stanford University
vliu@cs.stanford.edu

June 10, 2022

Abstract

Realistic speech synthesis through generative adversarial networks has garnered significant attention in recent years, primarily in the context of vocoder models that convert spectrograms into audio waveforms. However, the development of fully end-to-end text-to-speech systems remains challenging, with only a few attempts currently documented in the literature. Our contribution leverages monotonic alignment search to address the challenges of unaligned input-output sequences, enabling robust phoneme-to-audio mapping in a fully differentiable manner. The generator incorporates a Transformer-based encoder-decoder generator and hierarchical discriminators that operate on both raw audio waveforms and mel spectrograms. Our approach, which fuses the best practices across end-to-end text-to-speech literature, achieves audio quality on par with popular approaches that rely on spectrogram stopgaps. We further demonstrate the symbiotic relationship that this end-to-end approach has with pretrained natural language foundation models and extra supervision derived from nonparametric methods. The implementation for this work can be found at https://github.com/vliu15/adversarial-tts.

1 Introduction

Synthesizing realistic and natural speech remains one of the most compelling challenges in artificial intelligence. Recent advances in generative models, particularly generative adversarial networks (GANs), have catalyzed significant progress in the field of text-to-speech (TTS) synthesis [3, 1, 4, 22]. Traditionally, the TTS pipeline involves a multi-stage process, where intermediate representations, such as spectrograms, act as stopgaps between text processing and audio waveform generation [10, 17, 23, 18]. This multi-step paradigm, while effective, introduces cascading errors, increases inference latency, and constrains the flexibility and adaptability of TTS systems. Fully end-to-end TTS systems, which directly map textual input to audio waveforms, represent an emerging direction that holds the promise of overcoming these limitations by streamlining the pipeline and enhancing the naturalness of synthesized speech [4, 16, 24].

Despite their potential, end-to-end TTS systems remain underexplored, with most existing solutions struggling to address key challenges. One primary obstacle is the alignment problem: the need to effectively match phonetic or linguistic features from text to temporal acoustic features in the audio. This problem exists in tex-to-spectrogram models, but the one-to-many problem is exacerbated when modeling raw audio waveforms directly, as a single phoneme can correspond to thousands of audio samples. In traditional systems, explicit alignment mechanisms, such as forced aligners or external duration models, are often used to manage this correspondence [17, 28]. However, these methods introduce reliance on additional models, breaking the differentiability of the pipeline and complicating training.

Another challenge lies in the generation of high-quality waveforms. GAN-based methods have demonstrated remarkable success in vocoders [15, 11], where the focus is on converting spectrograms to waveforms. However, these methods condition on frequency inputs like spectrograms, and so the issues of mode collapse, difficulty in generating fine-grained temporal details, and the lack of robust adversarial TTS objectives are much more difficult. These limitations are further compounded by the need to model both short-term spectral features and long-term prosodic variations, which are critical for capturing the natural rhythm, intonation, and expressiveness of human speech.

In this work, we aim to bridge these gaps by developing a novel end-to-end TTS system that combines the strengths of monotonic alignment search [10] with recent advancements in adversarial training [4, 11]. Monotonic alignment search provides a robust mechanism for phoneme-

to-audio mapping by dynamically aligning input-output sequences in a fully differentiable manner. Unlike traditional forced aligners, this approach is end-to-end differentiable, enabling joint optimization of all components. Our work makes the following contributions:

1. We consolidate and extend best practices from the TTS literature, offering a unified framework that achieves high-quality audio synthesis while maintaining the simplicity and efficiency of an end-to-end design.
2. We demonstrate on LJSpeech dataset [9] that our system achieves audio quality comparable to established methods that rely on intermediate representations.
3. We ablate non-parameteric supervision and leverage unsupervised pretrained modules to show how to bootstrap the pure end-to-end setup symbiotically with other scaling paradigms in Table 1.

2 Related Work

The current TTS literature can be broken down into spectrogram predictors, vocoders, and end-to-end methods that aim to learn these two components simultaneously.

Spectrogram Predictors. Spectrogram-based TTS systems, which transform text into spectrograms, have dominated TTS research for many years, as these could plug into non-parametric spectrogram inversion methods to reconstruct audio reasonably well [5]. These methods often rely on text conversion to phonemes with the CMU Pronouncing Dictionary [19], as phonemes contain more comprehensive acoustic cues than characters and letters do. To align textual and spectrogram sequences, [23, 18] introduce attention mechanisms to align textual and acoustic sequences, enabling high-quality spectrogram generation. These systems rely on vocoders to synthesize waveforms from spectrograms [20]. Despite their success, spectrogram-based approaches suffer from inefficiencies and cascading errors due to the multi-stage pipeline. Other works address these issues by explicitly modeling duration and prosody, making the spectrogram generation faster and more robust [17, 16]. However, explicitly predicting the alignment with a duration predictor module relies on external annotation of the duration of each phoneme, which further introduces sources of error. Flow-based models like [10] further improve the differentiability of the pipeline by integrating monotonic alignment search for sequence mapping. [12] also explore adversarial methods for modeling spectrogram from text.

Vocoders. Vocoder models invert spectrograms into high-quality waveforms. Spectrograms are frequency-space representations of audio signals, generated from short-time Fourier transformations. As a result, they are downsampled but aligned temporally with the original audio waveform, making the inversion process easier than their generation from text input. Early vocoding approaches A popular vocoder is WaveNet [20], an autoregressive generative model. Because of the high frequency of audio samples, autoregressive sampling is extremely slow and most TTS applications need to be real-time. To parallelize audio generation, [21] distills WaveNet into a student network that is fully parallel and [26] adapts the WaveNet architecture to train a fully parallel GAN. However, these parallel adaptations of WaveNet do not usually exceed the quality of the original autoregressive WaveNet itself. Another paradigm of parallel spectrogram inversion relies on deep normalizing flow networks, which map a standard Gaussian volume into another probabilistic volume of audio waveforms [15, 27]. However, these networks often have to be quite deep in order to learn this complex probability density mapping and end up being quite slow in order to be at the par of WaveNet. These reasons are why the adversarial framework has become extremely popular, as GANs are simple feedforward models that are computationally expensive during training but relatively lightweight in inference [11]. These models excel at capturing fine-grained acoustic details while reducing computational overhead, making them suitable for real-time applications.

End-to-End Models. A recent paradigm in text-to-speech is to skip the spectrogram generation and model raw audio waveform directly from text. Early end-to-end TTS models replace spectrograms with latent representations to demonstrate the feasibility of direct synthesis [24]. [4] extend this idea with an adversarial framework, achieving improved naturalness and robustness [4]. However, these models face significant challenges, such as alignment and waveform fidelity. Monotonic alignment search [10] provides a robust solution for sequence alignment in end-to-end pipelines. [16] get address the difficulty of learning frequency information by introducing a variance adaptor, which is supervised to predict phonemic duration, pitch, and energy, which is used to decode out the audio waveform. Other end-to-end approaches rely on modeling a discretized latent space, which leverage self-supervised learning on unpaired datasets and benefit from developments in sequence-to-sequence learning in natural language processing [7].

3 Method

Similar to [4] and [24], our goal is to learn a neural network generator that maps an input sequence of phonemes to raw audio at 22.05 kHz. Because of the vastly different lengths of the input and output signals with no ground truth alignment, we adopt the monotonic aligner method proposed in [4]. Below, we discuss our architecture and training procedure in detail.

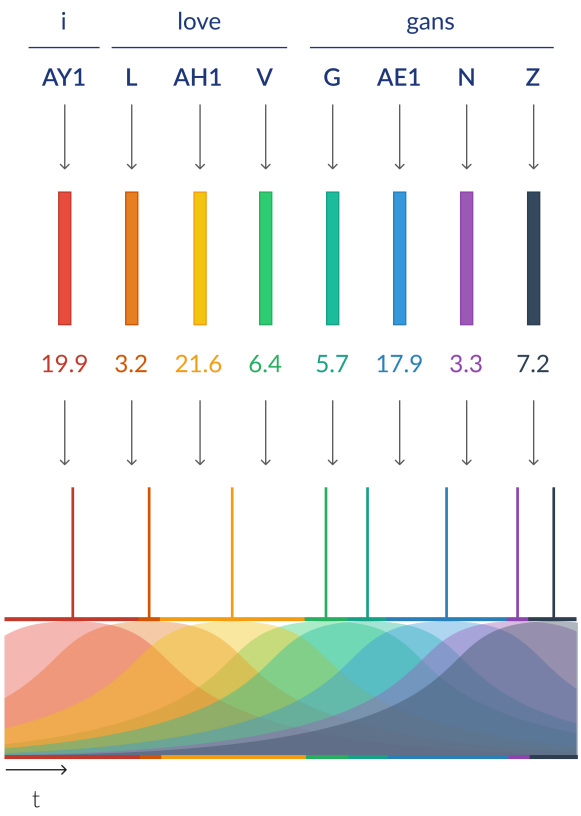

Figure 1: A visualization of the forward pass from phonemization, duration prediction, and the sequence-to-sequence softmax-weighted averaging of phonemic embeddings into waveform embeddings. The numbers represent the predicted duration of each phoneme in deciseconds, and the t axis on the bottom represents the temporal dimension of the aligned output waveform activations.

3.1 Generator

The generator is a feed-forward network consisting of an encoder, aligner, and decoder. Its feed-forward nature allows for fast, batched inference, a desirable trait in text-to-speech models. At a high level, the encoder outputs a latent vector for each phoneme, as well as a scalar indicating its duration. The predicted durations are used to monotonically interpolate the sequence of latent vectors to a lower frequency that the decoder ingests and upsamples to full frequency.

Encoder. Unlike [4], we opt for Transformer layers as in [10] over stacked dilated convolutions in the encoder, as is common in sequential text processing. To this end, we also adopt Gaussian Error Linear Units (GELU) as our activation functions, which provide more gradient flow and boost in performance [6]. The self-attention mechanism is able to model long- and short-term dependencies between input phonemes more explicitly than convolutions, which allows us to construct shallower models with the same expressivity. Unlike [10], we use masked layer normalization to prevent padded positions from affecting the statistics of each layer, though the difference in performance is trivial. As mentioned above, the encoder outputs mean and log variance vectors for each phoneme as well the corresponding durations. We use the reparameterization trick to sample from these distributions.

Aligner. To model the alignment between phonemes and their corresponding waveform timesteps, we use the monotonic aligner proposed in [4], which interpolates the latent vectors from the encoder with the predicted durations using a Gaussian kernel of temperature $\sigma^2 = 10$. See Figure 2 for a visualization and description of how this module performs alignment.

Decoder. The decoder then takes this interpolated sequence of vectors and upsamples them across a series of residual blocks. Similar to [4], we adapt the generator architecture from [1], but adjust the upsampling factors to be whole factors of the target frequency (22.05 kHz) of our dataset. We also use GELU activations here.

3.2 Discriminator

There are two discriminators: one that operates on audio waveforms and one that operates on its corresponding mel spectrogram. Note that the generator outputs raw waveforms and the mel spectrogram is simply extracted from it by computing short-time Fourier transform and applying mel filters to transformer output. Architectural details can be found in the Appendix.

Audio Discriminator. Random window discriminators have been proven to be effective at learning distinctions between real and synthetic audio [1, 4]. Like [4], we apply a set of unconditional random window discriminators that each operate on different lengths of audio using residual blocks from [1].

Spectrogram Discriminator. Similar to [4], we adopt a discriminator that operates on the mel spectrograms computed from input audio. Unlike [4], we replace 2D convolutions with 1D convolutions, framing the spectrogram discriminator as a sequence rather than an image. Following [22, 8], we adopt a multi-scale architecture for the spectrogram discriminator, which outputs a matrix of scores for each resolution.

3.3 Objective Function

We use a composite loss function to optimize all networks end-to-end. All losses are summed with the exception of the length loss, which is scaled by 0.1 before being summed with the other losses.

Latent loss. We impose a KL divergence loss on the latent vectors from the encoder to encourage the distributions

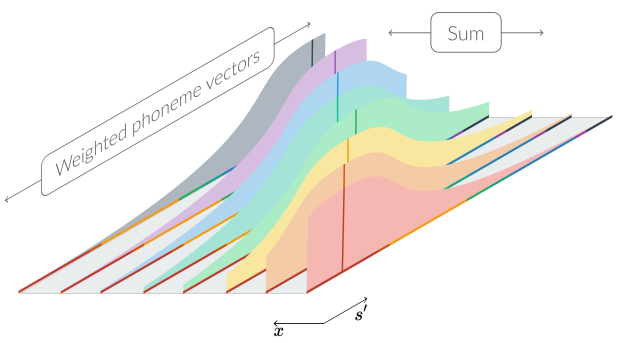

Figure 2: A visualization of the alignment module, which aggregates the sequence of phoneme embeddings into a sequence of audio samples via duration predictions. The x axis represents the phonemes, whose representations are place at the cumulative duration centers predicted by the duration predictor (represented as a dark vertical line). Their representations are decayed from the center with a Gaussian kernel, similar to [4], and these scaled phonemes are averaged via softmax sum into each position of the output sequence, represented as s'. This output sequence is then upsampled and decoded into the raw audio waveform.

corresponding to each phoneme to be close to a standard multivariate Gaussian.

$$L_{latent} = \mu_x^2 + \sigma_x^2 - \log \sigma_x^2 - 1 \tag{1}$$

where $\mu_x, \log \sigma_x^2$ are the phonemic statistics outputted by the encoder.

Length loss. Because we have no access to ground truth alignment, the only duration loss we can impose is with respect to the total duration. Thus,

$$L_{length} = \left(l_{\hat{y}} - \sum_{i=0}^{l_x} \hat{l}_{i\hat{y}} \right)^2 \tag{2}$$

where $l_{\hat{y}}$ is the total audio length and $\hat{l}_{i\hat{y}}$ is the predicted length for phoneme i. Both quantities refer to the audio length at the low-frequency of the aligner output.

Prediction loss. As in [4], we apply soft dynamic time warping loss (SDTW) [2] to the real and synthetic spectrograms with a warping penalty of 1. This alone, however, is not enough for the model to learn alignment so we weight it against an L1 loss. Thus, for a synthesized $\hat{z}$ from the model and ground truth z, the prediction loss becomes

$$L_{prediction} = \alpha_i L_{hard} + (1 - \alpha_i) L_{soft} \tag{3}$$

where

$$L_{hard} = |z - \hat{z}| \tag{4}$$

and

$$L_{soft} = \texttt{SDTW}(\hat{z}, z) = r_{-1,-1} \tag{5}$$

Following notation from [2], $r_{i,j}$ is the accumulated cost at $\hat{z}_i$ and z_j and $\delta(\hat{z}_i, z_j)$ is the L1 distance between $\hat{z}_i$ and z_j. $r_{i,j}$ can be computed recursively with dynamic programming.

$$r_{i,j} = \delta(\hat{z}_i, z_j) - \gamma \log \left[e^{-\frac{r_{i-1,j}+\omega}{\gamma}} + e^{-\frac{r_{i,j-1}+\omega}{\gamma}} + e^{-\frac{r_{i-1,j-1}}{\gamma}} \right] \tag{6}$$

The computational complexity of this soft alignment table is $\mathcal{O}(|\hat{z}||z|)$ and the total alignment loss will be the last entry computed since it is backed up with dynamic programming by the cost of the path that was taken to compute it. Because L_{hard} and L_{soft} will inevitably conflict in how they enforce alignment, they are scheduled such that the hard alignment loss is annealed throughout training while the soft alignment loss is increased throughout training. This weighting α_i is computed per epoch i as

$$\alpha_i = e^{-i/\sqrt{T}} \tag{7}$$

where T is the total number of epochs to train for.

Adversarial loss. Both discriminators are trained with least squares loss [13]. The generator (G) loss is

$$L_{adv,G} = (D_y(\hat{y}) - 1)^2 + (D_z(\hat{z}) - 1)^2 \tag{8}$$

where $\hat{y}$ is the generated output and $\hat{z}$ is its corresponding mel spectrogram. The audio discriminator (D_y) loss is

$$L_{adv,D_y} = D_y(\hat{y})^2 + (D_y(y) - 1)^2 \tag{9}$$

and the spectrogram discriminator (D_z) loss is

$$L_{adv,D_z} = D_z(\hat{z})^2 + (D_z(z) - 1)^2 \tag{10}$$

where y is the ground truth audio and z is its corresponding mel spectrogram.

4 Experiments

In all experiments, we train the end-to-end model from scratch. Our system integrates both a speech synthesis network and an adversarial loss component, which allows speech features to be learned in a fully unsupervised manner. We utilize the Mean Opinion Score (MOS) evaluation metric by running all audio samples for all baselines on Mechanical Turk, as is standard for all work in the TTS literature. Each Mechanical Turker listens to all audio samples from all baselines to ensure rating consistency. The MOS scores are then aggregated with 95% confidence intervals, where 5 indicates natural realistic speech and 1 indicating poor quality speech. We evaluate against the Glow-TTS [10] with HiFiGAN [11] vocoder pipeline, since this is the

best two-stage approach that is open-source and available. None of [4, 16, 24] are open-source at the time of writing this paper.

We find that this base approach approach achieves competitive MOS scores, demonstrating that the combination of adversarial training with external supervision and pretrained components is a viable solution to both stability and naturalness of the synthesized speech.

4.1 Dataset

We trained our model on the LJSpeech dataset, which consists of approximately 24 hours of speech from a single speaker. The dataset contains 13,100 short audio clips paired with their corresponding text transcriptions, providing rich phonemic coverage necessary for training a high-quality end-to-end text-to-speech system. The dataset is monospeaker, ensuring that speaker-specific characteristics are controlled during training.

4.2 Training Stability

Training the end-to-end model from scratch on the LJSpeech dataset proved to be a challenging and finicky process. We observed that, without adequate supervision, the model exhibited unstable behavior during training. One of the key challenges was the optimization of the speech synthesis pipeline, which required a delicate balance between the generative adversarial loss and the reconstruction loss. Early training iterations often led to instability and poor convergence, with generated speech suffering from unnatural intonation and distorted phonetic structure. The L_{hard} versus L_{soft} scheduling was very brittle on a dataset this scale – in general, alignment would probbaly be easier to learn and require less supervision at a larger dataset scale.

4.3 Duration Supervision

To experiment with stability, we also run ablations where the duration predictor is supervised with durations computed with Montreal Forced Alignment (MFA) [14]. Because the duration predictor is directly responsible for how the phonemic embeddings are broadcasted and averaged, we observe that providing explicit signal for this module greatly reduces the local minima during optimization, similar to [17]. These labels guide the duration of phonemes during training and make the alignment problem much easier to learn. Incorporating this additional supervision significantly improved the training stability and accelerated convergence. Because MFA is a non-parametric method of getting pseudo-ground truth, it could be used similarly as spectrogram computation to supervise portions of text-to-speech systems. We leave explorations of incorporating this as a auxiliary scheduled loss up to future work.

4.4 Pretrained Phonemizer

We further enhance the performance and stability of the model by replacing the phoneme encoder with a pretrained model that maps text directly to phonemic embeddings. Specifically, we experiment with the ByT5 model, a recent token-free model based on byte-level text processing [25], and simply add a single Transformer layer after computing its embeddings on the phonemic input. We do not update the Byt5 layers in training our model to prevent catastrophic forgetting from overfitting. Because Byt5 is a large pretrained model trained on large amounts of text, it is able to capture much more complex representations of phonemes than one trained from scratch on LJSpeech. Using pretrained representations also improves the stability of the training process. We find that, in line with transfer learning in other domains, that a pretrained base model provides a help initialization for the rest of the model for training. We note that there are other similar grapheme-to-phoneme pretrained models such as T5G2P [29] and leave this up to future work to investigate the benefits of large pretrained natural language foundation models on end-to-end text-to-speech systems.

Method	**MOS Score**
Human baseline	4.43 ± 0.07
Glow-TTS [10] + HiFiGAN [11]	4.11 ± 0.08
Ours (from scratch)	3.95 ± 0.06
Ours + MFA loss	4.01 ± 0.06
Ours + MFA loss + Byt5	4.08 ± 0.07

Table 1: Mean Opinion Scores (MOS) with 95% confidence intervals for all baselines, experiments, and ablations.

5 Discussion

This work presents an end-to-end speech synthesis model that combines adversarial training, external supervision, and pretrained phoneme encoders to address challenges in generating natural and stable speech. We design the system after the best practices in TTS literature and train the model from scratch on the LJSpeech dataset. We also investigate auxiliary forms of supervision and pretraining and find that bootstrapping the system with these compute- and data-free methods improves training stability and convergence. Our results show that adversarial training continues to be a viable method in end-to-end learning. Future research could explore scalability to multi-speaker systems, integration of other foundation models, and hybridization with self-supervised learning techniques.

6 Acknowledgements

I thank Chris Donahue for providing me with mentorship through most of this work.

References

[1] M. Bińkowski, J. Donahue, S. Dieleman, A. Clark, E. Elsen, N. Casagrande, L. C. Cobo, and K. Simonyan. High fidelity speech synthesis with adversarial networks, 2019.

[2] M. Cuturi and M. Blondel. Soft-dtw: a differentiable loss function for time-series, 2018.

[3] C. Donahue, J. McAuley, and M. Puckette. Adversarial audio synthesis, 2019.

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

[5] T. Gerkmann, M. Krawczyk-Becker, and J. Le Roux. Phase processing for single-channel speech enhancement: History and recent advances. *IEEE Signal Processing Magazine*, 32(2):55–66, 2015.

[6] D. Hendrycks and K. Gimpel. Gaussian error linear units (gelus), 2020.

[7] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units, 2021.

[8] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks, 2018.

[9] K. Ito and L. Johnson. The lj speech dataset. https://keithito.com/LJ-Speech-Dataset/, 2017.

[10] J. Kim, S. Kim, J. Kong, and S. Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search, 2020.

[11] J. Kong, J. Kim, and J. Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis, 2020.

[12] 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, 2019.

[13] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks, 2017.

[14] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger. Montreal forced aligner: Trainable text-speech alignment using kaldi. In *Interspeech 2017*, pages 498–502, 2017.

[15] R. Prenger, R. Valle, and B. Catanzaro. Waveglow: A flow-based generative network for speech synthesis, 2018.

[16] 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, 2022.

[17] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu. Fastspeech: Fast, robust and controllable text to speech, 2019.

[18] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions, 2018.

[19] C. M. University. Cmu pronouncing dictionary. http://www.speech.cs.cmu.edu/cgi-bin/cmudict, 2014. Version 0.7b, accessed November 25, 2024.

[20] A. van den 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, 2016.

[21] A. van den Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. van den Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, N. Casagrande, D. Grewe, S. Noury, S. Dieleman, E. Elsen, N. Kalchbrenner, H. Zen, A. Graves, H. King, T. Walters, D. Belov, and D. Hassabis. Parallel wavenet: Fast high-fidelity speech synthesis, 2017.

[22] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans, 2018.

[23] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark, and R. A. Saurous. Tacotron: Towards end-to-end speech synthesis, 2017.

[24] R. J. Weiss, R. Skerry-Ryan, E. Battenberg, S. Mariooryad, and D. P. Kingma. Wave-tacotron: Spectrogram-free end-to-end text-to-speech synthesis, 2020.

[25] L. Xue, A. Barua, N. Constant, R. Al-Rfou, S. Narang, M. Kale, A. Roberts, and C. Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models, 2021.

[26] R. Yamamoto, E. Song, and J.-M. Kim. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram, 2020.

[27] 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, 2020.

[28] A. Łańcucki. Fastpitch: Parallel text-to-speech with pitch prediction, 2021.

[29] M. Řezáčková, J. Švec, and D. Tihelka. T5g2p: Using text-to-text transfer transformer for grapheme-to-phoneme conversion. In *Interspeech 2021*, pages 6–10, 2021.