PARALLEL TACOTRON: NON-AUTOREGRESSIVE AND CONTROLLABLE TTS

Isaac Elias, Heiga Zen, Jonathan Shen, Yu Zhang, Ye Jia, Ron J. Weiss, Yonghui Wu

Google

ABSTRACT

Although neural end-to-end text-to-speech models can synthesize highly natural speech, there is still room for improvements to its efficiency and naturalness. This paper proposes a non-autoregressive neural text-to-speech model augmented with a variational autoencoder-based residual encoder. This model, called *Parallel Tacotron*, is highly parallelizable during both training and inference, allowing efficient synthesis on modern parallel hardware. The use of the variational autoencoder relaxes the one-to-many mapping nature of the text-to-speech problem and improves naturalness. To further improve the naturalness, we use lightweight convolutions, which can efficiently capture local contexts, and introduce an iterative spectrogram loss inspired by iterative refinement. Experimental results show that Parallel Tacotron matches a strong autoregressive baseline in subjective evaluations with significantly decreased inference time.

Index Terms— Neural TTS; non-autoregressive; VAE; self-attention;

1. INTRODUCTION

Neural end-to-end text-to-speech (TTS) has been researched extensively in the last few years [1–4]. Tacotron 2, which combined an attention-based encoder-decoder model predicting a mel-spectrogram given a character sequence and a WaveNet model [5] predicting speech waveform given the predicted mel-spectrogram, could synthesize read speech almost as natural as human [6]. Following the same approach, a number of models have been proposed to synthesize various aspects of speech (*e.g.*, speaking styles) in a natural sounding way.

Tacotron 2 uses an autoregressive uni-directional long short-term memory (LSTM)-based decoder with the soft attention mechanism [7]. This architecture makes both training and inference less efficient on modern parallel hardware than fully feed-forward architectures. Furthermore, like other autoregressive models, Tacotron 2 uses teacher forcing [8], which introduces discrepancy between training and inference [9, 10]. Together with the soft attention mechanism, it can lead to robustness errors such as babbling, early cut-off, word repetition, and word skipping [11–14]. Although Transformer TTS [4] addresses the inefficiency during training, it is still inefficient during inference and has the potential for robustness errors due to the autoregressive decoder and the attention mechanism.

Recently a number of neural TTS models relying on explicit phoneme duration rather than the soft attention mechanism were proposed [15–24]. They are analogous to conventional parametric synthesis models [25, 26], where phoneme durations are often modeled explicitly. As these models are based on phoneme durations, they are less prone to synthesize speech with the robustness errors. Furthermore, rhythm of synthesized speech can be controlled precisely by modifying predicted durations. Some of these models train their explicit duration model to predict groundtruth phoneme duration extracted from autoregressive models [15, 18, 22], internal aligner models [17, 19], or external aligner models [16, 23, 24].

Among these duration-based models, some of them use the autoregressive decoder [23, 24], whereas others use a non-autoregressive decoder [15–22], which are more efficient during both training and inference.

Text-to-speech synthesis is a one-to-many mapping problem, as there can be multiple possible speech realizations with different prosody for a text input. Neural TTS models with autoregressive decoders can use the previous mel-spectrogram frame and the text to predict the next mel-spectrogram frame, where the previous frame can provide context to disambiguate between multi-modal outputs. However, models with non-autoregressive decoders need a different way to get context to choose a mode. FastSpeech [15] used knowledge distillation to handle this issue, whereas FastSpeech 2 [16] addressed this problem elegantly by adding supervised F_0 and energy as conditioning for its non-autoregressive decoder during training. They are predicted in the same manner as duration and fed to the decoder during inference. Yet another way to address this problem is to augment the model by incorporating an auxiliary representation to capture the latent factors in the speech data [27–30]. An example of such latent factors in speech is prosody, which is isolated from linguistic content (text). The auxiliary representation, which can be a form of style tokens [27] or a latent vector from a variational autoencoder (VAE), is extracted by an encoder taking groundtruth spectrogram as its input.

This paper presents a non-autoregressive neural TTS model augmented by a VAE. The model, called *Parallel Tacotron*[1], has the following properties which we found to be helpful to synthesize highly natural speech efficiently; (1) non-autoregressive architecture based on self-attention with lightweight convolutions [31], (2) iterative mel-spectrogram loss [32], (3) VAE-style residual encoder [28, 30].

The rest of the paper is organized as follows. Section 2 explains the architecture of Parallel Tacotron. Section 3 describes experimental results. Concluding remarks are given at the final section.

2. PARALLEL TACOTRON

Figure 1 illustrates the architecture of the Parallel Tacotron model. It consists of an input encoder, a residual encoder, a duration decoder, an upsampling block, and a spectrogram decoder stack.

The model heavily relies on self-attention blocks with either Transformer or lightweight convolutions (LConv). The LConv is a depth-wise convolution which shares certain output channels whose weights are normalized across time [31]. It has a few orders of magnitude less parameters than a standard non-separable convolution. Unlike Transformer-based self-attention, lightweight convolutions have a fixed context window and reuse the same weights for context elements, regardless of the current time-step. This can be more useful in TTS as relevant context elements can be more local than machine translation or other language tasks. The LConv block is

---

[1]Audio examples: `https://google.github.io/tacotron/publications/parallel_tacotron/`

also illustrated in Fig. 1. It consists of a gated linear unit (GLU), a lightweight convolution, and a feedforward (FF) layer with residual connections. As in [31] we use dropout 0.1, and perform FF mixing in the FF layer using the structure $\text{ReLU}(W_1 X + b_1)W_2 + b_2$ where W_1 increases the dimension by a factor 4.

2.1. Input Encoder

As in Tacotron 2 [6], the input encoder consists of a phoneme embedding lookup followed by three convolutional blocks. Thereafter, sinusoidal positional embeddings are added, then six Transformer blocks are applied.[2] The output of the encoder can be viewed as a contextual phoneme representation. The encoder outputs are concatenated with a speaker embedding of dimension 64 and latent representation from the residual encoder.

2.2. Variational Residual Encoder

Two different VAE models are investigated; a global VAE per speaker similar to [28] and a fine-grained phoneme-level VAE akin to [30].

2.2.1. Global VAE per Speaker

Like multi-speaker modeling in [28], the model is augmented by a global VAE per speaker, each with a speaker specific learned prior. The latent representation from the VAE can be viewed as the residual information which cannot be represented by input phoneme sequences and associated speaker IDs, such as prosodic variations [28]. The residual encoder, also known as the posterior network, takes target mel-spectrograms as its input and processes it using a stack of LConv blocks. The stack consists of; three 17×1 LConv blocks, followed by five 17×1 LConv blocks interleaved with strided 3×1 convolutions, where all LConv blocks have eight heads. This lets the model successively downsample the representation before applying global average pooling to get the final global latent representation. Although the dimension of the latent representation is 8, it is projected up to 32. During inference, a speaker specific prior mean is used.

In addition to modeling the residual information, this latent representation works to implicitly capture the dependencies among output mel-spectrogram frames given input tokens [33–35].

2.2.2. Phoneme-Level Fine-Grained VAE

A fine-grained phoneme-level VAE [30] has the posterior network which aligns the groundtruth spectrogram with the encoder outputs using attention. To achieve this, the positional embeddings from Section 2.4 and the speaker embedding are concatenated to the groundtruth spectrogram frames, subsequently five 8-headed 17×1 LConv blocks are applied. The attention is then computed with the layer normalized [36] encoder outputs. We use an 8-dimensional latent representation, concatenate the speaker embedding and encoder outputs, and then project it to dimension 32.

Furthermore, similar to [30], we train a separate autoregressive LSTM to predict a sequence of latent representation during inference. Similarly with the posterior network, this LSTM takes both speaker embedding and encoder outputs as its input. The LSTM is trained to predict the posterior mean using the L_2 loss with teacher forcing. This loss is computed independently of the posterior network; no error gradient is back-propagated to the posterior network.

---

[2]An LConv stack works equally well.

2.3. Duration Decoder

The encoder outputs augmented by the latent representation and speaker embeddings are fed to a duration decoder which predicts phoneme duration. We assume that groundtruth phoneme durations are provided by an external aligner, such as hidden Markov model (HMM)-based one [37]. Here the input sequence consists of regular phonemes, silences at all word boundaries, and punctuation marks. Whereas regular phonemes usually have non-zero duration, zero-length duration are often associated with silences and punctuation marks (*e.g.*, word boundaries without pauses).

To model such sequences, the duration decoder predicts two types of outputs: binary zero/non-zero duration and continuous phoneme duration in seconds. The duration decoder contains four LConv blocks with 3×1 lightweight convolutions followed by two independent projections. One is followed by a sigmoid activation predicting the probability of non-zero duration p_z. Another is followed by a softplus activation predicting the phoneme duration in seconds. If $p_z < 0.99$ then the predicted duration is zeroed out. Here the cross-entropy (CE) and L_1 loss functions are used for the former and latter, respectively.

2.4. Upsampling and Positional Embeddings

Upsampling takes the activation of the last LConv block in the duration decoder and upsamples them to the length of target spectrogram frame sequence using the phoneme duration. After upsampling, three different types of positional embeddings are added to make the spectrogram decoder aware of the frame positions within phonemes: (1) Transformer-style sinusoidal embedding [38] of a frame position within a phoneme, (2) Transformer-style sinusoidal embedding [38] of phoneme duration, (3) Fractional progression of a frame in a phoneme (1D CoordConv [39]).

Since several positional embeddings are added, per channel a learned weighted sum is used to combine them with the output of the duration decoder. Specifically, the weights are softmax normalized per channel; which lets the network select for each channel which embedding that is preferred by the spectrogram decoder.

2.5. Spectrogram Decoder with Iterative Loss

The output of the upsampling stack is fed to the spectrogram decoder. It has six 8-headed self-attention blocks with 17×1 LConv with dropout 0.1 as in [31]. The output of the last FF layer in each block is then projected to 128-bin mel-spectrogram.

We investigate the use of an iterative loss and its impact on the naturalness of synthesized speech; the outputs of the last FF layers in the self-attention blocks are independently projected to the mel-spectrogram then L_1 losses between the predicted and target mel-spectrogram are summed to form the final loss. This type of iterative loss has previously been used in [32, 34].

2.6. Training Objective

The overall loss functions for the Parallel Tacotron models with global and fine-grained VAEs become

$$\mathcal{L}_{\text{global}} = \frac{1}{KT} \sum_i \mathcal{L}_{\text{spec}_i} + \frac{1}{N}\lambda_{\text{dur}}\mathcal{L}_{\text{dur}} - \beta D_{\text{KL}}, \tag{1}$$

$$\mathcal{L}_{\text{phone}} = \frac{1}{KT} \sum_i \mathcal{L}_{\text{spec}_i} + \frac{1}{N}\lambda_{\text{dur}}\mathcal{L}_{\text{dur}} - \beta D_{\text{KL}} + \frac{1}{N}\mathcal{L}_{\text{prior}}, \tag{2}$$

where $\mathcal{L}_{\text{spec}_i}$ is the L_1 spectrogram loss for the i-th LConv block in the spectrogram decoder, $\mathcal{L}_{\text{dur}}$ is the two-level duration loss, D_{KL}
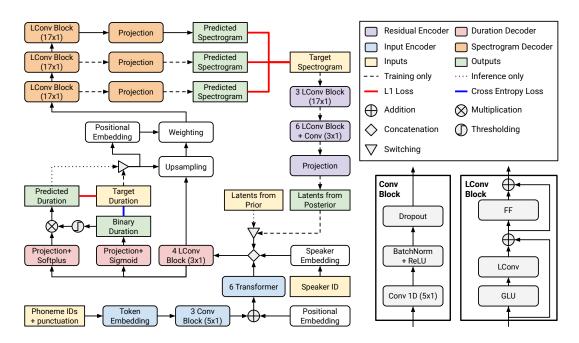
Fig. 1. Block diagram of the Parallel Tacotron model. The residual encoder (purple blocks) in this figure correspond to the global VAE in Section 2.2.1.

is the KL divergence between prior and posterior from the residual encoder, $\mathcal{L}_{prior}$ is the learned prior loss in the phoneme-level VAE, T is the total number of frames, K is the size of spectrogram, and N is the number of tokens.

3. EXPERIMENTS

3.1. Training Setup

We used a proprietary speech dataset containing 405 hours of speech data; 347,872 utterances including 45 speakers in 3 English accents (32 US English speakers, 8 British English, and 5 Australian English speakers). The Parallel Tacotron models were trained with Nesterov[3] momentum optimization and $\alpha = 0.99$. We used a linear warmup from 0.1–1.0 for the first 10K steps followed by the exponential decay from step 20K to 100K with a minimum value of 0.01. All models were trained for 120K steps with global gradient norm clipping of 0.2 and a batch size of 2,048 using Google Cloud TPUs. Training took less than one day.

The fine-grained phoneme-level VAE model was trained differently. It used a KL-weight schedule where β was increased linearly to 1.0 from step 6K to 50K. Then the fine-grained phoneme-level VAE model was trained with exponential decay between steps 40K and 120K and we chose the checkpoint at step 215K.

As a baseline we used the Tacotron 2 model [6] with some small modifications: specifically it used Gaussian mixture model (GMM) attention [40, 41] and a reduction factor of 2 [2], both of which have been shown to improve the robustness [24]. Tacotron 2 was trained for 500K steps with batch size 1,024 using Adam with exponential decay from step 50K to 400K.

Both baseline and proposed models were combined with the same pretrained WaveRNN neural vocoder [42] to reconstruct audio signals from predicted mel-spectrograms.

---

[3] Adam optimizer with the typical Transformer schedule performed worse.

3.2. Evaluation Setup

Subjective evaluations were conducted using 1,000 sentences. These sentences were different from the training data and was used in the previous papers. They were synthesized using 10 US English speakers (5 male & 5 female) in a round-robin style (100 sentences per speaker). The naturalness of the synthesized speech was evaluated through subjective listening tests, including 5-scale Mean Opinion Score (MOS) tests and side-by-side preference tests. For MOS tests, a five-point Likert scale score (1: Bad, 2: Poor, 3: Fair, 4: Good, 5: Excellent) was adopted with rating increments of 0.5. For side-by-side preference tests each rater listened to two samples then rated each with integral scores $[-3, 3]$; where a positive score indicated that the first sample sounded better than the second one [6, 41].

3.3. Experimental Results

The first experiment evaluated the effect of the iterative loss. Tables 1 and 2 show the experimental result. Although there was no significant difference in MOS and preference against Tacotron 2, the direct comparison between models with and without the iterative loss indicate that it can give small improvement.

The second experiment evaluated the impact of VAEs. Tables 5 and 6 show the experimental results. Parallel Tacotron without VAE was significantly worse than the baseline Tacotron 2 in both MOS and preference. The introduction of global VAE made it comparable to the baseline Tacotron 2 in both evaluations. Furthermore, the introduction of fine-grained phoneme-level VAE further boosted the naturalness.

The third experiment compared Transformer and LConv for self-attention. Tables 3 and 4 show the experimental results. Parallel Tacotron with both Transformer and LConv-based self-attention matched the baseline Tacotron 2 both in MOS and preference. When they were directly compared, Parallel Tacotron with LConv was more preferred.

Table 1. Subjective evaluations of Parallel Tacotron with and without the iterative loss. Positive preference scores indicate that the corresponding Parallel Tacotron model was rated better than the reference Tacotron 2.

Model	MOS	Preference
Tacotron 2	4.45 ± 0.04	Reference
Parallel Tacotron (LConv, w/o VAE)		
single loss	4.32 ± 0.04	**−0.09 ± 0.06**
iterative loss	4.33 ± 0.04	**−0.08 ± 0.06**

Table 2. Subjective preference scores between Parallel Tacotron with and without the iterative loss. Positive preference scores indicate that the corresponding model with the iterative loss was rated better than the one without the iterative loss.

Model	Preference
LConv w/o VAE	**0.09 ± 0.05**
LConv w/ Global VAE	**0.07 ± 0.05**
Transformer w/ Global VAE	**0.08 ± 0.05**

Table 3. Subjective evaluations of Parallel Tacotron with different self-attention (with Global VAE and iterative loss). Positive preference scores indicate that the corresponding Parallel Tacotron was rated better than Tacotron 2.

Model	MOS	Preference
Parallel Tacotron (w/ iterative loss & Global VAE)		
Transformer	4.36 ± 0.04	−0.03 ± 0.06
LConv	4.40 ± 0.04	−0.01 ± 0.05

Table 4. Subjective preference score between Parallel Tacotron using LConv and Transformer-based self-attention. Positive preference scores indicate that LConv was rated better than Transformer.

Model	Preference
LConv *vs* Transformer	**0.05 ± 0.04**

Table 5. Subjective evaluations of Parallel Tacotron with different VAEs. Positive preference scores indicate that the corresponding Parallel Tacotron was rated better than Tacotron 2.

Model	MOS	Preference
Parallel Tacotron (w/ iterative loss & LConv)		
No VAE	4.33 ± 0.04	**−0.08 ± 0.06**
Global VAE	4.40 ± 0.04	**−0.01 ± 0.05**
Fine VAE	4.42 ± 0.04	**0.07 ± 0.07**

Table 6. Subjective preference scores between Parallel Tacotron using the global and fine-grained VAEs. Positive preference scores indicate that left models were rated better than the right ones.

Model	Preference
Global VAE *vs* No VAE	**0.13 ± 0.05**
Fine VAE *vs* Global VAE	**0.06 ± 0.06**

Table 7. Subjective preference scores between synthetic and natural speech. The preference scores become positive when synthetic speech was rated better than natural one.

Model	Preference
Tacotron 2	**−0.09 ± 0.07**
Parallel Tacotron w/ LConv & iterative loss	
Global VAE	−0.07 ± 0.07
Fine VAE	−0.01 ± 0.06

Table 8. Inference speed to predict mel-spectrograms for ∼20-second long utterance on a TPU (aggregated over ten trials).

Model	Mean (1/**RTF**)	Stddev
Tacotron 2	79	1
Parallel Tacotron		
Transformer w/ Global VAE	804	5
LConv w/ Global VAE	1051	13
LConv w/ Fine VAE	1013	10

The last evaluation compared was done over 1,000 utterances from a held-out test set. This allowed us to make direct comparisons with natural speech. The MOS of natural speech was 4.54 ± 0.04 on this test set. Experimental results are shown in Table 7. It can be seen from the table that all neural TTS models are doing well compared to human speech, however there is still a room for further improvement.

3.4. Inference Efficiency

Table 8 shows the inference time to predict mel-spectrograms for a 20 second-long utterance on a tensor processing unit (TPU). It can be seen from the table that Parallel Tacotron was about 13 times faster than Tacotron 2. It also shows that the use of lightweight convolutions is faster than of Transformer. It should be noted that little work has been done to trade-off model size and speed against the naturalness thus further speedup is highly possible.

4. CONCLUSIONS

A non-autoregressive neural TTS model called Parallel Tacotron was proposed. It matched the baseline Tacotron 2 in naturalness and offered significantly faster inference than Tacotron 2. We also showed that both variational residual encoders and an iterative loss improved the naturalness, and the use of lightweight convolutions as self-attention improved both naturalness and efficiency.

Future work includes the direct comparison against duration-based models with the autoregressive decoder and those using additional representation such as F_0 and energy. Investigating better fine-grained variational models is also necessary.

5. ACKNOWLEDGEMENTS

6. REFERENCES

[1] J. Sotelo, S. Mehri, K. Kumar, J. F. Santos, K. Kastner, A. C. Courville, and Y. Bengio, "Char2Wav: End-to-End Speech Synthesis," in *Proc. ICLR*, 2017.

[2] Y. Wang, RJ 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," in *Proc. Interspeech*, 2017, pp. 4006–4010.

[3] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep Voice 3: 2000-Speaker Neural Text-to-Speech," in *Proc. ICLR*, 2018.

[4] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural Speech Synthesis with Transformer Network," in *Proc. AAAI*, 2019, vol. 33, pp. 6706–6713.

[5] 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," *arXiv:1609.03499*, 2016.

[6] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, RJ Skerrv-Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," in *Proc. ICASSP*, 2018.

[7] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proc. ICLR*, 2015.

[8] R. J. Williams and D. Zipser, "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[9] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks," in *Proc. NIPS*, 2015, pp. 1171–1179.

[10] A. Goyal, A. Lamb, Y. Zhang, S. Zhang, A. Courville, and Y. Bengio, "Professor Forcing: A New Algorithm for Training Recurrent Networks," in *Proc. NIPS*, 2016, pp. 4601–4609.

[11] M. He, Y. Deng, and L. He, "Robust Sequence-to-Sequence Acoustic Modeling with Stepwise Monotonic Attention for Neural TTS," in *Proc. Interspeech*, 2019, pp. 1293–1297.

[12] Y. Zheng, J. Tao, Z. Wen, and J. Yi, "Forward–backward decoding sequence for regularizing end-to-end tts," *IEEE/ACM Trans. Audio Speech & Lang. Process.*, vol. 27, no. 12, pp. 2067–2079, 2019.

[13] H. Guo, F. K. Soong, L. He, and L. Xie, "A New GAN-Based End-to-End TTS Training Algorithm," in *Proc. Interspeech*, 2019, pp. 1288–1292.

[14] E. Battenberg, RJ Skerry-Ryan, S. Mariooryad, D. Stanton, D. Kao, M. Shannon, and T. Bagby, "Location-relative attention mechanisms for robust long-form speech synthesis," in *Proc. ICASSP*, 2020, pp. 6194–6198.

[15] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "FastSpeech: Fast, Robust and Controllable Text to Speech," *arXiv:1905.09263*, 2019.

[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," *arXiv:2006.04558*, 2020.

[17] S. Beliaev, Y. Rebryk, and B. Ginsburg, "TalkNet: Fully-Convolutional Non-Autoregressive Speech Synthesis Model," *arXiv:2005.05514*, 2020.

[18] D. Lim, W. Jang, H. Park, B. Kim, and J. Yoon, "JDI-T: Jointly trained Duration Informed Transformer for Text-To-Speech without Explicit Alignment," *arXiv:2005.07799*, 2020.

[19] Z. Zeng, J. Wang, N. Cheng, T. Xia, and J. Xiao, "AlignTTS: Efficient Feed-Forward Text-to-Speech System without Explicit Alignment," in *Proc. ICASSP*, 2020, pp. 6714–6718.

[20] C. Miao, S. Liang, M. Chen, J. Ma, S. Wang, and J. Xiao, "Flow-TTS: A Non-Autoregressive Network for Text to Speech Based on Flow," in *Proc. ICASSP*, 2020, pp. 7209–7213.

[21] J. Donahue, S. Dieleman, M. Bińkowski, E. Elsen, and K. Simonyan, "End-to-End Adversarial Text-to-Speech," *arXiv:2006.03575*, 2020.

[22] A. Lańcucki, "FastPitch: Parallel Text-to-speech with Pitch Prediction," *arXiv:2006.06873*, 2020.

[23] C. Yu, H. Lu, N. Hu, M. Yu, C. Weng, K. Xu, P. Liu, D. Tuo, K. Kang, G. Lei, D. Su, and D. Yu, "DurIAN: Duration informed attention network for multimodal synthesis," *arXiv:1909.01700*, 2019.

[24] J. Shen, Y. Jia, M. Chrzanowski, Y. Zhang, I. Elias, H. Zen, and Y. Wu, "Non-Attentive Tacotron: Robust and Controllable Neural TTS Synthesis Including Unsupervised Duration Modeling," *arXiv:2010.04301*, 2020.

[25] H. Zen, K. Tokuda, and A. Black, "Statistical Parametric Speech Synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[26] H. Zen, A. Senior, and M. Schuster, "Statistical Parametric Speech Synthesis Using Deep Neural Networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[27] Y. Wang, D. Stanton, Y. Zhang, RJ Skerry-Ryan, E. Battenberg, J. Shor, Y. Xiao, Y. Jia, F. Ren, and R. A. Saurous, "Style Tokens: Unsupervised Style Modeling, Control and Transfer in End-to-End Speech Synthesis," in *Proc. ICML*, 2018, pp. 5167–5176.

[28] W.-N. Hsu, Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Y. Wang, Y. Cao, Y. Jia, Z. Chen, J. Shen, P. Nguyen, and R. Pang, "Hierarchical Generative Modeling for Controllable Speech Synthesis," in *Proc. ICLR*, 2019.

[29] Y. Zhang, R. J. Weiss, H. Zen, Wu Y., Z. Chen, RJ Skerry-Ryan, Y. Jia, A. Rosenberg, and B. Ramabhadran, "Learning to Speak Fluently in a Foreign Language: Multilingual Speech Synthesis and Cross-Language Voice Cloning," in *Proc. Interspeech*, 2019, pp. 2080–2084.

[30] G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, and Y. Wu, "Fully-Hierarchical Fine-Grained Prosody Modeling for Interpretable Speech Synthesis," *arXiv:2002.03785*, 2020.

[31] F. Wu, A. Fan, A. Baevski, Y. N. Dauphin, and M. Auli, "Pay Less Attention with Lightweight and Dynamic Convolutions," in *Proc. ICLR*, 2019.

[32] A. Tjandra, C. Liu, F. Zhang, X. Zhang, Y. Wang, G. Synnaeve, S. Nakamura, and G. Zweig, "DEJA-VU: Double Feature Presentation and Iterated Loss in Deep Transformer Networks," in *Proc. ICASSP*, 2020, pp. 6899–6903.

[33] J. Gu, J. Bradbury, C. Xiong, V. O. K. Li, and R. Socher, "Non-Autoregressive Neural Machine Translation," *arXiv:1711.02281*, 2017.

[34] J. Lee, E. Mansimov, and K. Cho, "Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement," *arXiv:1802.06901*, 2018.

[35] R. Shu, J. Lee, H. Nakayama, and K. Cho, "Latent-Variable Non-Autoregressive Neural Machine Translation with Deterministic Inference Using a Delta Posterior," in *Proc. AAAI*, 2020.

[36] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer Normalization," 2016.

[37] D. Talkin and C. W. Wightman, "The Aligner: Text to Speech Alignment using Markov Models and a Pronunciation Dictionary," in *ESCA/IEEE SSW2*, 1994.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Proc. NeurIPS*, 2017.

[39] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski, "An Intriguing Failing of Convolutional Neural Networks and the CoordConv Solution," in *Proc. NeurIPS*, 2018, pp. 9605–9616.

[40] A. Graves, "Generating Sequences with Recurrent Neural Networks," *arXiv:1308.0850*, 2013.

[41] RJ Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. Weiss, R. Clark, and R. A. Saurous, "Towards End-to-End Prosody Transfer for Expressive Speech Synthesis with Tacotron," in *Proc. ICML*, 2018.

[42] 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 *Proc. ICML*, 2018, pp. 2410–2419.