INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Eden-TTS: A Simple and Efficient Parallel Text-to-speech Architecture with Collaborative Duration-alignment Learning

*Youneng Ma, Junyi He, Meimei Wu, Guangyue Hu and Haojun Fei**

Qifu Technology, China

{mayouneng-jk,hejunyi-jk,wumeimei-jk, huguangyue-jk,feihaojun-jk}@360shuke.com

Abstract

In pursuit of high inference speed, many non-autoregressive neural text-to-speech (TTS) models have been proposed for parallel speech synthesis recently. A critical challenge of parallel speech generation lies in the learning of text-speech alignment. Existing methods usually require an external aligner for guidance or involve complex training process. In this work, we propose Eden-TTS, a simple and efficient parallel TTS architecture which jointly learns duration prediction, text-speech alignment and speech generation in a single fully-differentiable model. The alignment is learned implicitly in our architecture. A novel energy-modulated attention mechanism is proposed for alignment guidance which leads to fast and stable convergence of our model. Our model can be easily implemented and trained. Experiments demonstrate that our method can generate speech of high quality with high training efficiency.

Index Terms: Text To Speech, Non-autoregressive Speech Synthesis, Text-Speech Alignment, Guided Attention

1. Introduction

Text-to-speech (TTS) systems based on neural network have seen rapid development in recent years. Many popular neural TTS systems use melspectrogram (mel) as their learning target, which is then sent to a separately trained vocoder such as wavenet [1], waveRNN [2], WaveGlow [3], hifigan [4] and so forth to synthesize waveform. Although autoregressive models such as tacotron [5], tacotron2 [6], Deep Voice [7], Transformer TTS [8] can produce speech with good quality for in-domain texts, these methods usually suffer from slow inference speed and robustness issues such as missing and repeated words especially in long utterances [9]. To increase the synthesis speed, many efforts have been put on the parallel speech prediction recently [9–19]. The main challenge for parallel speech prediction lies in the learning of alignment between input text and output speech. Once the alignment is known, a feed-forward decoder can be readily used to do the parallel prediction with aligned features. Many brilliant methods have been proposed for the task. These methods can be roughly categorized into two groups. The first group of methods [9–12, 20] use external aligners such as pre-trained autoregressive models [6, 8, 21] or Montreal Forced Aligner [22] for alignment guidance. This kind of methods have several limitations. In real scenarios, the external aligner can be unavailable, training an aligner and extracting the alignments from it can be time-consuming, meanwhile the extracted alignments can be sub-optimal. The second group of methods [13–19, 23] jointly learn the alignment and speech prediction without an external aligner. Methods in [13, 14, 23] construct the alignments only based on the prediction of phonemes, their training is not very efficient and stable. Methods in [15–18] build their models by first considering all possible monotonic text-mel alignments and then extract the most probable one via an independent search algorithm. These methods preclude differentiation and usually require a scheduled training process. EF-TTS [19] utilizes index mapping vector (IMV) for efficient alignment modeling. It jointly learns the speech generation and alignment through a single network, thus it is easier to apply than its counterpart models. However, this method is not very good at long-sequence alignment learning, though it introduces a way to alleviate the problem.

In this work, inspired by the close relationship between text duration and text-mel alignment, we propose a simple and efficient non-autoregressive architecture for speech synthesis. Experiments on the LJSpeech dataset [24] show that our method can synthesize high-quality speech with high training efficiency. Our contribution is summarized as below:

- A simple fully differentiable feed-forward TTS architecture that can jointly learn duration prediction, sequence alignment and speech prediction without introducing additional alignment targets.
- A novel energy-modulated attention mechanism that can guide monotonic alignment learning. The direct modulation on the energy of the text-mel alignment plays an important role for the fast convergence of our model.
- The proposed method can produce high-quality speech with much easier implementation and simpler training procedure than most existing methods, which can ease the application of neural TTS in real scenarios.

Our source code[1] and synthesized audio samples[2] are publicly available.

2. Proposed Method

In TTS scenario, input token durations and text-speech alignment are closely related. Most existing parallel TTS models rely on durations for alignment during inference. Inspired by this fact, we propose a collaborative duration-alignment learning process: first compute initial alignment with monotonic guidance from text and speech features, then calculate token durations from the guided alignment, finally construct monotonic alignment with durations for speech prediction. The text-speech alignment, duration and speech prediction are learned collaboratively in the process. We will first introduce the architecture and learning procedure of proposed method, then go through each component of the architecture in detail.

*Corresponding author

[1]Code: https://github.com/younengma/eden-tts

[2]Audio examples: https://edenynm.github.io/edentts-demo/

10.21437/Interspeech.2023-700

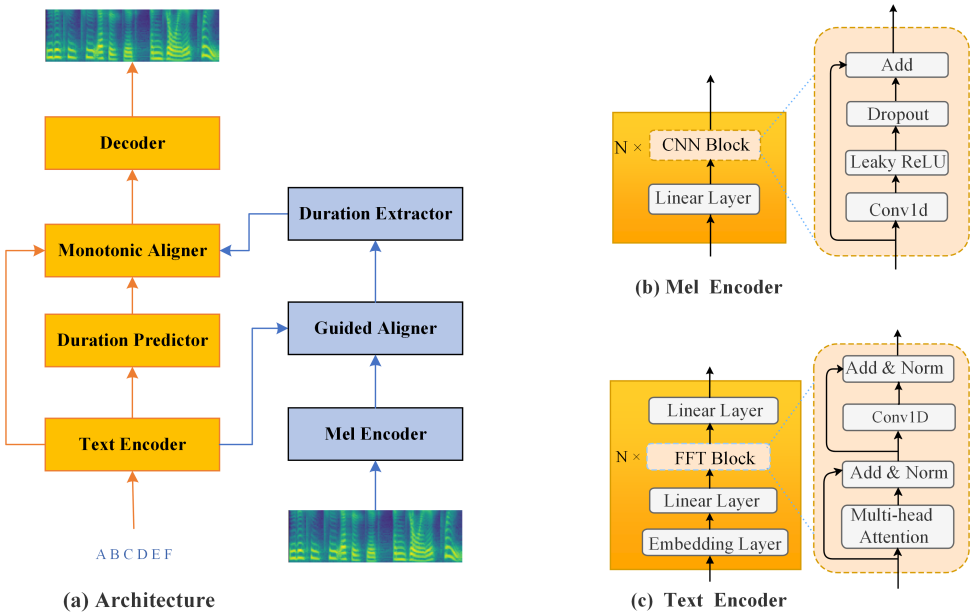


Figure 1: *Architecture of proposed Eden-TTS. (a) depicts the overall architecture. Only the yellow components on the left are used during inference. The blue components on the right are needed only in the training phase. (b) shows the architecture of Mel Encoder. (c) depicts the architecture of Text Encoder.*

2.1. Architecture

The overall architecture of Eden-TTS is depicted in Figure 1(a). In the training phase, text encoder encodes text token sequence as hidden features and mel encoder processes the melspectrogram to output its corresponding hidden representations. Then the guided aligner computes a proper initial alignment between text and mel features, from which token durations are extracted by the duration extractor. The extracted durations serve as targets for the training of duration predictor. Afterwards, the monotonic aligner constructs a hard text-mel alignment from durations and computes time-aligned text features. Finally, the aligned-features are sent to the decoder for speech prediction.

In the inference stage, the duration predictor predicts token durations from encoded text features. Monotonic aligner constructs alignments based on the predicted durations and outputs time-aligned features, which are then used for mel prediction. Only yellow components on the left are needed during inference. Blue components on the right are needed only in the training phase.

2.2. Text and Mel Encoder

The text encoder is depicted in Figure 1(c). It comprises of an embedding layer, followed by a linear layer, a stack of Feed-Forward Transformer (FFT) blocks and a linear layer at last. We use the same FFT block in [9] which consists of a self-attention and 1D convolutional network. The self-attention network consists of a multi-head attention to extract the cross-position information. The text encoder processes the token sequence $\mathbf{x} = (x_0, x_1, ..., x_{N-1})$ of length N as its corresponding hidden representations: $\mathbf{h} = (\mathbf{h}_0, \mathbf{h}_1, ..., \mathbf{h}_{N-1})$.

The mel encoder as shown in Figure 1(b) processes the melspectrogram of length T and outputs its corresponding hidden representations: $\mathbf{m} = (\mathbf{m}_0, \mathbf{m}_1, ..., \mathbf{m}_{T-1})$. It has a linear layer at bottom followed by several blocks of CNN which consists of a convolution layer with leaky relu as activation function with a residual connection. Weight normalization [25] is applied on the convolution layer.

2.3. Guided Aligner

The guided aligner serves to provide a proper initial text-speech alignment during training, from which token durations are to be obtained. The scaled-dot product attention [26] can be utilized:

$$\alpha_{n,t} = \frac{e^{-(\mathbf{h}_n \cdot \mathbf{m}_t)/\sqrt{D}}}{\sum_{i=0}^{N-1} e^{-(\mathbf{h}_i \cdot \mathbf{m}_t)/\sqrt{D}}} \tag{1}$$

where $\mathbf{m}$ and $\mathbf{h}$ are the encoded features of melspectrogram and text respectively, D is their dimensionality. In TTS scenario, the alignment between text and speech is desired to be monotonic and surjective. In other words, α should have larger values near the diagonal and small values off the diagonal. Token durations and alignment are closely related. As the scaled-dot attention has no constraint on the alignment, it can be hard for the duration extractor to obtain reasonable token durations from it. The idea to guide the attention to be monotonic has been proved effective in [27] by introducing a guided loss function and in [18] using a diagonal prior for attention matrix. However, we found using the guided loss not helpful in our experiments, possibly because the constraint of guided loss is not strong enough for efficient learning in our architecture. Introducing a prior on the attention matrix is not suitable either, as it affects latter duration extraction and precludes differentiation.

Inspired by the fact that alignment score $\alpha_{n,t}$ is essentially determined by its corresponding energy $\mathbf{h}_n \cdot \mathbf{m}_t$, we propose to modulate the energy term by directly multiplying it with a proper weight:

$$\alpha_{n,t} = \frac{e^{-(w_{n,t}\mathbf{h}_n \cdot \mathbf{m}_t)/\sqrt{D}}}{\sum_{i=0}^{N-1} e^{-(w_{i,t}\mathbf{h}_i \cdot \mathbf{m}_t)/\sqrt{D}}} \tag{2}$$

where:

$$w_{n,t} = e^{-(n/(N-1)-t/(T-1))^2/(2g^2)} \quad (3)$$

where g is a hyper-parameter used to control the diagonal form of $\mathbf{w}$ and is set as 0.2 in this paper. Figure 2(a) depicts the values of the weight matrix $\mathbf{w}$. As the values of $\mathbf{w}$ are large near the diagonal and small off the diagonal, the off-diagonal energies will be penalized and near-diagonal energies are kept almost intact, forcing the attention to be of similar form. We believe other weight schemes can also be applied as long as their weights have similar diagonal distribution. Simple as it it, our method can directly constrain the alignment during training, therefore much more efficient than the guided loss [27]. When the training has converged, the guided alignment between text and speech is almost monotonic as shown in Figure 2(b), and even near monotonic after removing energy weight as shown in Figure 2(c), which indicates that the guided aligner has learned to produce near-monotonic alignment. Based on the guided alignment, monotonic alignment can be obtained as shown in Figure 2(d).

2.4. Duration Extractor and Predictor

The duration of each token is usually defined as the number of melspectrograms attended to it [9]. However, as the computation in [9] precludes differentiation, it is not suitable in our architecture. We adopt a different method which can lead to similar results. The duration extractor computes the duration of each token by summing its alignment scores with speech features:

$$d_n = \sum_{i=0}^{T-1} \alpha_{n,i}, n = 0, 1, ..., N-1 \quad (4)$$

where d_n is the duration of n^{th} token. It should be noted that to make the above computation reasonable, the alignment should be monotonic (or close to monotonic at least) which is achieved by alignment guidance and the full differentiability of our architecture. The durations are used as the learning targets of the duration predictor during training. The duration predictor stacked on the text encoder has the same architecture as fastspeech [9]. It consists of 2 convolutions with relu activation, each followed by the layer normalization, a linear layer at last to output the duration in logarithmic domain as $\ln \hat{d}$. The duration loss is defined as:

$$L_{dur} = \frac{1}{N} \sum_{n=0}^{N-1} |\ln d_n - \ln \hat{d}_n| \quad (5)$$

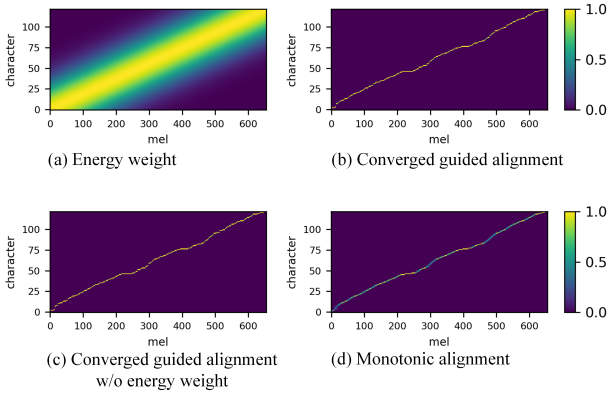
2.5. Monotonic Aligner

The monotonic aligner first constructs the monotonic text-mel alignment from durations, then computes the time-aligned text features for the decoder. We use the method proposed in [23] for monotonic computation. Firstly, the end positions of each token are computed: $e_n = \sum_{i=0}^{n} d_i$ and the center positions: $c_n = e_n - 0.5 d_n$. Then monotonic alignment is calculated by:

$$\beta_{t,n} = \frac{e^{-\sigma^{-2}(t-c_n)^2}}{\sum_{i=0}^{N-1} e^{-\sigma^{-2}(t-c_i)^2}} \quad (6)$$

where σ^{-2} is set as 0.2 in the paper. Finally, time-aligned text representations are computed:

$$\gamma_t = \sum_{n=0}^{N-1} \beta_{t,n} \mathbf{h}_n, t = 0, ..., T-1 \quad (7)$$



(a) Energy weight (b) Converged guided alignment (c) Converged guided alignment w/o energy weight (d) Monotonic alignment

Figure 2: *Visualization of alignment attention matrices when the training has converged. The vertical axis represents text tokens from bottom to top. The horizontal axis denotes mel-frames from left to right. (a) depicts the energy weight matrix. (b) shows the guided alignment when the model has converged. (c) depicts the guided alignment computed without energy modulation when the model has converged. (d) shows the monotonic alignment computed from the guided alignment.*

2.6. Decoder

The decoder processes the aligned text features and output melspectrogram. It comprises of a linear layer at the beginning, followed by several CNN blocks same as that in mel encoder, a linear layer to output melspectrogram and a tacotron2-style postnet [6] at last. While we use the CNN-based decoder for simplicity and efficiency, other popular decoder models such as transformer-based decoder in [9], or flow-based decoder in [15] may also be applied in the architecture. The loss for melspectrogram prediction is defined as: $L_{mel} = (mel - \hat{mel})^2$ and the final loss for our model is the sum of duration loss and mel loss: $L_{tol} = L_{mel} + L_{dur}$.

Note that all the calculations in our architecture are differentiable. Compared with the counterpart model EF-TTS [19], our formulation is much simpler, therefore easier to implement. EF-TTS proposes IMV for monotonic alignment learning. It involves a complex process to obtain relative positions which are comparable to token durations. Eden-TTS computes the token durations by a simple summation on the guided alignment, which is much more straightforward and simpler.

3. Experiments

3.1. Dataset

We conduct all our experiments on the LJSpeech dataset [24], which consists of 13,100 short audio clips with a total duration of approximately 24 hours. 500 samples are used for testing, 100 for validation and the rest for training. We use characters as input tokens, which involves more challenging alignment learning than using phonemes because the character sequence is much longer than the phoneme sequence for the same sentence. We normalize the texts with a space character at the beginning and a period character at the end, e.g. "I have 5 dollars" is processed as " I have five dollars.". Melspectrogram is computed the same way as hifigan [4].

3.2. Experimental Settings

For text encoder, we use 6 FFT blocks with settings same as [9]. The token embedding size of embedding layer in text encoder is set as 512. The number of CNN blocks in mel encoder, decoder are set as 3, 6 respectively. The kernel size and hidden size of CNN blocks are all set as 5, 512 respectively. Dropout rate of CNN layer is all set as 0.1. The duration predictor has a dimension of 256, kernel size of 3.

We compare our model with Glow-TTS [15] which adopts a monotonic search algorithm for alignment learning and shows very good performance, tacotron2 [6] which is the best publicly available autoregressive TTS model, and the end-to-end differentiable EF-TTS [19] which is most comparable to our method. We use the official released models of tacotron2[3] and glow-TTS[4]. For a fare comparison, we implement EF-TTS and Eden-TTS with the same model settings, which means the only difference between them lies in the alignment learning module. We use hifigan [4] as vocoder to synthesize waveform from mel-spectrogram. We use the open-source model of hifigan [5] with v1 configuration. All the models are trained on a single Telsa V100 GPU with batch size of 96. The Adam optimizer is used with learning rate of 10^{-4}.

3.3. Training

Our model can be trained easily due to its full differentiability. Note that energy weights used by the guided aligner only need to be calculated once in the data preparation stage, which will not add training time. Figure 3 shows the mel loss of EF-TTS and Eden-TTS at different training steps. Obviously, Eden-TTS converges faster and reaches a lower mel loss bound than EF-TTS. The training of Eden-TTS can stop at around 150k for satisfactory performance. Meanwhile, as Eden-TTS' alignment module is much simpler than the EF-TTS, it costs around 6% less time at each training step compared with EF-TTS. It takes less than 50% training time of EF-TTS to reach EF-TTS' lowest loss bound. Therefore, our method is much more efficient than EF-TTS.

In our ablation study, we find that the stability and fast convergence have much relevance to the energy-modulated attention. As can be seen in Figure 3, if we replace the energy-modulated attention with regular scaled-dot attention [26] (*Eden w/o energy weight*) or if we replace it with regular scaled-dot attention together with a guided loss [27] (*Eden with guided loss*), the training becomes less efficient with loss curves of larger fluctuation and much higher loss bound.

3.4. Evaluation

Voice Quality: We use the text transcripts in the testset to synthesize speech with models tacotron2 [6], Glow-TTS [15], EF-TTS [19] and Eden-TTS. Model-generated audios and audios synthesized from ground-truth (GT) melspectrogram are rated together by 20 people specialized in English. The 9-scale Mean Opinion Score (MOS) is shown in Table 1. As can be seen from the table, our method can generate better speech than its counterpart parallel models. Though tacotron2 is rated with higher score on the random-chosen test samples, it is not as robust as the parallel models. It can suffer from pronunciation issues for longer sentences while the parallel models don't have this problem [9], [15].

[3]https://github.com/NVIDIA/tacotron2

[4]https://github.com/jaywalnut310/glow-tts

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

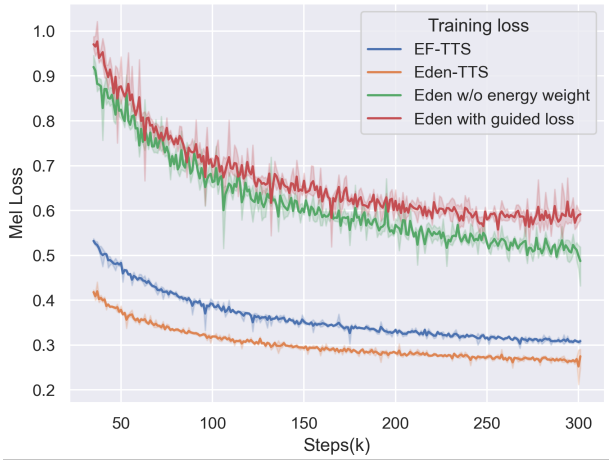


Figure 3: *Mel loss of different methods under same experimental settings. "Eden w/o energy weight" means removing the energy weight in Equation 2 in the guided aligner, namely using the scaled-dot attention to replace the guided aligner. "Eden with guided loss" means using scaled-dot attention applied with guided loss [27] to replace the guided aligner.*

Table 1: *The comparison of 9-scale MOS with 95% confidence intervals and time cost*

Methods	**MOS**	**Time Cost(s)**
GT	4.55 ± 0.13	-
Tacotron2 [6]	4.47 ± 0.08	0.8889
Glow-TTS [15]	4.09 ± 0.14	0.0644
EF-TTS [19]	4.02 ± 0.12	0.0633
Eden-TTS	4.32 ± 0.08	0.0636

Inference Speed: The inference time for the melspectrogram generation of one sentence corresponding to 8.7-seconds ground-truth audio on PC with NVIDIA GPU RTX-2070 is shown in Table 1. We ran the inference 20 times and then took the average inference time. As can be seen in the table, our model's inference speed is comparable to EF-TTS and glow-TTS. It is more than 13.9 times faster than tacotron2.

Controllability: As Eden-TTS uses durations for alignment, it can control the voice speed and part of the prosody by adjusting the input token durations just like [15], [19]. Audio examples of speed control by multiplying a positive scalar value across the predicted token durations are provided in demo page[2]. As you can find in the samples, Eden-TTS can adjust the voice speed rate from 0.8x to 1.6x smoothly.

4. Conclusions

We propose Eden-TTS, an efficient non-autoregressive TTS architecture that can jointly learn text-mel alignment, duration prediction, and speech generation in a single model. It can synthesize high-quality speech with much easier implementation and simpler training procedure than its counterparts. We propose a simple energy-modulated attention mechanism for alignment guidance which is the key to stable, fast convergence of our model. Our work can ease the application of parallel speech generation in real scenarios.

5. References

[1] 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," in *9th ISCA Speech Synthesis Workshop*, 2016.

[2] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *ICML*, 2018, pp. 2410–2419.

[3] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP*, 2019, pp. 3617–3621.

[4] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 022–17 033, 2020.

[5] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio *et al.*, "Tacotron: Towards end-to-end speech synthesis," in *Interspeech*, 2017.

[6] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," in *ICASSP*, 2018, pp. 4779–4783.

[7] 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 *ICLR*, 2018.

[8] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, and M. Zhou, "Neural speech synthesis with transformer network," in *AAAI*, 2019.

[9] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, robust and controllable text to speech," *Advances in neural information processing systems*, vol. 32, 2019.

[10] K. Peng, W. Ping, Z. Song, and K. Zhao, "Parallel neural text-to-speech," *arXiv preprint arXiv:1905.08459*, 2019.

[11] I. Elias, H. Zen, J. Shen, Y. Zhang, Y. Jia, R. J. Weiss, and Y. Wu, "Parallel tacotron: Non-autoregressive and controllable tts," in *ICASSP*, 2021, pp. 5709–5713.

[12] A. Łańcucki, "Fastpitch: Parallel text-to-speech with pitch prediction," in *ICASSP*, 2021, pp. 6588–6592.

[13] 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 *ICASSP*, 2020, pp. 7209–7213.

[14] I. Elias, H. Zen, J. Shen, Y. Zhang, Y. Jia, R. Skerry-Ryan, and Y. Wu, "Parallel tacotron 2: A non-autoregressive neural tts model with differentiable duration modeling," *arXiv preprint arXiv:2103.14574*, 2021.

[15] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-tts: A generative flow for text-to-speech via monotonic alignment search," *Advances in Neural Information Processing Systems*, vol. 33, pp. 8067–8077, 2020.

[16] Z. Zeng, J. Wang, N. Cheng, T. Xia, and J. Xiao, "Aligntts: Efficient feed-forward text-to-speech system without explicit alignment," in *ICASSP*, 2020, pp. 6714–6718.

[17] K. J. Shih, R. Valle, R. Badlani, A. Lancucki, W. Ping, and B. Catanzaro, "Rad-tts: Parallel flow-based tts with robust alignment learning and diverse synthesis," in *ICML Workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models*, 2021.

[18] R. Badlani, A. Łańcucki, K. J. Shih, R. Valle, W. Ping, and B. Catanzaro, "One tts alignment to rule them all," in *ICASSP*, 2022, pp. 6092–6096.

[19] C. Miao, L. Shuang, Z. Liu, M. Chen, J. Ma, S. Wang, and J. Xiao, "Efficienttts: An efficient and high-quality text-to-speech architecture," in *ICML*, 2021, pp. 7700–7709.

[20] 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 preprint arXiv:2006.04558*, 2020.

[21] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, and M. Zhou, "Moboaligner: A neural alignment model for non-autoregressive tts with monotonic boundary search," *arXiv preprint arXiv:2005.08528*, 2020.

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

[23] J. Donahue, S. Dieleman, M. Bińkowski, E. Elsen, and K. Simonyan, "End-to-end adversarial text-to-speech," *arXiv preprint arXiv:2006.03575*, 2020.

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

[25] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[27] H. Tachibana, K. Uenoyama, and S. Aihara, "Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention," in *ICASSP*, 2018, pp. 4784–4788.