INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Reformer-TTS: Neural Speech Synthesis with Reformer Network

Hyeong Rae Ihm , Joun Yeop Lee, Byoung Jin Choi, Sung Jun Cheon, Nam Soo Kim

Department of Electrical and Computer Engineering and INMC,
Seoul National University, Seoul, South Korea

{hrim, jylee, bjchoi, sjcheon}@hi.snu.ac.kr, nkim@snu.ac.kr

Abstract

Recent End-to-end text-to-speech (TTS) systems based on the deep neural network (DNN) have shown the state-of-the-art performance on the speech synthesis field. Especially, the attention-based sequence-to-sequence models have improved the quality of the alignment between the text and spectrogram successfully. Leveraging such improvement, speech synthesis using a Transformer network was reported to generate human-like speech audio. However, such sequence-to-sequence models require intensive computing power and memory during training. The attention scores are calculated over the entire key at every query sequence, which increases memory usage. To mitigate this issue, we propose Reformer-TTS, the model using a Reformer network which utilizes the locality-sensitive hashing attention and the reversible residual network. As a result, we show that the Reformer network consumes almost twice smaller memory margin as the Transformer, which leads to the fast convergence of training end-to-end TTS system. We demonstrate such advantages with memory usage, objective, and subjective performance evaluation.

Index Terms: speech synthesis, attention-based TTS, Reformer network

1. Introduction

With the development of neural network techniques and high computing power, the performance of synthesized speech by text-to-speech(TTS) has been improved. Traditional TTS system is based on statistical parametric speech synthesis [1] such as HMM-based speech synthesis [2], SPSS using deep neural network [3]. These SPSS approaches are composed of text feature extraction module, duration model, acoustic model, and vocoder. In multi-stage approach, errors can accumulate over each stage which could deteriorate the synthesized speech quality. Furthermore, lots of linguistic and acoustic knowledge is needed to design the SPSS architecture. To resolve this problem, integrated end-to-end TTS models like Deep Voice[4, 5], Tacotrons [6, 7] has been proposed. These models also eliminate the need for the complex design of the feature extraction models. Especially, attention-based TTS models such as Tacotrons [6, 7], DCTTS [8], Transformer TTS [9] take place in end-to-end TTS. In the attention-based TTS, attention mechanism enables the model to learn the alignment between the text and spectrogram without explicit arrangement between text and spectrogram.

However, There are two main problems in the attention-based TTS: attention failure and memory problem. The attention failure occurs when alignment path do not have monotonic curve which result skipping and repeating. To overcome the attention failure, There have been attempts, such as forward attention[10] and guided attention[8].

Second, the memory problem is that the attention mechanism causes memory inefficiency in the training step. The attention score is calculated for the entire key sequence at every query samples. Hence, the memory complexity is $O(L_q L_k)$ where L_q and L_k are the length of query and key sequences, respectively. Especially, in the Transformer TTS, there are three stages that use the attention mechanism: two self attentions in text encoder and mel decoder and an encoder-decoder attention that aligns text with mel spectrogram. Therefore, a considerable amount of memory is required compared to the number of parameters while training. Since it is reported that large batch size is crucial for the stable training [9], training with small memory causes quality degradation of the synthesized speech.

In the Transformer network, the attention scores are almost zero except on a few key vectors. Thus, it is not necessary to calculate attention scores for the entire key sequences. In Reformer network [11], the existing attention mechanism is replaced with locality-sensitive hashing (LSH) [12] attention. In addition, the reversible residual network (RevNet)[13] was also used as an alternative to the residual network [14], which has the effect of refining the latent variables but has a memory disadvantage due to storing activation. It was reported that in the NLP task, the Reformer network has a positive effect on memory while maintaining the performance.

In this paper, we propose a Reformer TTS model where some of the self-attention modules are replaced with LSH self-attention, and reversible residual networks are introduced, which reduce the memory cost while maintaining the advantages of the Transformer network. In addition, we employed a forward attention [10] that easily forms monotonic alignment path. From the experimental evaluation, we demonstrate that the proposed system is efficient in terms of memory while the speech quality is not degraded compared to the Transformer TTS. We found that Reformer TTS can be trained on approximately half memory usage compared to Transformer TTS of similar parameter size and same batch size. This advantage leads to more robust and faster convergence during training.

2. Background

In this section, we first briefly describe the Transformer TTS model and the Reformer network.

2.1. Transformer TTS

Text-to-speech using Transformer network converts and input text sequence $(x_1, x_2, x_3, ..., x_N)$ into and acoustic feature sequence $(\boldsymbol{y_1}, \boldsymbol{y_2}, ..., \boldsymbol{y_T})$, by attention-based sequence-to-sequence [15] model. Similar to other general end-to-end TTS models, the Transformer TTS follows the encoder-decoder scheme, where text and mel spectrogram features $\boldsymbol{h}_n^x, \boldsymbol{h}_t^y$ are extracted from encoder and decoder respectively, and feature

http://dx.doi.org/10.21437/Interspeech.2020-2189

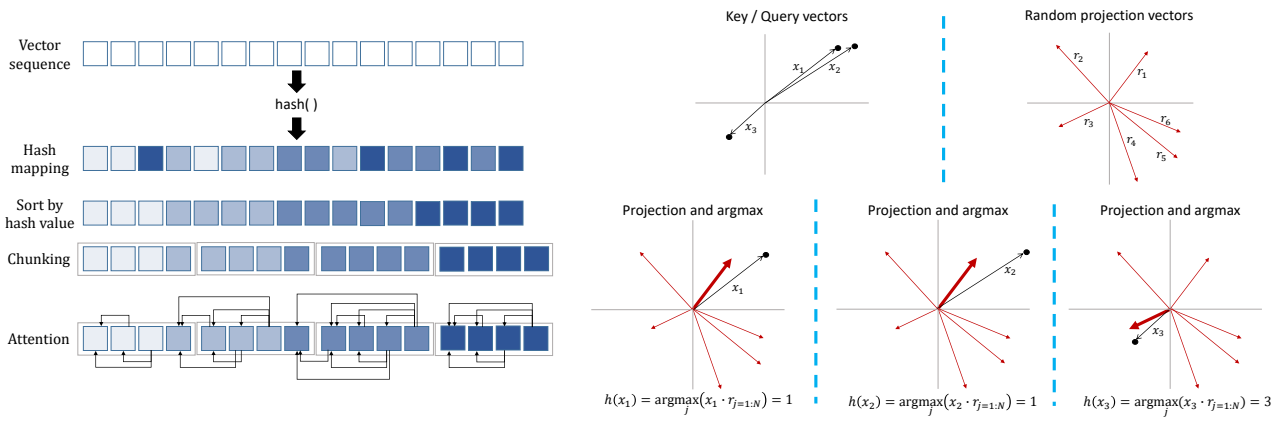

Figure 1: *procedure of locality-sensitive hashing attention*

alignment $\boldsymbol{\alpha}$ is performed through attention mechanism:

$$\boldsymbol{h}_n^x = Encoder(PreNet_{enc}(\boldsymbol{x_{1:N}})) \tag{1}$$

$$\boldsymbol{h}_t^y = PreNet_{dec}(\boldsymbol{y}_{1:t}) \tag{2}$$

$$\boldsymbol{\alpha}_t = Attn(\boldsymbol{h}_{t-1}^y, \boldsymbol{h}_{1:N}^x) \tag{3}$$

$$\boldsymbol{c}_t = \sum_i^N (\boldsymbol{\alpha}_t[i]\boldsymbol{h}_i^x) \tag{4}$$

$$\hat{\boldsymbol{y}}_t = Decoder(\boldsymbol{c}_t, \boldsymbol{h}_{:t-1}^y), \tag{5}$$

where N is the length of text, and n, t is sample indices of text and mel spectrogram. $PreNet$s which consume text and mel include feature extraction module and inject locational information by adding positional embedding. $Attn$ is multihead attention. $\boldsymbol{c}$ is attended context vector that is weighted sum of text encoder output. In Transformer TTS, $Enc()$ and $Dec()$ are stacks of the block which include the pairs of self-attention network and feedforward network:

$$\boldsymbol{\alpha}_{1:N}^E = Attn(\boldsymbol{x}_{1:N}, \boldsymbol{x}_{1:N}) \tag{6}$$

$$\boldsymbol{\alpha}_{1:T}^D = Attn(\boldsymbol{y_{1:T}}, \boldsymbol{y_{1:T}}) \tag{7}$$

$$\boldsymbol{h}_n^x = \sum_i^N (\boldsymbol{\alpha}_n^E(i)\boldsymbol{h}_i^x) \tag{8}$$

$$\boldsymbol{h}_t^y = \sum_j^T (\boldsymbol{\alpha}_t^D(j)\boldsymbol{h}_j^y), \tag{9}$$

where T is the length of mel spectrogram. With self-attention, the Transformer network can extract text and audio features by considering long term dependencies.

However, self-attention mechanism in Transformer network has memory redundancy. Attention score is computed by matrix multiplication $\boldsymbol{QK}^T$, where $\boldsymbol{Q}$ and $\boldsymbol{K}$ are the key and query vector sequence respectively. Therefore, attention module occupies $O(L_k L_q)$ memory complexity, given L_k and L_q are the lengths of key and query. However, although the attention score is calculated for the entire sequence, the dominant score is concentrated in small portion of the sequence. In addition, residual networks between input and output of the attention network require more significant memory occupancy during the backpropagation process.

2.2. Reformer network

Recently, the Reformer[11] is proposed as an alternative to an efficient Transformer using LSH attention and reversible residual network. LSH performs clustering on the key sequences to gather vectors with close distance into same cluster. The attention score is calculated only for the own cluster and adjacent to itself, which results the memory-efficient attention. A hashing method maps high dimensional vector x into a hash $h(x)$. x values with the same $h(x)$ are placed in the same cluster. Generally, the hash function output values are independent of how similar values the inputs are, but LSH uses a function that maps nearby vectors to the same hash values. For example,

$$h(\boldsymbol{x}) = argmax(\boldsymbol{x}\boldsymbol{R}_{+-}) \tag{10}$$

$$\boldsymbol{R}_{+-} = cat([\boldsymbol{R}, -\boldsymbol{R}]) \tag{11}$$

can be a locality-sensitive hash function, where R is a random projection matrix with size $[d_k, h/2]$, and d_k is the dimension of key vectors. In Figure 1, Three vectors x_1, x_2, and x_3 are projected into random projection vectors (h=6) and the highest projections are from r_1, r_2, and r_6, respectively. In such a way, there is a high probability that vectors at close range have the same hash outputs. Therefore, the probability of entering the same bucket also increases. After QK vectors in the sequence are clustered in hash buckets, the attention score is calculated with key sequence in the same buckets by masking the others:

$$\boldsymbol{\alpha}_i[j] = softmax(\sum_{i \in P} exp(\boldsymbol{q}_i \cdot \boldsymbol{k}_j - m(j, \boldsymbol{P_i})) \tag{12}$$

$$m(x, \boldsymbol{A}) = \begin{cases} 0, & \text{if } x \in \boldsymbol{A} \\ \infty, & \text{otherwise} \end{cases} \tag{13}$$

$$\boldsymbol{P_i} = \{j | h(\boldsymbol{q}_i) = h(\boldsymbol{k}_j)\} \tag{14}$$

$$\boldsymbol{c}_t = \sum_i^N (\boldsymbol{\alpha}_t[j]\boldsymbol{h}_j) \tag{15}$$

Feedforward network and residual network follow the attention module. The residual network is an effective method to overcome the vanishing gradient problem. However, It must store the activation in residual layers for updating the weight during backpropagation, which leads to the memory usage in proportion to the number of network repetition. Hence, RevNet can relieve the memory burden on multiple feedforward networks and attention networks.

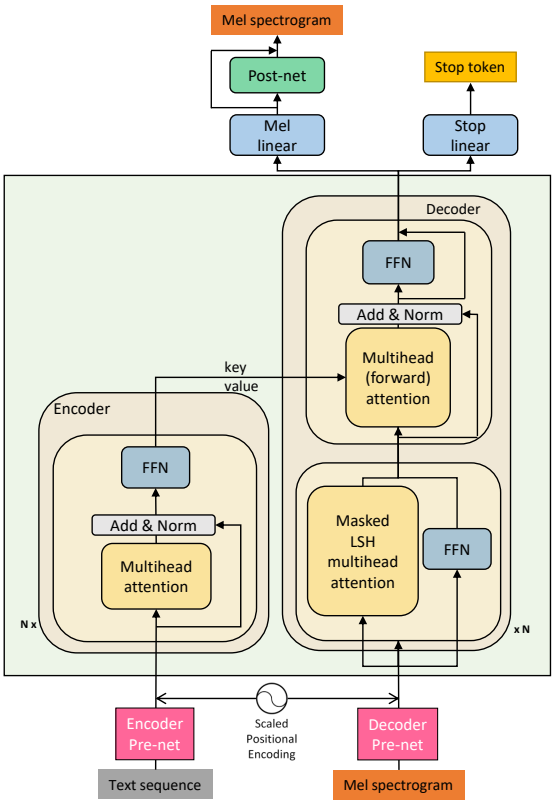

Figure 2: *Overall architecture of the proposed model*

3. Memory efficient Reformer TTS

We propose Reformer TTS, memory-efficient end-to-end speech synthesis system. In Transformer TTS, it is reported that batch size has an essential effect on stable training. The Reformer TTS can solve the problem by relieving memory redundancy through the LSH and RevNet. In this section, we will describe the structure of Reformer TTS model and the function of each part. The entire architecture of our model is shown in Figure 2.

3.1. Reformer TTS

3.1.1. feature extraction

In Reformer TTS, linguistic context information in text sequence is obtained through positional encoding which ensures the sequential locational information, character-level encoding with 512-dimension, and Pre-net sequentially. The encoder Pre-net network is comprised of three 1-dimensional convolution layer with 5 kernel size, to extract a 256-dimensional context sequence, followed by a batch normalization and ReLU activation. A linear network is added at the end of the encoder to compensate for the 0-centered positional encoding passing through the ReLU network, where the output ranges $[0, +\infty)$ Besides, the decoder pre-net network is composed of two linear projections of 256 dimensions, ReLU activations, and dropouts. In mel feature extraction, linear projection is also used behind the pre-net in order to match the dimension of the 0-centered issue as well as to fit the same dimension when calculating the attention. The positional encoding is also placed between the mel spectrogram and decoder prenet.

3.1.2. Encoder

In the encoder, LSH attention may cause the problem in alignment because the bucket size affects attention accuracy. If bucket size is set too small, it is difficult to predict which key vectors are attended accurately. Conversely, if it is set large to be close to the text length, the attention is equivalent to full-sight attention. In addition, the text has a shorter sequence length than mel spectrogram. We observed that there was no significant difference in memory between using and not using the LSH attention. Therefore, Text encoder is composed of the blocks of three self-attention layers and feedforward networks where residual network and layer normalization are included. Each hidden layer dimension is 256, and the output is sent to encoder-decoder attention in the decoder.

3.1.3. Decoder

As in the general end-to-end TTS scheme, mel spectrogram sequence is predicted by the decoder network, which consumes the character feature sequence. LSH attention is used for self-attention of the decoder, but full-length attention is used for encoder-decoder attention that follows. The LSH attention combines query and key sequences to determine which buckets to attend. However, the mel at query and text at key within close vectors have different information. Therefore, mel and text vectors are bucketed regardless of whether they are aligned with each other. Therefore, we did not use the LSH attention at the encoder-decoder attention and employed the same decoder block in the Transformer TTS. Same as Tacotron2, the mel spectrogram and stop token are predicted by two distinct linear projection networks. In addition, we give balancing weight to positive stop token because positive stop tokens are sparser than negative tokens over mel frames as used in Transformer TTS [9].

The LSH attention can significantly save memory since it is a partial attention. Because the LSH attention is unstable compared to the Transformer self-attention, there could be misaligned attention in the encoder-decoder network in the Reformer network. When using the dot product attention, the encoder-decoder module produced the result of poor alignment. Meanwhile, forward attention technique[10] has been used in end-to-end TTS to resolve misalignment problems. We use forward attention with the transition agent, which is a strategy explained from the products-of-experts model[16]. By applying this, the text feature was well-aligned with the speech feature, and as a result, it can be observed that the problem of repeating or skipping was resolved. With multihead attention, forward attention is processed as Algorithm 1. The procedure and the notation for Algorithm 1 is as follows:

$$head^{\boldsymbol{Q}}_{1:H} = Chunk(\boldsymbol{Q}, H) \tag{16}$$

$$head^{\boldsymbol{K}}_{1:H} = Chunk(\boldsymbol{K}, H) \tag{17}$$

$$head^{\boldsymbol{V}}_{1:H} = Chunk(\boldsymbol{V}, H) \tag{18}$$

$$Attn_{1:H} = softmax(\frac{\boldsymbol{Q}_{i:H}\boldsymbol{K}^{T}_{i:H}}{\sqrt{d_k}}) \tag{19}$$

In encoder-decoder attention, forward attention scheme is used in one layer of the decoder stack. In Transformer TTS, it is noticeable from the attention that alignment is not formed from all of the attention in the decoder stack. The attention layers in the rest of the decoder stacks generate the attention

Algorithm 1 Multihead forward attention

Initialize:

$\boldsymbol{\alpha}_{1:H,0}(1) \leftarrow 1.0$

$\boldsymbol{\alpha}_{1:H,0}(n) \leftarrow 0.0, n = 2, ..., N$

$\boldsymbol{u}_{1:H,0} \leftarrow 0.5$

for t=1 to T do:

$\boldsymbol{\alpha}'_{1:H,t}(n) \leftarrow ((1 - \boldsymbol{u}_{1:H,t-1})\boldsymbol{\alpha}_{1:H,t-1}(n) + \boldsymbol{u}_{1:H,t-1}\boldsymbol{\alpha}_{1:H,t-1}(n-1))Attn_{1:H,t}(n))$

$\boldsymbol{\alpha}_{1:H,t}(n) \leftarrow \boldsymbol{\alpha}'_{1:H,t}(n) / \sum_{i=1:N} \boldsymbol{\alpha}'_{1:H,t}(i)$

$\boldsymbol{c}_{1:H,t} \leftarrow \sum_{i=1:N} \boldsymbol{\alpha}_{1:H,t}(i) head^{\boldsymbol{V}}_{1:H,i}$

$\boldsymbol{u}_{1:H,t} \leftarrow DNN(\boldsymbol{c}_{1:H,t}, \boldsymbol{o}_{1:H,t-1}, head^{\boldsymbol{Q}}_{1:H,t})$

end for

scores regardless of the alignment path between text and mel spectrogram. Therefore, aligning is performed well by using the forward attention only for one attention layer among the decoder stack.

4. Experiments

In this section, we explain the training setup and experimental evaluations.

4.1. Experimental setup

In the training process, The LJSpeech [17], English speech dataset spoken by a female speaker is used. The database contains about 24-hour 13,100 utterances with a 22,050 Hz sampling rate. The mel spectrogram feature is 80-dimension extracted from 12.5 ms of hop length and 50 ms of frame length. The notations and symbols such as numbers in the dataset are re-written as pronounced in the text script. e.g., "1828" is written as "eighteen twenty-eight". 0-padding was done after the text and mel spectrogram sequences to fit the size of the bucket for LSH attention in the Reformer network. We also used WaveGlow[18] vocoder, which can reconstruct audio samples from the mel spectrogram in parallel based on generative flow network[19] trained on the ground truth mel spectrogram features and audio waveforms. We trained WaveGlow with 12 stacks of Glow depth and 8 layers of non-autoregressive kernel size 3 WaveNet in WaveGlow affine network. We used a TITAN RTX to the Reformer TTS and the baseline Transformer TTS. The largest batch size of allowed memory capacity was used. Compared to the Transformer TTS which could contain 32 batch size with slightly over 9 million parameters, The reformer TTS could be trained on 84 batch size on a single GPU with approximately 13 million parameters. We trained the model using the Adam optimizer [20] with β_1 = 0.9, β_2 = 0.999, $epsilon$ = 10^{-8} and a fixed learning rate of 0.0005.

4.2. Evaluation

We observed the number of the memory needed to train the models. The memory usages on the Reformer TTS and Transformer TTS given the number of model parameters, batch size, and the fixed length of text and mel spectrogram (256 and 1024, respectively) are shown in Table 1: The results show that the Reformer TTS model can be trained more efficiently in terms of memory. Through the results, LSH attention and reversible residual networks were used to configure the model to be able to use a batch size almost twice as large as the model with almost the same number of parameters.

For the perceptual subjective listening test, we also selected

The number of memory cached			
Model	Batch size	Number of parameters	Memory cached
Transformer	32	12.47×10^6	16.76×10^9
Reformer	32	12.37×10^6	10.59×10^9
Transformer	44	12.47×10^6	23.07×10^9
Reformer	84	12.37×10^6	24.20×10^9

Table 1: *Comparison of cached memory consuming on the Transformer TTS and Reformer TTS*

100 test text samples which are out of the dataset. 17 Korean native participants who are fluent in English listened to the speech samples generated by Reformer TTS and Transformer TTS. The order of the speech samples are randomly shuffled. Listeners are asked to evaluate the speech quality in terms of the comparative mean opinion score (CMOS). The listeners rated each sample in five categories: A is very bad, bad, normal, good, and very good compared to B and each scored from 1 to 5 points to calculate the CMOS score.

Model	CMOS	WER
Transformer with batch size 44	0.035	18.5%
Reformer with batch size 84	0	17.7%

Table 2: *CMOS test and WER on Reformer TTS and Transformer TTS*

In addition, for the objective test, we estimated word error rate (WER) using ASR model in Google API was used to verify the intelligibility of synthesized speech. Fastspeech[21] borrowed WER as a measure of whether synthesized speech pronounces well without skipping or repeating. This measure was employed to supplement the fact that because CMOS participants were not given text scripts, they may not have noticed that synthesized speech may contain omitted words. As shown in Table 2, the Reformer TTS model showed almost similar preference to Transformer TTS. By constructing a partial attention network using LSH attention and using a reversible residual network, the Reformer TTS has sufficient expressive power to predict mel spectrogram from text sequence. In addition, the WER was lower in the Reformer TTS, which is expected to be the effect of forward attention. Also, listeners can give high CMOS scores on speech including skipping word by Transformer TTS, because listeners evaluated with no given text scripts.

5. Conclusions and discussion

In this paper, we describe Reformer TTS, a memory-efficient end-to-end neural TTS system. The state-of-the-art neural TTS becomes more accessible and stable training of this becomes possible. The speech sample generated from the Reformer TTS shows the comparable quality to Transformer TTS.

Even though the Reformer TTS has enabled memory-efficient training, LSH attention contains sorting, which slow down the training speed. If the hash function with $O(1)$ time complexity were devised, faster training can be conducted.

6. Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No.2020-0-00059, Deep learning multi-speaker prosody and emotion cloning technology based on a high quality end-to-end model using small amount of data)

7. References

[1] A. W. Black, H. Zen, and K. Tokuda, "Statistical parametric speech synthesis," in *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 4. IEEE, 2007, pp. IV–1229.

[2] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for hmm-based speech synthesis," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*, vol. 3. IEEE, 2000, pp. 1315–1318.

[3] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *2013 ieee international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 7962–7966.

[4] S. Ö. Arik, M. Chrzanowski, A. Coates, G. Diamos, A. Gibiansky, Y. Kang, X. Li, J. Miller, A. Ng, J. Raiman *et al.*, "Deep voice: Real-time neural text-to-speech," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 195–204.

[5] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," *arXiv preprint arXiv:1710.07654*, 2017.

[6] 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," *Proc. Interspeech 2017*, pp. 4006–4010, 2017.

[7] 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 *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.

[8] H. Tachibana, K. Uenoyama, and S. Aihara, "Efficiently trainable text-to-speech system based on deep convolutional networks with guided attention," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4784–4788.

[9] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, "Neural speech synthesis with transformer network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 6706–6713.

[10] J.-X. Zhang, Z.-H. Ling, and L.-R. Dai, "Forward attention in sequence-to-sequence acoustic modeling for speech synthesis," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4789–4793.

[11] N. Kitaev, L. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," in *International Conference on Learning Representations*, 2019.

[12] A. Andoni, P. Indyk, T. Laarhoven, I. Razenshteyn, and L. Schmidt, "Practical and optimal lsh for angular distance," in *Advances in neural information processing systems*, 2015, pp. 1225–1233.

[13] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse, "The reversible residual network: Backpropagation without storing activations," in *Advances in neural information processing systems*, 2017, pp. 2214–2224.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[16] G. E. Hinton, "Products of experts," 1999.

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

[18] R. Prenger, R. Valle, and B. Catanzaro, "Waveglow: A flow-based generative network for speech synthesis," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3617–3621.

[19] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, 2018, pp. 10 215–10 224.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] 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, pp. 3171–3180.