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

MoBoAligner: a Neural Alignment Model for Non-autoregressive TTS with Monotonic Boundary Search

Naihan Li[1], *Shujie Liu*[2], *Yanqing Liu*[3], *Sheng Zhao*[3], *Ming Liu*[1] *and Ming Zhou*[2]

[1]University of Electronic Science and Technology of China
[2] Microsoft Research Asia, Beijing, China
[3]Microsoft STC Asia, Beijing, China

lnhzsbs1994@163.com, {shujliu, yanqliu, szhao, mingzhou}@microsoft.com,
csmliu@uestc.edu.cn

Abstract

To speed up the inference of neural speech synthesis, non-autoregressive models receive increasing attention recently. In non-autoregressive models, additional durations of text tokens are required to make a hard alignment between the encoder and the decoder. The duration-based alignment plays a crucial role since it controls the correspondence between text tokens and spectrum frames and determines the rhythm and speed of synthesized audio. To get better duration-based alignment and improve the quality of non-autoregressive speech synthesis, in this paper, we propose a novel neural alignment model named MoBoAligner. Given the pairs of the text and mel spectrum, MoBoAligner tries to identify the boundaries of text tokens in the given mel spectrum frames based on the token-frame similarity in the neural semantic space with an end-to-end framework. With these boundaries, durations can be extracted and used in the training of non-autoregressive TTS models. Compared with the duration extracted by TransformerTTS, MoBoAligner brings improvement for the non-autoregressive TTS model on MOS (3.74 comparing to FastSpeech's 3.44). Besides, MoBoAligner is task-specified and lightweight, which reduces the parameter number by 45% and the training time consuming by 30%.

Index Terms: text to speech, spectrum alignment, monotonic alignment.

1. Introduction

Speech synthesis (text to speech, TTS) plays a pivotal role with a wide application in the speech-related scenario. After a step made from statistic and parametric TTS models [1, 2, 3, 4], neural TTS models become the mainstay owing to the advance of deep learning. There are two modules in the neural TTS pipeline, the acoustic model to convert the input text to spectrum, and the vocoder to synthesize the audio conditioning on the spectrum. The vocoders include Wavenet [5], Parallel Wavenet [6], WaveRNN [7], WaveGlow [8], Parallel WaveGAN [9], etc. As for the acoustic models, Tacotron2 [10] and TransformerTTS [11] are two sequence-to-sequence based acoustic models that can synthesize mel spectrums in an autoregressive manner. The autoregressive dependencies between spectrum frames constrain parallel computing, resulting in low efficiency during inference. Therefore, a non-autoregressive architecture named FastSpeech [12] is proposed.

FastSpeech employs a novel Transformer-based model, which is called "Feed-forward Transformer", as shown in Fig. 1. An encoder is firstly leveraged to project the input text into hidden states, which are expanded according to the token durations. After that, the expanded hidden states are consumed by the decoder to generate the mel spectrum. To expand the encoder hidden states and train the duration predictor in the training procedure of non-autoregressive TTS models, token durations are needed as shown in the top of Fig. 1. These durations play critical roles for two reasons: 1) they directly control the correspondence between text tokens and spectrum frames, thus the decoder learns to pronounce the given token and back-propagate the gradient to corresponding encoder hidden states during training; 2) accurate duration supervision for the duration predictor can benefit the rhythm and speed of the synthesized audio during inference.

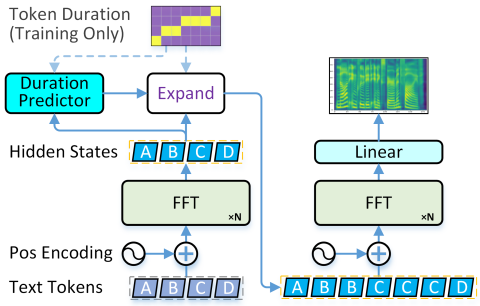

Figure 1: *Feed-forward Transformer. Schematically, the input and the token duration is the same as it is in Fig. 2, which is processed by the left FFT (Transformer encoder), expanded and fed into the right FFT (Transformer decoder) to generate the mel spectrum. The token duration is required only for training.*

FastSpeech leverages a well-trained autoregressive TransformerTTS to provide the duration of each text token by the encoder-decoder attention. Specifically, based on the similarity to a diagonal matrix, one alignment matrix is selected from all the alignments of the multi-head encoder-decoder attention. In this alignment, one frame is counted to a text token if this token has the largest share in this frame's attention weights. However, since the attention mechanism in TransformerTTS is not designed for duration prediction, using it as supervision to train the duration predictor brings two problems: 1) The attention weights of each frame are not concentrated on one single token; instead, they are dispersed to obtain a broad context of the input text. Although the text token being pronounced is allocated with the largest share in most cases, there is still some noise which can disturb the extracted duration. 2) For the speech synthesis task, the alignment of the encoder and decoder hidden states should be monotonic, and the aligned frames of one token should be continuous. The alignment of TransformerTTS sometimes violates these two properties, causing poor duration accuracy. With inaccurate duration as training data, the duration

http://dx.doi.org/10.21437/Interspeech.2020-1976

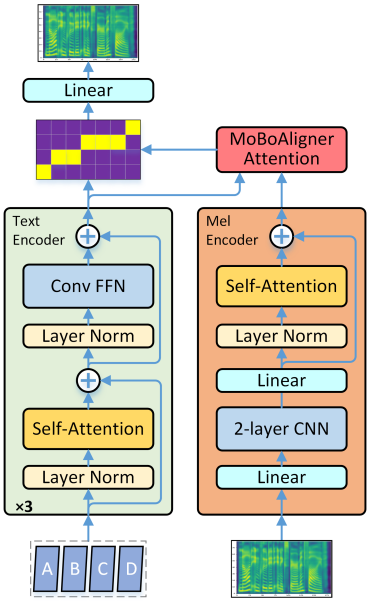

Figure 2: *The architecture of MoBoAligner. Schematically, there are four tokens A, B, C and D in the input text, of which the durations are 1, 2, 3 and 1 as shown in the alignment.*

predictor may be biased and harm the rhythm of the synthesized audio, and the wrong encoder hidden states may be used to reconstruct the mel spectrum.

Following the properties of monotonicity and continuity, we are inspired by the monotonic alignment [13], and propose MoBoAligner, a novel monotonic aligner for non-autoregressive TTS. MoBoAligner leverages two separate encoders to project both the input text and the mel spectrum into the same semantic space, after that these two sequences are aligned with a novel attention mechanism named MoBoAligner attention. MoBoAligner attention can monotonically scan the mel spectrum frames according to the text tokens, and spot the token boundaries in these frames. According to the alignment, the text hidden states are expanded to frame-level and reconstruct the mel spectrum in an end-to-end framework, which is under the instruction of Mean Squared Error (MSE) loss. To evaluate the quality of extracted durations and evaluate our proposed MoBoAligner, we train non-autoregressive models with different duration sources and conduct mean opinion score (MOS) tests. Our contributions are summarized as follows:

1. We propose MoBoAligner, a neural alignment model which can monotonically scan the mel spectrum and search the boundaries of the given text token sequence, and form the text-to-mel alignment based on these boundaries.

2. The durations extracted by MoBoAligner are more accurate, which brings an improvement for the non-autoregressive TTS model on MOS (3.74 comparing to FastSpeech's 3.44).

3. Comparing to TransformerTTS, our model is task-specified and lightweight, which reduces the parameter number by 45% and reduces the training time consuming by 30%.

2. MoBoAligner

As mentioned in the introduction, the non-autoregressive TTS model requires additional duration inputs, obtained by a TransformerTTS as designed in FastSpeech, which may not be suitable for this task and have some disadvantages. To generate better alignment results for the training of duration predictor, in this section, we propose a novel neural aligner MoBoAligner, which can monotonically search the text token boundaries in the mel spectrum, as shown in Fig. 2. In MoBoAligner, the input text and mel spectrum are processed by the text encoder and mel encoder respectively. After that, MoBoAligner attention (will be introduced in Section 3) is employed to monotonically align these two sequences and produce the alignment. Based on the alignment, the text hidden states are expanded to frame-level and reconstruct the mel spectrum by linear projection. Mean squared error (MSE) loss is employed as cost function, and the alignment between the text and the mel spectrum is automatically learned to minimize this loss. In the following of this section, we will introduce the text and mel encoders, and our MoBoAligner attention will be introduced in Section 3

2.1. Text encoder

Each text token is firstly embedded with a 512-dim vector, which is added with a scaled positional embedding is added, then fed into three Transformer FFT blocks. Here we change the fully-connected (FC) layers in the FFN with convolutional layers, of which the kernel size is 3. All attention sizes are 512, the head number is 8, and the hidden size of Conv FFN is 2048.

2.2. Mel encoder

The mel spectrum is processed by a 2-layer CNN (channel number is 256, dilation is 2, kernel size is 3, followed by ReLU and dropout layers), which is designed for extracting the local context and strengthening the boundary information. A leading linear projection (unit size from 80 to 256, followed by the ReLU and dropout layers) and a tailing linear projection (unit size from 256 to 512) are used for dimension consistency, and a scaled positional embedding is added after the tailing linear projection. Finally, a self-attention layer is added to provide a sequence-level context.

3. MoBoAligner attention

Based on the duration prediction of the input tokens, MoBoAligner attention tries to align the text input and the mel spectrum frames, using the hidden states from text and mel encoders. As shown in Fig. 3, text token boundaries are monotonically searched in mel spectrum, then the alignments of frames between boundaries are filled by copying the alignments of the corresponding boundaries.

3.1. Attention formulation

In this section, we introduce our novel attention mechanism to monotonically search the text token boundaries in the mel spectrum. Let $\{x_i\} = x_1, x_2, ..., x_I$ be the text token sequence with the length I, and $\{y_j\} = y_1, y_2, ..., y_J$ be the corresponding mel spectrum frames with the length J. Following the attention mechanism [14], scaled dot production is used to calculate the energy $e_{i,j}$ of each text token and mel frame pair:

$$e_{i,j} = \exp(\frac{Q_i \cdot K_j}{\sqrt{\mathbf{d}_{QK}}}) \quad (1)$$

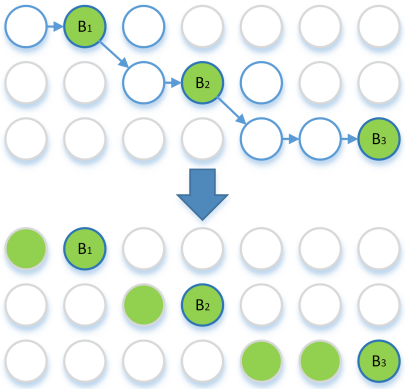

Figure 3: *Illustration of the MoBoAligner attention.*

where $\mathbf{d}_{QK}$ is the dimension of query Q_i and key K_j. Here we take text token x_i as query and mel frame y_j as key. Supposing that the max duration number of a text token is D (we use $D = 20$ in our experiments), the boundary probability $\alpha_{i,j}$ that the mel frame y_j is the boundary of the text token x_i can be calculated as:

$$\begin{aligned}\alpha_{i,j} =& P(B_i = j) \\ =& \sum_{k=max(j-D,0)}^{j-1} P(B_{i-1} = k)P(B_i = j|B_{i-1} = k)\end{aligned} \tag{2}$$

where $P(B_{i-1} = k)$ is the probability of that the text token x_{i-1} stops at frame y_k, and $P(B_i = j|B_{i-1} = k)$ is the conditional probability that y_j is the boundary of x_i given the boundary (y_k) of the previous text token x_{i-1}. The conditional boundary probability $P(B_i = j|B_{i-1} = k)$ is calculated as:

$$P(B_i = j|B_{i-1} = k) = \frac{e_{i,j}}{\sum_{m=k+1}^{min(k+D,J)} e_{i,m}}. \tag{3}$$

The probability of the special case that the boundary index is 0 is defined as:

$$P(B_{i'} = 0) = \begin{cases} 1 & \text{if } i' = 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Following the properties of monotonicity and continuity, the alignment probability $\beta_{i,j}$, indicating y_j is aligned to x_i, is the probability that the boundary of x_i is not before y_j, and the boundary of x_{i-1} is before y_j:

$$\begin{aligned}\beta_{i,j} =& P(B_{i-1} < j \leq B_i) \\ =& \sum_{k=max(j-D,0)}^{j-1} P(B_{i-1} = k)P(B_i \geq j|B_{i-1} = k)\end{aligned} \tag{5}$$

where

$$P(B_i \geq j|B_{i-1} = k) = \sum_{j'=j}^{min(k+D,J)} P(B_i = j'|B_{i-1} = k) \tag{6}$$

Based on the alignment probability β, we can tile the text hidden states to get the decoder input for the j^{th} frame as:

$$\tilde{h_j} = \sum_{i=1}^{I} \beta_{i,j} h_i \tag{7}$$

3.2. Encourage discreteness

To extract the text token boundaries in the mel spectrum, we want to concentrate the frames to only one text token. Following Gumbel-Softmax [15, 16], two measures (the Gumbel noise G_i and the temperature τ_i) are introduced, and the energy $e_{i,j}$ in Eq. 1 is rewritten as:

$$e^G_{i,j} = \exp(\frac{Q_i \cdot K_j}{\sqrt{\mathbf{d}_{QK}}\tau_i} + \frac{G_{i,j}}{\tau_i}) \tag{8}$$

The Gumbel noise G_i is transformed by a random variable sampled from a uniform distribution:

$$\begin{aligned}U_{i,j} &\sim U(0, 1) \\ G_{i,j} &= -log(-log(U_{i,j}))\end{aligned} \tag{9}$$

The temperature τ_i is a non-negative scalar to control the sharpness of the distribution: the closer τ_i is to 0, the sharper this distribution becomes . Instead of using τ_i anneals from 1 to 0.1, we use $\tau_i \sim U(0.1, \tau_{max})$, where τ_{max} linearly anneals from 1 to 0.1 in the training procedure, to make the discreteness process smoother and more efficient.

3.3. Frame interlacement for acceleration

Since the adjacent mel frames are similar, to improve the training speed, we select the first one out of every 2 frames to calculate the energy e, the boundary probability α and the alignment probability β, which are then recovered to the original length by inserting 0 to the intervals for α, and repeating each frame twice for β.

3.4. Inference

Instead of sampling a random number for τ_i, a fixed number 0.1 is used during the inference, and the Gumbel noise is removed. Besides, after each boundary probability $\alpha_{i,j}$ is calculated, an $onehot(\cdot)$ function is used to select the frame with the highest mass as the hard boundary for x_i:

$$\hat{\alpha}_{i,j} = onehot(\alpha_{i,j}) \tag{10}$$

then $\hat{\alpha}_i$ is used to calculate the boundary probabilities α_{i+1} for the next text token.

To ensure $\sum_{i=1}^{I} d_i = J$, where d_i is the duration of x_i, we set the duration of the last text token as $d_I = J - \sum_{i=1}^{I-1} d_i$. Since there is another constraint that the text token durations should be smaller than the maximum duration D, the sample will be abandoned if $d_I > D$ during inference, and we use all the success samples for the non-autoregressive TTS model training. Such a process can filter bad samples with noise from the training data.

4. Experiment

In this section, we train our proposed new aligner MoBoAligner and extract text token durations, based on which a Feed-forward Transformer [12] (same as that in FastSpeech) is trained to synthesize the mel spectrum and generate the audio. Mean option score (MOS) is leveraged for the evaluation.

4.1. Dataset

Our experiments are conducted on LJSpeech dataset [17], which includes 13,100 clips (splitted into 12,600/250/250 parts for train/dev/eval respectively) and totally 24 hours.

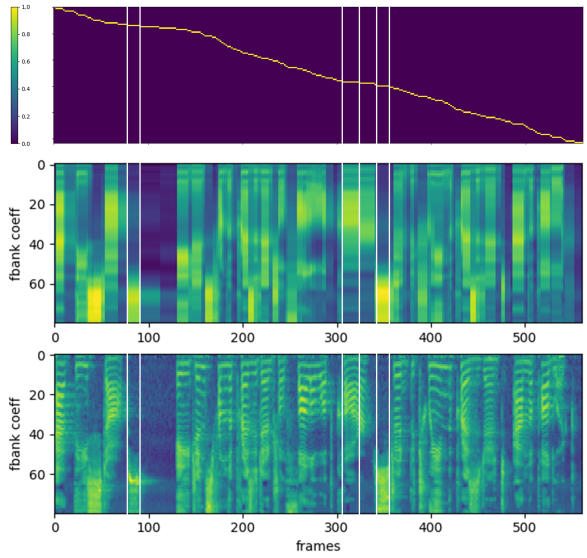

Figure 4: *Visualization of aligning results. The top figure is the alignment matrix, the middle one is the reconstructed mel by the linear layer in MoBoAligner, and the bottom one is the ground truth mel spectrum.*

Model	MOS	CI
Recording	4.61	0.10
GT Mel Wavs	4.48	0.07
TransformerTTS	4.18	0.09
FastSpeech	3.44	0.10
MoBoAligner TTS	3.74	0.10

Table 1: *MOS test results. "GT Mel Wavs" is the synthesized result with the ground truth mel spectrum. TransformerTTS is the autoregressive model. FastSpeech and MoBoAligner TTS are two non-autoregressive TTS models both using the Feed-forward Transformer as the TTS model, while FastSpeech uses the duration from TransformerTTS, and MoBoAligner TTS uses that of our proposed MoBoAligner. "CI" means the confidence interval radius with confidence level 0.95.*

4.2. Setup

Our MoBoAligner is implemented based on ESPnet [18, 19], an open-sourced repository available from GitHub, in which pretrained vocoders are also available. We use *ljspeech.wavenet.mol.v1*, a pretrained Wavenet, as our vocoder. We also implement a "DistributedDataParallel" model in pytorch to improve the training speed. The frame size of each batch is 15000 (80 samples per batch on average). Adam optimizer is employed for model training, with the same setting and the Noam decay method [14]. Our model is trained for 400 epoches, and the total training time is 14 hours ($\sim$1 iter/sec) with 4 Tesla P40 GPUs (24G RAM), which reduces 30% training time comparing to TransformerTTS (20 hours).

4.3. Visualization of aligning results

As shown in Fig. 4, the aligning results can be visualized by the alignment matrix and the reconstructed mel spectrum generated by the linear projection layer in MoBoAligner. From the alignment matrix, we can find that the two properties of monotonicity and continuity are well obeyed: each frame focuses on one text token, and all text tokens are covered by continuous frames. As for reconstructed mel spectrum, although the reconstructed mel spectrum is blurred, the text token boundaries are still clear and accurate, from which it can be derived that MoBoAligner has a tremendous ability in locating the token boundary.

4.4. TTS model and baselines

With the extracted text token durations by MoBoAligner, a Feed-forward Transformer is trained to evaluate the duration quality. We compare it with FastSpeech, in which TransformerTTS is used to provide the duration. To exclude the effect of the vocoder, we also include a group of waveforms synthesized by the same vocoder conditioning on the ground truth mel spectrum. Our Feed-forward Transformer employs the same architecture as FastSpeech, and together with TransformerTTS, they are of the same attention size (384), attention head number (2), and FFT hidden size (1536).

4.5. Result

The results of the mean option score (MOS) are shown in Table 1. The result with ground truth mel spectrum (GT Mel Wavs) achieves close quality with recordings followed by the autoregressive model TransformerTTS. Two non-autoregressive models are worse than TransformerTTS since the synthesized audios sound hoarse sometimes. Leveraging the same Feed-forward TTS model, replacing the TransformerTTS aligner with our proposed MoBoAlignercan still get a significant improvement (3.74 vs 3.44) on the MOS score.

4.6. Comparison with Forced alignment

To extract the text token duration, there is another way called Forced alignment. We also employ an internal HMM-based Forced alignment model, trained with the speech recognition data of $\sim$10,000 hours, and adapted with LJSpeech dataset. Although this Forced alignment tool uses larger training data and more complex training procedures, the final result using MoBoAligner achieves the same MOS with that using this Forced alignment tool.

5. Conclusions

In this paper, we propose MoBoAligner, which employs a novel attention mechanism to monotonically search the text token boundaries in the mel spectrum, and extract durations of high accuracy. Our model is trained under the instruction of MSE loss with an end-to-end framework. The ability in locating token boundaries can be proven in two aspects. On the one hand, the reconstructed mel spectrum has clear token boundaries among frames, although the detail is blurred. On the other hand, our model provides better durations which is revealed by the improvement of the non-autoregressive TTS model on MOS (3.74 comparing to FastSpeech's 3.44). Besides, our model is task-specified and lightweight, which reduces the parameter number by 45% and the training time consuming by 30%.

6. References

[1] K. Tokuda, T. Yoshimura, T. Masuko, T. Kobayashi, and T. Kitamura, "Speech parameter generation algorithms for hmm-based speech synthesis," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 1315–1318.

[2] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, no. 11, pp. 1039–1064, 2009.

[3] H. Ze, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7962–7966.

[4] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura, "Speech synthesis based on hidden markov models," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1234–1252, 2013.

[5] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[6] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg *et al.*, "Parallel wavenet: Fast high-fidelity speech synthesis," *arXiv preprint arXiv:1711.10433*, 2017.

[7] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. v. d. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," *arXiv preprint arXiv:1802.08435*, 2018.

[8] 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.

[9] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," *arXiv preprint arXiv:1910.11480*, 2019.

[10] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan *et al.*, "Natural tts synthesis by conditioning wavenet on mel spectrogram predictions," *arXiv preprint arXiv:1712.05884*, 2017.

[11] N. Li, S. Liu, Y. Liu, S. Zhao, M. Liu, and M. Zhou, "Neural speech synthesis with transformer network," *arXiv preprint arXiv:1809.08895*, 2018.

[12] 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. 3165–3174.

[13] C. Raffel, M.-T. Luong, P. J. Liu, R. J. Weiss, and D. Eck, "Online and linear-time attention by enforcing monotonic alignments," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2837–2846.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[15] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[16] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," *arXiv preprint arXiv:1611.01144*, 2016.

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

[18] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "Espnet: End-to-end speech processing toolkit," in *Interspeech*, 2018, pp. 2207–2211. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1456

[19] T. Hayashi, R. Yamamoto, K. Inoue, T. Yoshimura, S. Watanabe, T. Toda, K. Takeda, Y. Zhang, and X. Tan, "Espnet-tts: Unified, reproducible, and integratable open source end-to-end text-to-speech toolkit," 2019.