SPTTS: Parallel Speech Synthesis without Extra Aligner Model

Zeqing Zhao*‡, Xi Chen*†‡, Hui Liu*, Xuyang Wang*, Lin Yang* and Junjie Wang*
* AI Lab, Lenovo Research, Beijing, China
E-mail: zhaozq14@lenovo.com
† E-mail: chxsunshine@gmail.com

***Abstract*—In this work, we develop a novel non-autoregressive TTS model to predict all mel-spectrogram frames in parallel. Different from the previous non-autoregressive TTS methods, which typically require an external aligner implemented by an attention-based autoregressive model, our model can be optimized jointly without sophisticated external aligners. Motivated by the CTC-based speech recognition, which is a simple and effective manner to achieve the frame-level forced-alignment between the speech and text, our main idea is to consider the aligner learning of TTS as a CTC-based speech recognition like task. Specifically, our model learns the alignment generator by adopting the CTC-loss, to provide supervision for the duration predictor learning on the fly. In this way, we are able to learn a one-stage TTS system by optimizing the aligner with the feed forward transformer jointly. In inference phase, the aligner is removed and the duration predictor is used to predict duration sequence for synthesizing speech. To demonstrate our method, we conduct extensive experiments on an open-source Chinese standard Mandarin speech dataset[1]. The results show that our method achieves competitive performance compared with counterpart models (*e.g.* FastSpeech: a well-known non-autoregressive with extra aligner) in terms of the synthesized speech quality and robustness.**

I. Introduction

Thanks to the success of neural network, the text-to-speech (TTS) task has make tremendous progress recently. The current neural network based speech synthesis methods typically consist of two parts: the acoustic model and the vocoder. The acoustic model aims to generate mel-spectrogram from the text and the vocoder is used to convert the mel-spectrogram into speech. In this work, our goal is to improve the current speech synthesis system through a simplified and effective *acoustic model* without the external aligner.

Early works focus on the autoregressive attention model [1, 2, 3], which usually use the sequence-to-sequence model with attention mechanism [4] to generate the mel-spectrogram in a frame-by-frame style. These methods improve the quality of synthesized speech significantly, however, the Tacotron [1] and Tacotron2 [2] have low training efficiency due to the recurrent neural network, and the TranformerTTS [3] still suffers from slow inference because of the autoregressive inference style.

More recent efforts aim to resolve the autoregressive inference problem by developing various structures. [5, 6, 7] use the *variational auto-encoder* structure to gain the latent duration and generate the mel-spectrogram in inference. Nevertheless, this kind of method is difficult to control speed manually. An alternative strategy (*e.g.,* Fastspeech models [8, 9], DurIAN [10], Fastpitch [11] and Parallel Tacotron [12]) introduces a neural duration model to produce hard alignment, which could be used to synthesize mel-spectrogram in parallel.

Several types of alignment generation strategy have been proposed, to provide supervision for learning the duration model. [8] extracts the alignments from a pre-trained autoregressive attention-based TTS model. [10], [12] generate the alignment by learning a external HMM-based automatic speech recognition (ASR) model. In addition, montreal forced alignment (MFA) tool can also be used to extract phone duration. However, these non-autoregressive TTS methods typically have complicated pipeline and require to generate alignments from a pre-trained model or a forced-alignment tool.

Thus, how to get rid of the external aligner has attracted much attention recently. JDI-T [13] trains a non-autoregressive model jointly with an autoregressive model. AlignTTS [14] and GlowTTS [15] utilize the statistical features and forward-backward algorithms to get the alignment in training stage, where [15, 16] propose a flow-based method for generating mel-spectrogram, [14] uses the the Baum-Welch algorithm to design an alignment loss with the goal of maximize the likelihood, and proposes four different training stages for different model parts, which has a complicated training process and long training time. In this work, we propose the *Simplified Parallel Text-to-Speech* (SPTTS), a non-autoregressive TTS method which has a concise training pipeline without a pre-trained external aligner. Specifically, to maintain the advantage of parallel inference and controllable speed, our method adopts a duration model to predict the alignment between phone and mel-spectrogram. To learn the duration model, we propose a novel alignment generator module. Inspired by the connectionist temporal classification (CTC) based speech recognition [17, 18, 19], which is a simple and effective manner to achieve the frame-level forced-alignment between speech and text, the alignment generator is trained by adopting the CTC-loss. It provides the alignment of phone and mel-spectrogram, which served as the supervision for the duration predictor training on the fly. The overall system has a simplified pipeline and all

‡Equal contribution
[1]https://www.data-baker.com/open_source.html

components could be optimized jointly[2].

The main contributions of the paper are as follows:

- We propose a recognition based module as an internal aligner for TTS model, which can be used to provide supervision for the duration model;
- The internal aligner can be optimized with other components jointly, which simplifies the training pipeline of the non-autoregressive TTS model;
- The results show that our method achieves competitive performance compared with previous works in terms of the synthesized speech quality and robustness.

II. Model Architecture

To simplify the speech synthesis pipeline, a novel internal alignment generator is proposed by adding a recognition-based module in TTS model, which can obtain the correspondence between text sequence and acoustic features. With this, a non-autoregressive TTS model can be trained with the alignment generated by an internal aligner in a unified framework. The architecture of the proposed SPTTS is shown in Fig. 1, which consists of three main components: *1) attention-based synthesizer*, *2) duration predictor* and *3) alignment generator*. The details of each component are below.

A. Attention-based Synthesizer

The input of attention-based synthesizer is phone sequence and the output is mel-spectrogram. It is based on a Feed-Forward Transformer (FFT) like [8]. The synthesizer contains five parts: *1) phone embedding layer*, *2) lower FFT blocks*, *3) length regulator*, *4) higher FFT blocks* and *5) linear layer*. The structure of FFT block is shown in Fig. 2(a).

Assuming that the input phone sequence is $\boldsymbol{C} = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_L)$, $\boldsymbol{c}_i \in \mathbb{R}^{1\times v}$, v represent the number of symbols, L is the phone sequence length.

First, the phone embedding layer embeds the phone sequence to embedding features, followed by the lower FFT blocks, which map the embedding features to phone-level hidden states, which is denoted as $\boldsymbol{H}_{pho} = \{\boldsymbol{h}_1, \ldots, \boldsymbol{h}_L\} \in \mathbb{R}^{L\times d}$, d is the number of hidden units.

Then, the length regulator expands the hidden states $\boldsymbol{H}_{pho}$ with the phone duration sequence $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_L\} \in \mathbb{R}^{L\times 1}$ predicted by duration predictor in repeat way, then the expanded hidden states $\boldsymbol{H}_{mel} = \{\boldsymbol{h}_1, \boldsymbol{h}_1, \ldots, \boldsymbol{h}_T\} \in \mathbb{R}^{T\times d}$ is generated, where T is the length of mel-spectrogram. For example, let $L = 3$ and the duration sequence $\boldsymbol{D} = \{1, 2, 2\}$, the output of length regulator is $\{\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_2, \boldsymbol{h}_3, \boldsymbol{h}_3\}$.

Finally, the higher FFT blocks and the linear layer decode the $\boldsymbol{H}_{mel}$ to predict mel-spectrogram $\boldsymbol{M}' = \{\boldsymbol{m}'_1, \ldots, \boldsymbol{m}'_T\}$, $\boldsymbol{m}'_i \in \mathbb{R}^{1\times n}$, n is the dimension of mel features.

B. Duration Predictor

Inspired by [8, 10], we adopt the duration model to generate the duration sequence to regulate the length of phone. In this way, we are able to generate the mel-spectrogram in parallel and control the speed of speech easily. The duration predictor takes hidden states generated by the lower FFT blocks as inputs, and predicts the duration sequence $\boldsymbol{D}'$ for each phone. The duration predictor is typically implemented by two sequential 1D convolution and a linear layer, which is illustrated in Fig. 2(c).

C. Alignment Generator

To learn the duration model, we propose a novel alignment generator. Inspired by the CTC-based speech recognition [17, 18, 19], which is a simple and effective manner to achieve the frame-level forced-alignment between speech and text. We introduce a CTC-based recognition module as the internal alignment generator to provide the alignment of mel-spectrogram and phone sequence, served as the supervision for the duration predictor learning on the fly.

The inputs of the alignment generator are ground truth mel-spectrogram $\boldsymbol{M} = \{\boldsymbol{m}_1, \ldots, \boldsymbol{m}_T\}$, $\boldsymbol{m}_i \in \mathbb{R}^{1\times n}$. The outputs are frame-level posterior probability $\boldsymbol{Y} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T\}$, $\boldsymbol{y}_i \in \mathbb{R}^{1\times(v+1)}$ with an extra _ label. The _ label is a special token without pronunciation. To compute the CTC loss, there defines a many-to-one map $\mathcal{B}$: $\mathcal{B}(a, \_, a, b, \_) = \mathcal{B}(\_, a, \_, \_, a, b, b) = (a, a, b)$, the repeated labels and all _ labels are removed. For given phone sequence $\boldsymbol{C} = (\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_L)$, we can use the map $\mathcal{B}$ define the conditional probability as Eq. 1, where $\mathcal{B}^{-1}$ is the set of all paths whose $\mathcal{B}$ mapping result is $\boldsymbol{C}$.

$$p(\boldsymbol{C} \mid \boldsymbol{M}) = \sum_{\pi\in\mathcal{B}^{-1}(\boldsymbol{C})} p(\pi \mid \boldsymbol{M}) \tag{1}$$

The CTC loss is defined as a negative log likelihood in Eq. 2.

$$\mathcal{L}_{ctc} = -log\left(p(\boldsymbol{C} \mid \boldsymbol{M})\right) \tag{2}$$

We can calculate the conditional probability directly following Eq. 1, but there are always a large number of path corresponding to given labeling, which could be more complicated when computing the gradient. For efficient computing, the problem can be solved by CTC forward-backward algorithm, which applies dynamic programming ideas.

For each sequence $\boldsymbol{q}$ of length $\boldsymbol{r}$, can be split to $\boldsymbol{q}_{1:p}$ and $\boldsymbol{q}_{r:p}$. Considering the _ label, the sequence $\boldsymbol{C}$ is expanded to $\boldsymbol{C}' = (\_, \boldsymbol{c}_1, \_, \boldsymbol{c}_2, \_, \ldots, \_, \boldsymbol{c}_L, \_,)$. Then for sequence $\boldsymbol{C}'$, the forward variable $\alpha_t(s)$ is defined to represent the total probability of $\boldsymbol{C}'_{1:s}$ at time t, which can be calculated using the previous α. The initialisation is defined as Eq. 3. $y^t_{\_}$ represents the posterior probability of the _ at time t, and $y^t_{c_i}$ represents the posterior probability of c_i at time t. The variable s is the index of $\boldsymbol{C}'$, and the range of s is $[1, 2\times L + 1]$.

$$\begin{aligned} \alpha_1(1) &= y^1_{\_} \\ \alpha_1(2) &= y^1_{c_1} \\ \alpha_1(s) &= 0, \quad \forall s > 2 \end{aligned} \tag{3}$$

And the recursion process is shown in Eq. 5. When current

[2]Synthesized audio samples are available at the following URL: https://zhaozeqing.github.io/SPTTS/

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) y^t_{1'_s}, & \text{if } \boldsymbol{C}'_s = \_ \text{ or } \boldsymbol{C}'_{s-2} = \boldsymbol{C}'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) y^t_{\boldsymbol{C}'_s}, & \text{otherwise} \end{cases} \tag{4}$$

$$\bar{\alpha}_t(s) \overset{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1) \tag{5}$$

Similarly, the backward variables $\beta_t(s)$,

$$\begin{aligned} \beta_T(|\boldsymbol{C}'|) &= y^T_{\_} \\ \beta_T(|\boldsymbol{C}'|-1) &= y^T_{\boldsymbol{c}_L} \\ \beta_T(s) &= 0, \quad \forall s < |\boldsymbol{C}'| - 1 \end{aligned} \tag{6}$$

$$\beta_t(s) = \begin{cases} \bar{\beta}_t(s) y^t_{\boldsymbol{C}'_s}, & \text{if } \boldsymbol{C}'_s = \_ \text{ or } \boldsymbol{C}'_{s+2} = \boldsymbol{C}'_s \\ (\bar{\beta}_t(s) + \beta_{t+1}(s+2)) y^t_{\boldsymbol{C}'_s}, & \text{otherwise} \end{cases} \tag{7}$$

$$\bar{\beta}_t(s) \overset{\text{def}}{=} \beta_{t+1}(s) + \beta_{t+1}(s+1) \tag{8}$$

$$p(\boldsymbol{C} \mid \boldsymbol{M}) = \sum_{s=1}^{|\boldsymbol{C}'|} \frac{\alpha_t(s)\beta_t(s)}{y^t_{\boldsymbol{C}'_s}} \tag{9}$$

With the forward-backward algorithm, redundant calculations are removed. After the forward variables and backward variables are calculated,it's easier to compute loss values and gradients. Using the Viterbi algorithm, we are able to find the maximum likelihood path as the alignment. If the alignment is like $[\_, c_1, c_1, \_, \_, c_2, \_, c_3, c_3, c_3, \_]$, we choose the location of the next token appear firstly as the end time of the token. So the extracted duration sequence is $[5, 2, 4]$.

The alignment generator is implemented by stacking the Pre-Net (two sequential 1D convolution), multiple FFT blocks, and a classifier (one linear layer and softmax), as shown in Fig. 2(b).

D. Training and Inference

For training, our model consists of three loss terms.

The first term is mel loss $\mathcal{L}_{mel}$, which can be computed as Eq. 10.

$$\mathcal{L}_{mel} = \textbf{MAE}\left(\boldsymbol{M}, \boldsymbol{M}^{'}\right) \tag{10}$$

The second term is the duration model loss $\mathcal{L}_d$, which is the logarithmic domain MSE loss as Eq. 11. Same with [9], the outputs of the duration predictor $\boldsymbol{D}^{'}$ is the phone duration sequence in the logarithmic domain. Same with [15], we stop gradient of the duration loss.

$$\mathcal{L}_d = \textbf{MSE}\left(log\left(\boldsymbol{D}+1\right), \boldsymbol{D}^{'}\right) \tag{11}$$

The last term is the CTC loss $\mathcal{L}_{CTC}$ used for learning the alignment generator as Eq. 2.

The total loss $\mathcal{L}$ can written as Eq. 12. And the length regulator used the duration sequence generated by alignment generator.

$$\mathcal{L} = \mathcal{L}_{mel} + \mathcal{L}_d + \mathcal{L}_{CTC} \tag{12}$$

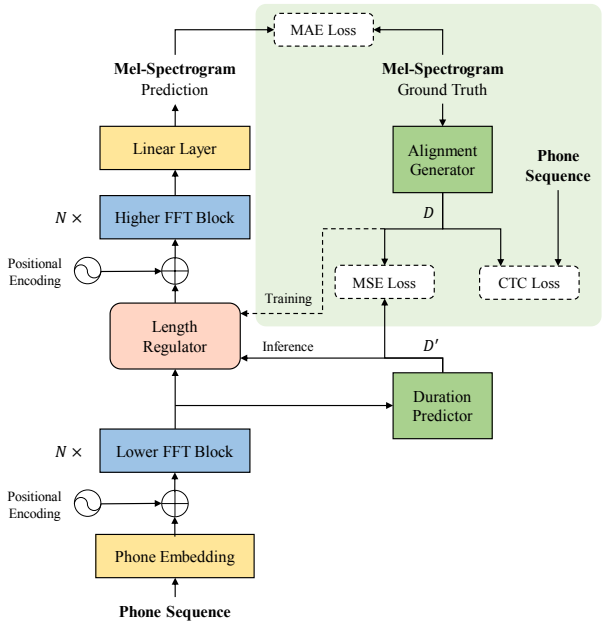

Fig. 1. The overview architecture of SPTTS. The left part is attention-based synthesizer, the right part is alignment generator and duration predictor. The dotted line means $\boldsymbol{D}$ is only used for training process.

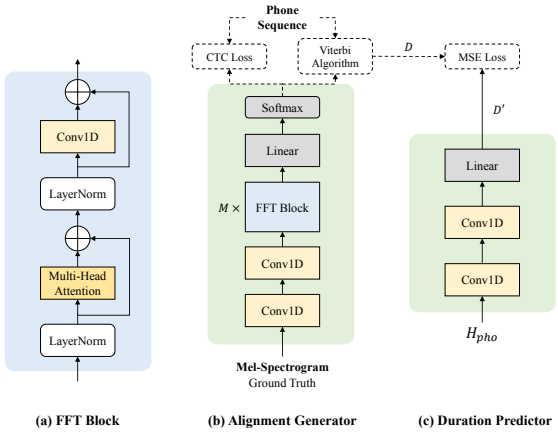

Fig. 2. (a) The Feed-Forward Transformer block. (b) The alignment generator. (c) The duration predictor. The dotted line means it is only used during training.

In the training stage, the duration sequence $\boldsymbol{D}$ is extraced by the alignment generator. Simultaneously, the duration model is trained by $\boldsymbol{D}$ as the ground truth. While inference, the duration sequence is predicted by the duration predictor.

III. Experiments

In this section, we conduct a series of comprehensive experiments to validate the effectiveness of SPTTS. Below we first present the experimental configuration in Sec. III-A and Sec. III-B, followed by the evaluation in Sec. III-C. We also

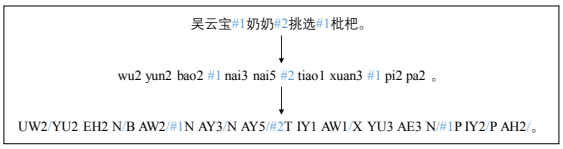

Fig. 3. An illustration of text process. **#1** represents the boundary of prosodic words, **#2** represents the boundary of prosodic phrases, **/** represents the boundary of syllables.

report the alignment analysis in Sec. III-D.

A. Datasets

The Baker dataset is a Chinese standard Mandarin speech synthesis dataset, which includes 10000 sentences, about 12 hours. All sentences in the dataset are spoken by a single female speaker. We sample the raw audio to 24 kHz, and extracted the 80 dimension mel-spectrogram feature using $f_{min} = 80, f_{max} = 7600$. The hop size is 300, the windows size is 1200 and the FFT points is 2048. We split the dataset into train set and valid set, which contain 9500 and 500 audios respectively. We use the International Phonetic Alphabet (IPA) standard to convert pinyins to phones, and the prosody label in dataset is also used for modeling. An example of text process is illustrated in Fig. 3.

B. Model Configuration

The configuration of each component of SPTTS is in Tab. I. **MHA** means Multi-Head Attention.

TABLE I
MODEL CONFIGURATION DETAILS

Components		Structure
Phone Embedding		Linear(131,384)
Lower FFT Block		$\begin{bmatrix}\text{LayerNorm}()\\ \text{MHA}(head=2, units=384)\\ \text{LayerNorm}()\\ \text{Conv1D}(384,1024,kernel=3)\\ \text{Conv1D}(1024,384,kernel=3)\end{bmatrix} \times 4$
Higher FFT Block		$\begin{bmatrix}\text{LayerNorm}()\\ \text{MHA}(head=2, units=384)\\ \text{LayerNorm}()\\ \text{Conv1D}(384,1024,kernel=3)\\ \text{Conv1D}(1024,384,kernel=3)\end{bmatrix} \times 4$
Alignment Generator	Pre-Net	$\begin{bmatrix}\text{Conv1D}(80,320,kernel=3)\\ \text{Conv1D}(320,320,kernel=3)\end{bmatrix}$
	FFT Block	$\begin{bmatrix}\text{MHA}(head=4, units=320)\\ \text{LayerNorm}()\\ \text{Linear}(320,320)\\ \text{Linear}(320,320)\\ \text{LayerNorm}()\end{bmatrix} \times 6$
	Classifier	Linear(320, 132)
Duration Predictor		$\begin{bmatrix}\text{Conv1D}(384,384,kernel=3)\\ \text{Conv1D}(384,384,kernel=3)\\ \text{Linear}(384,1)\end{bmatrix}$

We train the SPTTS model on 1 NVIDIA V100 GPU, with a batch size of 32. We use the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 10^{-9}$ and the same learning rate schedule as [4]. We train the alignment generator with 8k warm-up steps, the attention-based synthesizer with 12k warm-up steps, and duration predictor with 12k warm-up steps. Specially, the model is trained with 120k steps in total (about 32 hours).

C. Evaluation

We present the audio quality comparison of different methods. Specifically, we use the mean opinion score (MOS) on the valid set as the evaluation metric. We randomly choose 50 samples and invite 20 native speakers for subjective evaluation.

1) Audio Quality: We compare the MOS of generated audios by SPTTS with other methods and report the result in Tab. II. Our model outperforms the Fastspeech and AlignTTS, and achieves competitive performance compared with Tacotron2. Moreover, we also train GlowTTS [15] with its open-source code, but couldn't get the idea audio quality, thus we only report its results on the demo webpage rather than adding it in subjective evaluations.

In addition, we also compute the real time factor (RTF) on 80 Intel(R) Xeon(R) Gold 6230 CPU of different models. For the RTF metric, our SPTTS can achieve 26x speed up compared with the Tacotron2 (with the highest audio quality), and on the par with Fastspeech and AlignTTS.

TABLE II
THE PERFORMANCE COMPARISON OF DIFFERENT TTS MODELS. GT MEANS GROUND TRUTH. MB MEANS MULTIBAND MELGAN [20] VOCODER.

Method	MOS	RTF
GT	4.69	-
GT (Mel+MB)	4.52	0.033
Tacotron2 (Mel+MB)	4.46	1.271
Fastspeech (Mel+MB)	4.16	0.046
AlignTTS (Mel+MB)	4.25	0.055
SPTTS (Mel+MB)	4.39	0.049

2) Training Strategy: We also conduct experiments based on our proposed SPTTS to investigate the influence of different training strategies. Specifically, our method does not require an extra aligner, thus we can train the whole system in parallel. We denote this kind of training strategy as *SPTTS*. Moreover, we also follow AlignTTS and adopt a multi-stage training pipeline to learn each component sequentially, which is denoted as *SPTTS-separate*.

The performance of two strategies is in Tab. III. From the table, we find that the parallel training strategy achieves superior performance than the multi-stage training strategy. In our model, the alignment generator, duration predictor and the synthesizer are optimized jointly, which enable the model to achieves better speech quality.

TABLE III
THE MOS OF SPTTS AND SPTTS-SEPARATE

Method	MOS
SPTTS (Mel+MB)	4.39
SPTTS-separate (Mel+MB)	4.27

D. Alignment Analysis

1) Alignment Accuracy: To illustrate the accuracy of alignment quantitatively, we choose one sentence from the valid

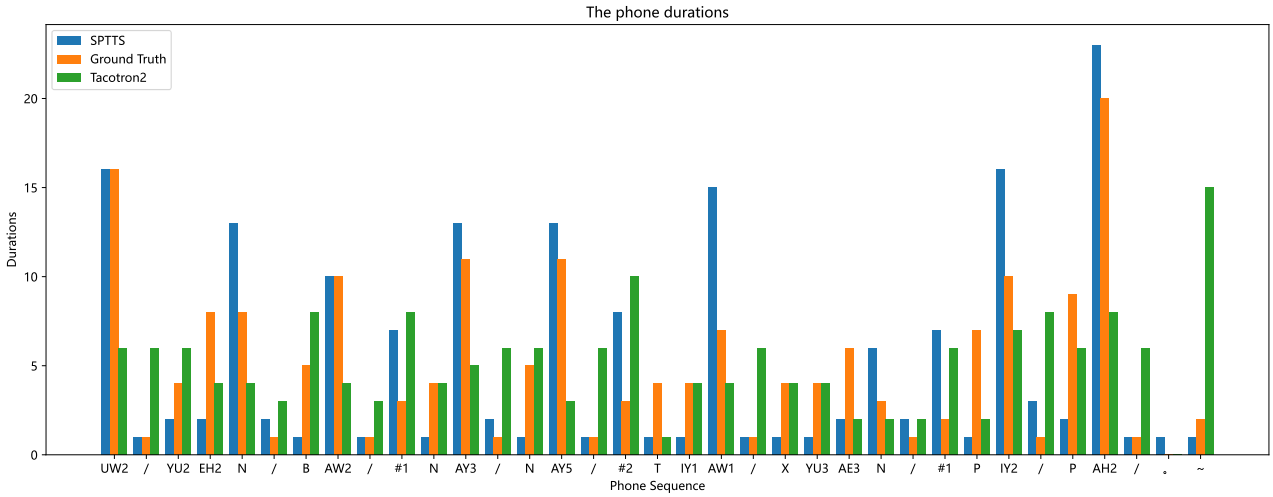

Fig. 4. The phone duration sequence extracted with SPTTS, Tacotron2 and the Ground Truth.

set randomly, and plot its alignments predicted by SPTTS and Tactron2. Because there is no duration label for phone sequence in the dataset, we manually marked one as the ground truth. The phone duration comparison is illustrated in Fig. 4. The figure shows that our proposed alignment generator tends to allocate more duration to vowels and less to syllable boundaries, which may help to learn a better speech synthesis model.

2) Robustness of Speed Control: We also evaluate the audio quality when speed is controlled in inference. We first multiply the duration sequence by 0.8 or 0.9 to speed up the speech, and then multiply the duration sequence by 1.1 or 1.2 to slow down the speech. We find that our SPTTS shows more robustness of speed control. The supervision provided by the alignment generator for the duration predictor is changed dynamically during the training process, which improves the generalization ability of the duration predictor and robustness of speech quality when controlling speed. The synthesized samples are all available on the demo webpage.

IV. Conclusions

In this work, we introduce a simplified parallel non-autoregressive TTS (SPTTS) method. We develop a novel alignment generator, implemented by a recognition-based module to extract the alignment on the fly, and can be optimized with other components in parallel. To demonstrate our method, we evaluate our model on a single speaker Mandarin TTS task. The performance result is competitive compared with previous works, while our method has a simplified structure and training strategy. In addition, we also conduct the alignment analysis, it shows that the duration sequence generated by SPTTS has high accuracy and the synthesized speech has good robustness of speed control.

References

[1] 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," *arXiv:1703.10135*, 2017.

[2] 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.

[3] 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, no. 01, 2019, pp. 6706–6713.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N.Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv:1706.03762*, 2017.

[5] Y. Yasuda, X. Wang, and J. Yamagishi, "End-to-end text-tospeech using latent duration based on vq-vae," *arXiv:2010.09602*, 2020.

[6] Y. Lee, J. Shin, and K. Jung, "Bidirectional variational inference for non-autoregressive text-to-speech," in *International Conference on Learning Representations*, 2020.

[7] K. Peng, W. Ping, Z. Song, and K. Zhao, "Non-autoregressive neural text-to-speech," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7586–7598.

[8] 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.

[9] Y. Ren, C. Hu, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech2: Fast and high-quality end-to-end text-to-speech," *arXiv:2006.04558*, 2020.

[10] C. Yu, H. Lu, N. Hu, M. Yu, C. Weng, K. Xu, P. Liu, D. Tuo, S. Kang, G. Lei et al., "Durian: Duration informed attention network for multimodal synthesis," *arXiv:1909.01700*, 2019.

[11] A. Ła´ncucki, "Fastpitch: Parallel text-to-speech with pitch prediction," *arXiv:2006.06873*, 2020.

[12] I. Elias, H. Zen, J. Shen, Y. Zhang, Y. Jia, R. Weiss, and Y. Wu, "Parallel tacotron: Non-autoregressive and controllable tts," *arXiv:2010.11439*, 2020.

[13] D. Lim, W. Jang, H. Park, B. Kim, J. Yoon et al., "JDI-T: Jointly trained duration informed transformer for text-to-speech without explicit alignment," *arXiv:2005.07799*, 2020.

[14] Z. Zeng, J. Wang, N. Cheng, T. Xia, and J. Xiao, "AlignTTS: Efficient feed-forward text-to-speech system without explicit alignment," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6714–6718.

[15] J. Kim, S. Kim, J. Kong, and S. Yoon, "Glow-TTS: A generative flow for text-to-speech via monotonic alignment search," *arXiv:2005.11129*, 2020.

[16] 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 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7209–7213.

[17] A. Graves, S. Fernandez, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," *Proc. ICML 2006*, vol. 148, 2006.

[18] Z. Tian, J. Yi, J. Tao, Y. Bai, S. Zhang, and Z. Wen, "Spiketriggered non-autoregressive transformer for end-to-end speech recognition," *Proc. Interspeech 2020*, pp. 5026–5030, 2020.

[19] Moritz, N. . Hori, and T. Roux, "Triggered attention for end-to-end speech recognition," in *ICASSP IEEE International Conference on Acoustics*, 2019.

[20] G. Yang, S. Yang, K. Liu, P. Fang, W. Chen, and L. Xie, "Multi-band MelGAN: Faster waveform generation for high-quality text-to-speech," in *arXiv:2005.05106*, 2020.