*INTERSPEECH 2021*
30 August – 3 September, 2021, Brno, Czechia

# EfficientSing: A Chinese Singing Voice Synthesis System Using Duration-Free Acoustic Model and HiFi-GAN Vocoder

*Zhengchen Liu, Chenfeng Miao, Qingying Zhu, Minchuan Chen, Jun Ma, Shaojun Wang, Jing Xiao*

Ping An Technology, Shanghai, P.R.China
{LIUZHENGCHEN871, MIAOCHENFENG448, ZHUQINGYING568, CHENMINCHUAN109}@pingan.com.cn

## Abstract

In this paper, we present EfficientSing, a Chinese singing voice synthesis (SVS) system based on a non-autoregressive duration-free acoustic model and HiFi-GAN neural vocoder. Different from many existing SVS methods, no auxiliary duration prediction module is needed in this work, since a newly proposed monotonic alignment modeling mechanism is adopted. Moreover, we follow the non-autoregressive architecture of EfficientTTS with some singing-specific adaption, making training and inference fully parallel and efficient. HiFi-GAN vocoder is adopted to improve the voice quality of synthesized songs and inference efficiency. Both objective and subjective experimental results show that the proposed system can produce quite natural and high-fidelity songs and outperform the Tacotron-based baseline in terms of pronunciation, pitch and rhythm.

**Index Terms**: EfficientSing, singing voice synthesis, non-autoregressive, monotonic alignment, HiFi-GAN

## 1. Introduction

Singing voice synthesis (SVS) aims at generating natural songs from text together with some musical score information (e.g. note and tempo). These years SVS technique has drawn more and more attention, and it can be applied to many aspects including virtual avatars, computer games and short video apps. In addition, SVS can be linked to other artificial intelligence tasks such as automatic lyrics generation and automatic composition.

Similar to text-to-speech (TTS) synthesis systems, methods based on unit selection [1, 2, 3] and statistical parametric synthesis [4, 5, 6], have been dominant in SVS for many years. Among them, hidden Markov model (HMM)-based methods are especially popular, since they can model several acoustic features simultaneously and make it easy to control duration and pitch. However, due to the limited modeling ability of statistical parametric models and the use of traditional vocoders, the quality, naturalness and expressiveness of the generated songs have a large gap to those of the natural recordings.

In recent years, many deep learning based models have been introduced into SVS. Feed-forward neural networks (FFNN) and convolutional neural networks (CNN) are employed to improve the acoustic model [7, 8, 9]. Recurrent neural networks (RNN) with long short-term memory (LSTM) cells are adopted to model the long temporal dependencies [10]. Moreover, deep autoregressive neural network based methods are proposed for SVS [11]. As one of the most popular generative models, generative adversarial networks (GAN) are also applied to SVS [12, 13].

An important development in TTS is the introduction of sequence-to-sequence (S2S) models with attention mechanism, such as Tacotron [14, 15], Deep Voice [16] and FastSpeech [17]. In these models, an attention-based encoder-decoder architecture receives linguistic features and predicts acoustic features, which are converted to waveform by a followed neural vocoder. Some relevant modules are introduced into SVS. In [18], an attention-based S2S model is proposed to take as input a sequence of phoneme-level note embeddings from musical scores and predict mel-spectrograms, followed by a WaveNet [19] vocoder to increase the quality of the generated samples. This work dispenses with the need of a separate model for allocating phoneme durations and demonstrates that the attention-based S2S architecture is capable of autonomously modeling singing-specific features. The main drawback, however, is that duration prediction is not precise enough, losing the rhythm over time. In [20], an SVS system consisting of a Tacotron-like acoustic model and a WaveRNN [21] vocoder is proposed, which can produce quite natural, expressive and highly-fidelity songs. However, the process of handling duration is complex: an auxiliary duration model and a post-processing step are needed. Besides, its inference efficiency is restricted due to the autoregressive manner. In [22], a high-quality and fast-speed SVS system employing an integrated network for spectrum, F0 and duration modeling is proposed. Still, it needs a separate duration predictor to generate phoneme durations and syllable durations. Furthermore, it requires quite a large dataset and extracting traditional vocoder features is costly.

The proposed SVS system *EfficientSing* is inspired by our prior work *EfficientTTS* [23], which achieves fast-speed and high-quality performance in TTS. To sute SVS task, some necessary modifications are made, mainly on the input features. The contribution of this paper can be summarized as follows. First, a newly proposed approach to produce monotonic alignments between the phoneme and musical score sequences and their corresponding acoustic features is introduced into SVS task. Thus, no additional duration models are needed. Second, the adoption of the non-autoregressive architecture and HiFi-GAN [24] neural vocoder makes the training and inference procedure fully parallel and fast-speed. Third, the size of the required dataset is comparable to those in the field [18, 20] and no data augmentation is needed.

## 2. The Proposed System

The proposed EfficientSing system integrates acoustic feature and duration modeling based on a variant of EfficientTTS and produces waveform through a HiFi-GAN vocoder, as illustrated in Figure 1. The self-designed input features contain linguistic, note and duration information which is inferred from musical scores. Following EfficietTTS, the acoustic model consists of several modules: a text-encoder, a mel-encoder, an index mapping vector (IMV) generator, an aligned position predictor and a decoder. At the training stage, the text-encoder and the mel-encoder convert text symbols and mel-spectrograms to hidden



http://dx.doi.org/10.21437/Interspeech.2021-771

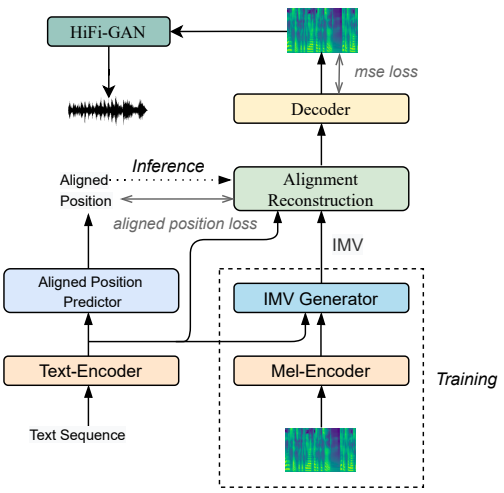



Figure 1: *Diagram of EfficientSing system. The dashed block only exists at the training stage and the dotted line indicates a flow only for inference. Two loss functions are also highlighted.*

representations, respectively. These representations are used to calculate the IMV through the IMV generator. The IMV is converted to a 2-dimentional alignment matrix, which is further used to generate a time-aligned representation by an alignment reconstruction layer. Finally, the time-aligned representation is received by the decoder to predict mel-spectrograms. Meanwhile, an HiFi-GAN vocoder is trained using the recorded songs and the corresponding ground-truth mel-spectrograms. At the inference stage, the mel-encoder is omitted and the input feature sequence is inferred from musical scores. The whole song is segmented into several short sentences for the convenience of modeling. Given the input features, the hidden representations are inferred by the text-encoder, which are fed into the aligned position predictor to predict the alignment matrix. Then the decoder generates the mel-spectrogram sequence in a parallel way. Finally the trained HiFi-GAN vocoder transforms the predicted mel-spectrograms to singing waveforms.

### 2.1. Feature Representation

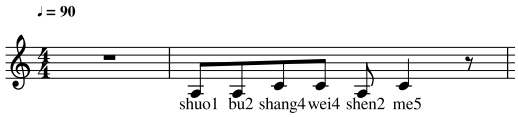
In this work, musical scores are described in MusicXML format [25]. The input feature sequence consists of three streams:

- The phoneme sequence for the song utterance. It is inferred from syllables in the lyrics and one-hot encoded. Tones of each syllable and punctuations are ignored. Two tokens which represent the silence at the start and end of an utterance ('∼') and Chinese character boundary ('<s>') are added.
- The note sequence to be sung on each phoneme. It is one-hot encoded. Accidentals, naturals and ties are carefully handled.
- The duration sequence of the note to be sung on each phoneme. It is firstly calculated as a floating point number according to tempo and note, denoted as $dur_f$. Then it is quantized using the equation below:

$$dur_q = \lfloor \frac{\ln dur_f - \ln dur_{f\_min}}{\ln dur_{f\_max} - \ln dur_{f\_min}} * 256 \rfloor, \quad (1)$$

where $dur_{f\_max}$ and $dur_{f\_min}$ are maximum and minimum value of $dur_f$ in the dataset, respectively.

The three streams are aligned on the note level and concatenated. More details are shown in Figure 2.




| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| phoneme | ~ | sh | uo | <s> | b | u | <s> | sh | ang | <s> |
| phoneme ID | 1 | 43 | 53 | 2 | 9 | 45 | 2 | 43 | 7 | 2 |
| note | - | A3 | A3 | A3 | A3 | A3 | A3 | C4 | C4 | C4 |
| note ID | 316 | 229 | 229 | 229 | 229 | 229 | 229 | 5 | 5 | 5 |
| duration | 2.67 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 |
| quantized duration | 193 | 80 | 80 | 80 | 80 | 80 | 80 | 80 | 80 | 80 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| w | ei | <s> | sh | en | <s> | m | e | ~ |
| 56 | 14 | 2 | 43 | 15 | 2 | 34 | 13 | 1 |
| C4 | C4 | C4 | A3 | A3 | A3 | C4 | C4 | - |
| 5 | 5 | 5 | 229 | 229 | 229 | 5 | 5 | 316 |
| 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.33 | 0.67 | 0.67 | 2.67 |
| 80 | 80 | 80 | 80 | 80 | 80 | 119 | 119 | 193 |

Figure 2: *An example of musical score representation.*

### 2.2. Acoustic Model

The implementation of the text-encoder follows that in FastSpeech, which consists of a text embedding layer and a stack of feed-forward transformer (FFT) blocks [26]. The text embedding layer receives categorical phoneme sequences. Each FFT block consists of a self-attention and a 1D convolutional network. Multi-head attention is used to capture the cross-position information. Residual connections, layer normalization [27] and dropout are also utilized. The mel-encoder receives the ground-truth mel-spectrograms and encodes them into high-dimentional representations. It consists of a linear layer and several convolutional layers, interspersed with residual connections, weight normalization and Leaky ReLU activation. Note that the mel-encoder only exists at the training stage, since the mel-spectrograms are not available at the inference stage.

Denote $\boldsymbol{\alpha} \in \mathcal{R}^{(T_1,T_2)}$ as the alignment matrix between the input sequence $\boldsymbol{x} \in \mathcal{R}^{T_1}$ and the output sequence $\boldsymbol{y} \in \mathcal{R}^{T_2}$. According to [23], the *index mapping vector* (IMV) $\boldsymbol{\pi}$ is defined as the weighted sum of index vector $\boldsymbol{p} = [0, 1, \ldots, T_1 - 1]$:

$$\pi_j = \sum_{i=0}^{T_1-1} \alpha_{i,j} * p_i, \quad (2)$$

where $0 \leq j \leq T_2 - 1$, $\boldsymbol{\pi} \in \mathcal{R}^{T_2}$ and $\sum_{i=0}^{T_1-1} \alpha_{i,j} = 1$. The IMV can be understood as the expectation of the location when an output timestep is "mapped" to the input timeline. At the training stage, the IMV generator produces the IMV in two steps. First, the alignment $\boldsymbol{\alpha}$ is calculated through a scaled dot-product attention:

$$\alpha_{i,j} = \frac{\exp(-D^{-0.5}\boldsymbol{q}_j \cdot \boldsymbol{k}_i)}{\sum_{m=0}^{T_1-1} \exp(-D^{-0.5}\boldsymbol{q}_j \cdot \boldsymbol{k}_m)}, \quad (3)$$

where $\boldsymbol{k}$ and $\boldsymbol{q}$ are the outputs of the text-encoder and mel-encoder, respectively, and $D$ is their dimentionality. Second, the IMV is calculated from $\boldsymbol{\alpha}$ following the equations (13-15) in [23]. At the inference stage, the IMV needs to be predicted from text only, which is challenging. This issue can be sidestepped by predicting the aligned positions $\boldsymbol{e}$ of each input token instead. $m(\cdot)$ is defined as the mapping function between $\boldsymbol{\pi}$ and $\boldsymbol{q}$: $\boldsymbol{\pi} = m(\boldsymbol{q})$. Since $m(\cdot)$ is a monotonic transformation,the aligned positions $\boldsymbol{e}$ can be computed as:

$$\boldsymbol{e} = m^{-1}(\boldsymbol{p}), \boldsymbol{p} = [0, 1, \ldots, T_1 - 1]. \quad (4)$$

Let $\boldsymbol{q} = [0, 1, \ldots, T_2 - 1]$ and we have

$$e_i = \sum_{n=0}^{T_2-1} \gamma_{i,n} * q_n, \quad (5)$$

where

$$\gamma_{i,j} = \frac{\exp(-\sigma^{-2}(p_i - \pi_j)^2)}{\sum_{n=0}^{T_2-1} \exp(-\sigma^{-2}(p_i - \pi_n)^2)}. \quad (6)$$

In practice, relative position $\Delta \boldsymbol{e}$ (where $\Delta e_i = e_i - e_{i-1}, 1 \leq i \leq T_1 - 1$) instead of $\boldsymbol{e}$ is learned to make parallel operation available. $\Delta \boldsymbol{e}$ is designed to be the training target and the corresponding loss function is defined as:

$$\mathcal{L}_{ap} = \|\log(\Delta \hat{\boldsymbol{e}} + \epsilon) - \log(\Delta \boldsymbol{e} + \epsilon)\|_1, \quad (7)$$

where $\Delta \hat{\boldsymbol{e}}$ and $\Delta \boldsymbol{e}$ is the predicted and ground-truth relative position, respectively, $\epsilon$ is a small number to avoid numerical instabilities, $\|\cdot\|_1$ is the $L1$ norm. The aligned position predictor is composed of convolutional layers with layer normalization and ReLU activation, and it is trained jointly with the rest of the acoustic model.

The decoder receives the time-aligned representations and outputs the predicted mel-spectrograms. It consists of convolutional layers with weight normalization, Leaky ReLU activation and residual connections, followed by a linear projection. The total loss function of the acoustic model is a combination of the aligned position loss and the mean square error (MSE) loss:

$$\mathcal{L} = \mathcal{L}_{ap} + \lambda * \mathcal{L}_{mse}, \quad (8)$$

where $\lambda$ is the weight to be tuned and $\mathcal{L}_{mse}$ is defined as:

$$\mathcal{L}_{mse} = \frac{1}{N} \sum_{i}^{N} (\hat{mel}_i - mel_i)^2, \quad (9)$$

where $\hat{mel}$ and $mel$ are the predicted and ground-truth mel-spectrograms, respectively. More details about the IMV could be found in [23].

### 2.3. HiFi-GAN Neural Vocoder

Neural vocoder techniques have achieved great success in recent years [19, 21, 24, 28, 29, 30]. Traditional vocoders such as WORLD [31] depend on the source-filter hypothesis and some hand-drafted features, which are difficult to extract accurate values from singing voice. In comparison, neural vocoders model audio samples directly and can be conditioned on mel-spectrograms only, which implicitly include all acoustic elements such as pitch and formant. Among these, HiFi-GAN achieves both higher computational efficiency and sample quality than autoregressive or flow-based models. Therefore, HiFi-GAN is adopted in this paper. The HiFi-GAN vocoder is trained on the exactly same data as the acoustic model.

## 3. Experiments

### 3.1. Experimental Setup

The performance of the proposed EfficientSing system was evaluated on an internal singing voice dataset. 230 songs from a female singer were recorded in a professional recording room without accompaniment music. 200 songs were randomly selected to form the training set and the rest 30 songs were for test set. The musical scores were finely reviewed and the recorded songs were segmented into short sentences according to the rest and phrase. All the songs were recorded at 48 kHz sampling rate and downsampled to 24 kHz. The musical scores were annotated in MusicXML format and the lyrics were manually transcribed to pinyin. 80-dimentional mel-spectrograms were extracted with an FFT size 1024 and hop size 384. $\lambda$ was empirically set to 1. Model parameters of the acoustic model and training settings are shown in Table 1 . The HiFi-GAN vocoder was trained using the official release[1] on config_v1 configuration for 600k steps. Two systems were built for comparison: *EfficientSing* (the proposed EfficientSing system) and *Baseline* (the SVS system to be introduced in 3.2).

Table 1: *Acoustic model architecture and training settings.*

| Modules | Parameters |
|---|---|
| Text-Encoder | FFTLayers = 5,<br>AttentionHeads = 2,<br>TextEmbeddingDim = 512,<br>NoteEmbeddingDim = 256,<br>DurEmbeddingDim = 256,<br>Dilation = [1, 1, 2, 2, 2]<br>HiddenDim = 512 |
| Mel-Encoder | Conv1DLayers = 3,<br>KernelSize = 5,<br>Dilation = [1, 1, 2],<br>FilterSize = 512 |
| Aligned Position Predictor | $\sigma^{-2}$ = 0.5,<br>Conv1DLayers = 3,<br>KernelSize = [3, 3, 1],<br>FilterSize = [128, 32, 1] |
| Decoder | Conv1DLayers = 8,<br>KernelSize = 5,<br>Dil = [1, 2, 2, 2, 1, 1, 1, 1],<br>FilterSize = 512 |
| Trainging Settings | BatchSize = 8,<br>LearningRate = 0.001,<br>Optimizer = Adam,<br>TrainingSteps = 7M |

### 3.2. Baseline

The proposed system was compared with a baseline consisting of a Tacotron-based acoustic model and the same HiFi-GAN vocoder, which was inspired by Tacotron2 [15]. A widely-used open-source implementation[2] was adopted and necessary modifications were made to apply to SVS task. Forward attention [32] was used to achieve faster convergence and higher stability. The dimensions of embedding layer for phoneme, note and duration stream were 512, 256 and 256, respectively. The encoder consisted of 3 convolutional layers and a single bidirectional LSTM layer with 256 units. The decoder had 2 unidirectional LSTM layers with 1024 units and each decoder step produced two mel-spectrogram frames. A 5-convolutional-layer postnet with kernel size 5 was adopted to improve the overall output quality. The input features and mel-spectrograms were same with the proposed EfficientSing system.

### 3.3. Objective Evaluation

Objective tests were conducted to evaluate the performance of these systems. Mel-cepstral distortion (MCD) and root mean square error (RMSE) of F0 values between the natural recordings and the synthesized singing voices from different systems were chosen to measure the performance of the acoustic model. Since waveforms from different systems had different

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

[2] https://github.com/Rayhane-mamah/Tacotron-2

Table 2: *Comparison of different systems on distortion between ground truth and predicted features.*

| system | EfficientSing | Baseline |
|---|---|---|
| MCD (dB) | **7.52** | 7.66 |
| F0 RMSE (Hz) | **6.89** | 7.39 |
| duracc (%) | **93.94** | 92.92 |

Table 3: *RTF on text-to-mel inference.*

| system | RTF (mean±standard deviation) |
|---|---|
| EfficientSing | **0.00692**±0.00095 |
| Baseline | 0.06478±0.00136 |

durations, a dynamic time warping (DTW) procedure was conducted. Besides, to evaluate the performance of duration modeling, a *duracc* (duration accuracy) [33] is designed:

$$duracc = 1 - \frac{\sum_i^N |predicted_i - real_i|}{\sum_i^N max(real_i,\ predicted_i)}, \qquad (10)$$

where $N$ is the total number of all the short sentences, $predicted_i$ and $real_i$ is the predicted duration and the ground-truth duration of the $i$th short sentence, respectively. The experimental results are presented in Table 2. We can see that *EfficientSing* achieves smaller spectral distortion and preciser pitch prediction than *Baseline*. Due to the limitations of the pitch range of humans and the genres of the selected songs, the data imbalance problem exists in singing voice corpus more or less. The repetitions in the songs such as refrains or repeated lyrics and pitch patterns make things even worse. For example, C4 appears 4885 times in our training set, while C6 only appears 6 times. The modeling accuracy for C6 may deteriorate since there is no enough training data, which leads to a larger F0 RMSE. We can also see that *EfficientSing* outperforms *Baseline* in terms of duration prediction, which demonstrates the effectiveness of the employed IMV-based alignment modeling mechanism. Notably, some common issues like too long or too short pauses encountered in *Baseline* do not appear in *EfficientSing*.

Evaluation on inference speed was also conducted. 281 sentences from test set were selected and inference from text to mel-spectrograms was run 10 times on a single NVIDIA Tesla V100 GPU to get an average real time factor (RTF: elapsed time / output sequence length). The lengths of the generated mel-spectrogram sequences ranged from 1.66s to 29.44s. Table 3 shows the results. We can see that *EfficientSing* can make it possible to produce mel-spectrograms from text more than 140× faster than real time ($1/0.00692 \approx 140$). This is owed to the duration-free and non-autoregressive architecture of the acoustic model, making the inference procedure straightforward and fully parallel.

### 3.4. Subjective Evaluation

A subjective test was conducted to better evaluate the difference of the synthesized songs between *EfficientSing* and *Baseline*. For each system, 15 audio samples were prepared[3]. Corresponding natural recordings were also provided for reference. 12 listeners were asked to rate the mean opinion score (MOS) in terms of three aspects: pronunciation, pitch and rhythm. As shown in Figure 3, *EfficientSing* outperforms *Baseline* on all the three aspects, and the superiority of *EfficientSing* over *Baseline* is significant according to the $t$-test. Generally, lyrics of the synthesized songs by *EfficientSing* could be easily understood. However, some mumbling or mispronunciation cases were observed, especially for phonemes sustained for a long time. Moreover, listeners commented that the breath parts sounded unnatural. These problems may be attributed to the lack of training data and vocoder glitches. The MOSs of *EfficientSing* in terms of pitch and rhythm were not so good as that of pronunciation. The main reason may be that the singer did not follow the musical scores exactly, so inaccurate pitch and tempo were introduced into the data. In addition, singing skills including vibrato, falsetto and overshoot were intractable. One possible way to improve the performance is to adopt a lager, multi-singer and more balanced corpus. Figure 4 presents a same part of F0 contour extracted from the synthesized songs by different systems. *EfficientSing* seems much closer to the ground truth.

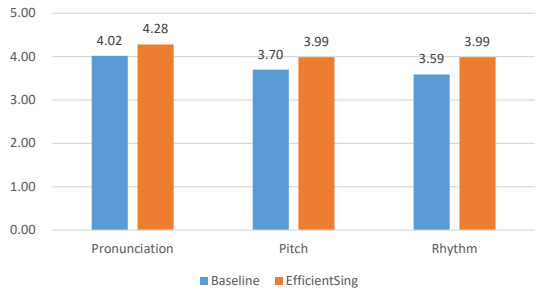



Figure 3: *MOS test results for both Baseline and EfficientSing systems on pronunciation, pitch and rhythm.*

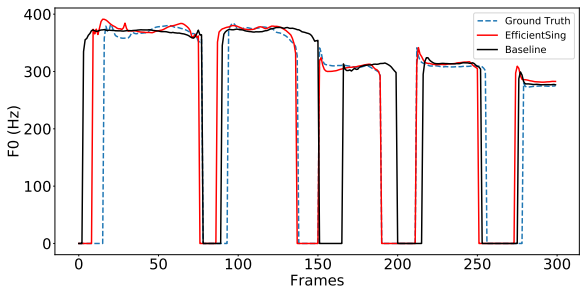



Figure 4: *Comparison of F0 contour for different systems.*

## 4. Conclusions

This paper presents EfficientSing, a Chinese singing voice synthesis system which adopts a non-autoregressive duration-free acoustic model and neural vocoder. With a novel alignment modeling mechanism and a fully parallel architecture, EfficientSing can be free of duration model and generate mel-spectrograms efficiently. HiFi-GAN vocoder is employed to improve the quality of synthesized waveforms. Objective and subjective experimental results demonstrate that the proposed system achieves better performance in terms of pronunciation, pitch and rhythm over the baseline. In the future, more effective methods for feature (text, note, duration, etc.) fusion and singing skills modeling will be studied. Techniques for data augmentation and automatic data labeling will also be explored.

[3]Examples of synthesized samples from different systems are available at `https://bangbuken.github.io/EfficientSing/index.html`

# 5. References


[1] M. Macon, L. Jensen-Link, E. B. George, J. Oliverio, and M. Clements, "Concatenation-based midi-to-singing voice synthesis," in *Audio Engineering Society Convention 103*. Audio Engineering Society, 1997.

[2] H. Kenmochi and H. Ohshita, "Vocaloid-commercial singing synthesizer based on sample concatenation," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[3] J. Bonada, M. Umbert Morist, and M. Blaauw, "Expressive singing synthesis based on unit selection for the singing synthesis challenge 2016," *Morgan N, editor. Interspeech 2016; 2016 Sep 8-12; San Francisco, CA.[place unknown]: ISCA; 2016. p. 1230-4.*, 2016.

[4] K. Saino, H. Zen, Y. Nankaku, A. Lee, and K. Tokuda, "An HMM-based singing voice synthesis system," in *Ninth International Conference on Spoken Language Processing*, 2006.

[5] K. Oura, A. Mase, T. Yamada, S. Muto, Y. Nankaku, and K. Tokuda, "Recent development of the HMM-based singing voice synthesis system—sinsy," in *Seventh ISCA Workshop on Speech Synthesis*, 2010.

[6] K. Nakamura, K. Oura, Y. Nankaku, and K. Tokuda, "HMM-based singing voice synthesis and its application to Japanese and English," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 265–269.

[7] M. Nishimura, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Singing voice synthesis based on deep neural networks." in *Interspeech*, 2016, pp. 2478–2482.

[8] Y. Hono, S. Murata, K. Nakamura, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Recent development of the DNN-based singing voice synthesis system—sinsy," in *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2018, pp. 1003–1009.

[9] K. Nakamura, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Singing voice synthesis based on convolutional neural networks," *arXiv preprint arXiv:1904.06868*, 2019.

[10] J. Kim, H. Choi, J. Park, S. Kim, J. Kim, and M. Hahn, "Korean singing voice synthesis system based on an LSTM recurrent neural network," in *Proc. Interspeech*, 2018, pp. 1551–1555.

[11] Y.-H. Yi, Y. Ai, Z.-H. Ling, and L.-R. Dai, "Singing voice synthesis using deep autoregressive neural networks for acoustic modeling," *Proc. Interspeech 2019*, pp. 2593–2597, 2019.

[12] Y. Hono, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, "Singing voice synthesis based on generative adversarial networks," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6955–6959.

[13] P. Chandna, M. Blaauw, J. Bonada, and E. Gómez, "WGANsing: A multi-voice singing voice synthesizer based on the WassersteinGAN," in *2019 27th European Signal Processing Conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[14] 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.

[15] 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.

[16] 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," in *International Conference on Learning Representations*, 2018.

[17] 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.

[18] O. Angelini, A. Moinet, K. Yanagisawa, and T. Drugman, "Singing synthesis: With a little help from my attention," *Proc. Interspeech 2020*, pp. 1221–1225, 2020.

[19] 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*, pp. 125–125.

[20] Y. Gu, X. Yin, Y. Rao, Y. Wan, B. Tang, Y. Zhang, J. Chen, Y. Wang, and Z. Ma, "Bytesing: A Chinese singing voice synthesis system using duration allocated encoder-decoder acoustic models and WaveRNN vocoders," in *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2021, pp. 1–5.

[21] 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 *International Conference on Machine Learning*. PMLR, 2018, pp. 2410–2419.

[22] P. Lu, J. Wu, J. Luan, X. Tan, and L. Zhou, "Xiaoicesing: A high-quality and integrated singing voice synthesis system," *Proc. Interspeech 2020*, pp. 1306–1310, 2020.

[23] C. Miao, S. Liang, Z. Liu, M. Chen, J. Ma, S. Wang, and J. Xiao, "Efficienttts: An efficient and high-quality text-to-speech architecture," *arXiv preprint arXiv:2012.03500*, 2020.

[24] 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, 2020.

[25] M. Good, "Musicxml in commercial applications," *Music Analysis East and West*, pp. 9–20, 2006.

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010.

[27] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[28] 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.

[29] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," *arXiv preprint arXiv:1910.06711*, 2019.

[30] R. Yamamoto, E. Song, and J.-M. Kim, "Parallel WaveGAN: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6199–6203.

[31] M. Morise, F. Yokomori, and K. Ozawa, "World: a vocoder-based high-quality speech synthesis system for real-time applications," *IEICE TRANSACTIONS on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[32] 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.

[33] H. Xue, S. Yang, Y. Lei, L. Xie, and X. Li, "Learn2sing: Target speaker singing voice synthesis by learning from a singing teacher," in *2021 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2021, pp. 522–529.