A UNIFIED MODEL FOR ZERO-SHOT SINGING VOICE CONVERSION AND SYNTHESIS

Jui-Te Wu[1] **Jun-You Wang**[2] **Jyh-Shing Roger Jang**[1,2] **Li Su**[1,3]
[1]NTU-AS Data Science Degree Program, National Taiwan University, Taiwan
[2]Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
[3]Institute of Information Science, Academia Sinica, Taiwan
Project page: `https://github.com/bryan051003/USVG`

ABSTRACT

Recent advances in deep learning not only facilitate the implementation of zero-shot singing voice synthesis (SVS) and singing voice conversion (SVC) tasks but also provide the opportunity to unify these two tasks into one generalized model. In this paper, we propose such a model that generate the singing voice of any target singer from any source singing content in either text or audio format. The model incorporates self-supervised joint training of the phonetic encoder and the acoustic encoder, with an audio-to-phoneme alignment process in each training step, such that these encoders map the audio and text data respectively into a shared, temporally aligned, and singer-agnostic latent space. The target singer's latent representations encoded at different granularity levels are all trained to match the source latent representations sequentially with the attention mechanisms in the decoding stage. This enables the model to generate unseen target singer's voice with fine-grained resolution from either text or audio sources. Both objective and subjective experiments confirmed that the proposed model is competitive with the state-of-the-art SVC and SVS methods.

1. INTRODUCTION

Singing voice generation is an innovative technology in contemporary music and multimedia content production. There are two major approaches to singing voice generation, which are 1) singing voice synthesis (SVS) [1–7] and 2) singing voice conversion (SVC) [8–12]. SVS features generation from a given musical score and lyrics (texts) and SVC from the given content (e.g., pitch, rhythm) of a source audio recording. Both approaches aim at imitating the performance style (e.g., expression, timbre) of a target singer's recordings in the generation result. In recent years, the SVS and SVC tasks have both witnessed a great improvement thanks to the advances in deep learning. Training a model that generates singing voice in different singer identities is no longer an issue. However, most of these works only support the generation of preset singers' voice, and learning the voices of new singers usually requires a sufficient amount of data to fine-tune the model [4, 7].

A practical singing voice generation system requires the capability to adapt to arbitrary singers, also known as the *zero-shot* scenario. Recent works on zero-shot SVC/SVS have utilized speaker encoders which pretrained on speaker verification tasks [13, 14] to generate speaker embeddings that enable the system to adapt to unseen singers' voices [15–17]. However, this approach has proven insufficient for unseen voice adaptation in recent speech generation studies. An advanced zero-shot voice conversion (VC) and text-to-speech (TTS) approach have utilized the attention mechanism to extract target speaker information from reference audio according to the desired content [18, 19], which outperforms the speaker encoder method [20, 21]. Such advances show potential in zero-shot singing voice generation, which is more challenging than VC and TTS due to the high variation of pitch, timbre, and expression in the singing voice, as well as the lack of publicly available datasets containing a large number of singers. These issues hinder a zero-shot singing voice generation model from sufficient generalization.

Recently, a semi-supervised SVS model combining an acoustic encoder and a phonetic encoder is proposed to deal with both audio and text inputs [7]. Jointly training the two encoders allows the SVS model to learn new singing voice identities in the fine-tuning stage with audios without text annotation. This approach provides new perspectives of zero-shot singing voice generation: from the aspect of technical concerns, joint training of both audio and phonetic encoders might improve the generalizability of zero-shot SVS/SVC; from the aspects of the application, SVS and SVC are no longer regarded as independent but unified into one task.

While [7] only supports SVS for preset singers' voice, in this paper we propose a unified model for SVC and SVS, which is able to generate arbitrary singing voice by either text or audio input. In addition, the system does not require well-aligned lyrics for synthesis since the phoneme duration is learned in a self-supervised manner. We utilize the attention mechanism to extract the target singer information, and introduce two variations of implementation, depending on the similarity or naturalness of the generated

audio. Results show that for both SVC and SVS on unseen singers, the proposed model outperforms state-of-the-art.

2. METHOD

Figure 1 illustrates the whole diagram of the proposed system. The left side of the diagram (blue part) is the source encoder, which encodes the content information of the desired output from either audio or text input. It contains an acoustic encoder (E_a) which encodes the spectrogram and a phonetic encoder (E_p) which encodes the phoneme sequence. The monotonic alignment search (MAS) module [22] encourages the output latent spaces of the two encoders to be in a similar distribution such that the phoneme embeddings are temporally aligned with the spectrogram. SVC or SVS can be achieved by choosing which encoder output to process. On the right side of the diagram, the target encoder (E_t) encodes the spectrograms of the target singer's audios into singer information, where the decoder (D) fuses it into the content information and generates the final output. The functions of all these building blocks will be introduced hereafter in this section.

2.1 Source encoder

Inspired by [7], our source encoder supports joint training for both audio and text inputs. The audio input $\mathbf{x}_s \in \mathbb{R}^{t_a \times n}$ is the log-scale mel-spectrogram of an input audio segment from the source singer s, where t_a is the number of frames and $n = 80$ is the number of mel-bands. The text input $\mathbf{p}_s$, which is corresponding to $\mathbf{x}_s$ is a phoneme index sequence with the length of t_p.

The acoustic encoder E_a maps $\mathbf{x}_s$ to a latent representation $\mathbf{z}_a \in \mathbb{R}^{t_a \times d}$ of source audio, where d is the dimension of this latent representation. Two methods are adopted to train the singer-agnostic $\mathbf{z}_a$. First, we employ a singer classifier C, which works as a domain confusion network [8] that attempts to predict the one-hot singer embedding s from $\mathbf{z}_a$ in a frame-by-frame manner by minimizing the loss function $\mathcal{L}_C := \text{CE}(C(\mathbf{z}_a), s)$, in which CE is the categorical cross-entropy. On the other hand, E_a is trained by maximizing $\mathcal{L}_C$ by using a gradient reversal layer. Second, after having $\mathbf{z}_a$ from E_a, we add Gaussian noise $\mathcal{N}(0, \alpha)$ to $\mathbf{z}_a$ to prevent E_a from overfitting the mel-spectrogram details and to encourage the stability of phoneme embedding [7]. We set $\alpha = 0.1$ in this paper.

The phonetic encoder E_p firstly maps $\mathbf{p}_s$ to a phonetic latent space $\mathbf{z}_p \in \mathbb{R}^{t_p \times d}$ at one of its intermediate layer. Then, the MAS module [22] is adopted to find the alignment between the acoustic embedding and the phonetic embedding and obtains the duration of each phoneme, which is required in the synthesis stage. Denote each frame of $\mathbf{z}_p$ as $\mathbf{z}_{p,i}$ for $i = 1, \cdots t_p$. Each $\mathbf{z}_{p,i}$ is fed into a linear layer which outputs a phonetic prior distribution $\hat{\mathbf{z}}_{p,i} \sim \mathcal{N}(\mu_i, \sigma_i)$. In the MAS process, the phoneme prior distribution sequence $\hat{\mathbf{z}}_p := \{\hat{\mathbf{z}}_{p,i}\}_{i=1}^{t_p}$ is aligned to $\mathbf{z}_a$ by finding the monotonic and surjective alignment path $A : [1 : t_a] \to [1 : t_p]$ such that the objective function

$$\mathcal{L}_{\text{mle}} := \sum_{j=1}^{t_a} \log \mathcal{N}(\mathbf{z}_{a,j}; \mu_{A(j)}, \sigma_{A(j)}) \tag{1}$$

is maximized. $\mathcal{N}(z; \mu, \sigma)$ is the likelihood function of a Gaussian variable z parametrized by μ and σ. In each training step, the alignment path is first searched by the Viterbi algorithm, and then the maximum likelihood estimation over this path updates the parameters. $\hat{\mathbf{z}}_p$ therefore approximates $\mathbf{z}_a$ iteratively along the training process. The optimal alignment path A^* indicates the temporal duration $d^* := \{d_i^*\}_{i=1}^{t_p}$ of every phoneme $\mathbf{p}_{s,i}$ in terms of frames, such that the texts could be well aligned with the log-scale mel-spectrogram.To avoid inaccurate pronunciation of the generated voice in some words, we duplicate expand $\mathbf{z}_{p,i}$ by d_i^* frames and pass it into the downstream layers of E_p, which finally outputs the phonetic latent representation $\mathbf{z}_p^* \in \mathbb{R}^{t_a \times d}$. To encourage the audio and the phoneme models to encode common information, the l_2 loss between $\mathbf{z}_p^*$ and $\mathbf{z}_a$ is minimized.

$$\mathcal{L}_{\text{enc}} := \|\mathbf{z}_p^* - \mathbf{z}_a\|_2^2 . \tag{2}$$

The final source embedding $\mathbf{z}$ is obtained by randomly switching between $\mathbf{z}_p^*$ and $\mathbf{z}_a$, that means,

$$\mathbf{z} = k\mathbf{z}_p^* + (1 - k)\mathbf{z}_a \tag{3}$$

where $k \sim \text{Bernoilli}(0.5)$.

To encode the pitch information, we extract the F0 contours of the source audio data using CREPE [23] and represent them in terms of MIDI pitch numbers. In preliminary experiments, we found that the output of CREPE caused inconsistent spikes in some audio segments. Therefore, we apply a median filter with the size of three to remove the noise, and obtain our input pitch sequence $\mathbf{f}_s$. $\mathbf{f}_s$ is then converted to a sequence of pitch embedding and concatenated with $\mathbf{z}$ as the decoder input.

2.2 Target encoder

Similar to E_a, the target encoder E_t is also constructed mainly with stacking three Conv Blocks (see Figure 1 for the diagram of a Conv Block). E_t takes the log-scale mel-spectrogram $\mathbf{x}_t$ of the audio segments from the target singer's corpus as input and then outputs timbre features with different granularity levels from each Conv Block. D then processes singer information following the order from the highest to the lowest level. Inspired by [19], these Conv Blocks in E_t are linked to the Style-Adaptive Layer Normalization (SALN) Feed Forward Transformer (FFT) blocks [24,25] in the decoder D (see Figure 2 Section 2.3), resembling the encoder-decoder pathway in the U-net [26].

2.3 Decoder

The decoder D incorporates two decoding steps. First, the *feature transformation module* fuses the target singer information into the content latent representation. Second, the *mel-spectrogram generator* reconstructs the log-scale

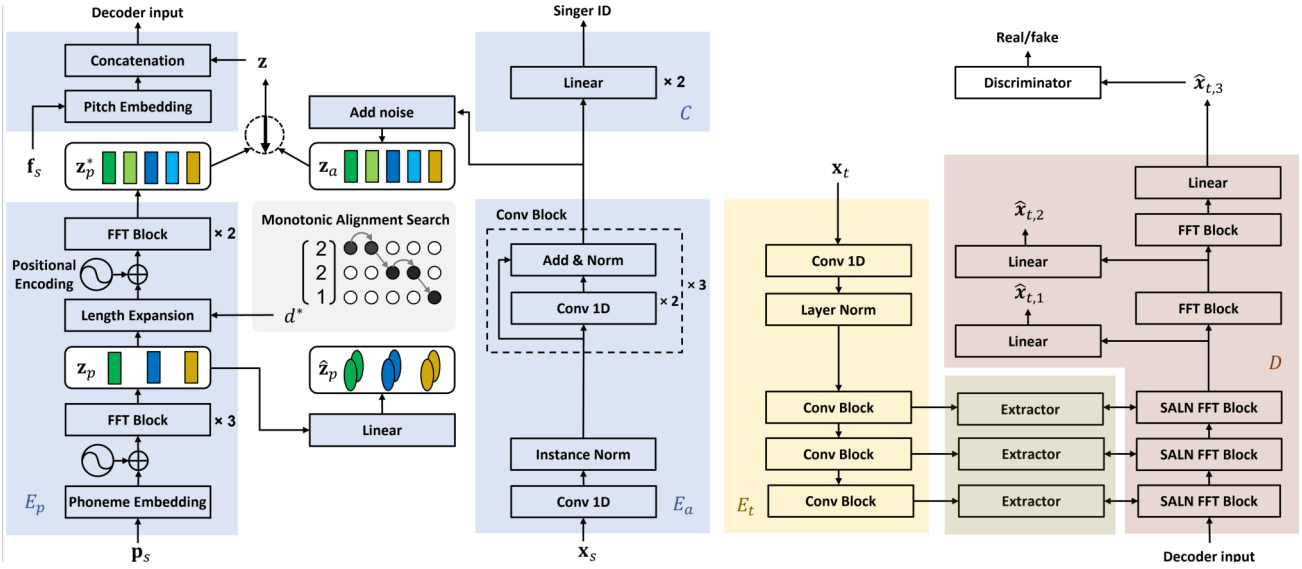

Figure 1. The architecture of the proposed model.

mel-spectrogram. The architecture of the feature transformation module is the same as in [25], in which the layer normalization layer in the FFT block [24] is replaced by a SALN layer [25]. Given an intermediate output h of the FFT and the target singer information w, the SALN operation can be formulated as follows:

$$y = g(w) \odot \text{LayerNormalization}(h) + f(w)\,, \quad (4)$$

where g and f are both linear layers and $\odot$ denotes element-wise multiplication.

The mel-spectrogram generator is comprised of two stacked FFT blocks and three output linear layers linked from the input/output nodes of the FFT blocks. This is similar to the idea of post-network [20, 27] using the FFT and linear layer [6, 24]. Each of the linear layers then outputs the log-scale mel-spectrogram. The outputted mel-spectrograms are denoted as $\hat{\mathbf{x}}_{t,1}$, $\hat{\mathbf{x}}_{t,2}$, and $\hat{\mathbf{x}}_{t,3}$. During training, suppose $\mathbf{x}_s$ is the ground-truth log-scale mel-spectrogram and the decoder is trained to minimize the average mel-spectrogram l_2 reconstruction loss:

$$\mathcal{L}_{\text{recon}} := \frac{1}{3}\sum_{i=1}^{3} \|\mathbf{x}_s - \hat{\mathbf{x}}_{t,i}\|_2^2\,. \quad (5)$$

2.4 Extractor

The extractor is an attention and layer normalization block [1] that connects each pair of Conv Block of E_t and the SALN FFT block of D, as shown in the right side of Figure 1. The extractor controls the mapping from the latent representation of the target singer outputted from the Conv Blocks of E_t to the latent representations of the source utterance such that they are structurally aligned. In other words, the attention mechanism guides E_t to extract the local timbre/style patterns that match the hidden latent states of D layer by layer in the training process [19].

[1] The affine transformation is removed in this block.

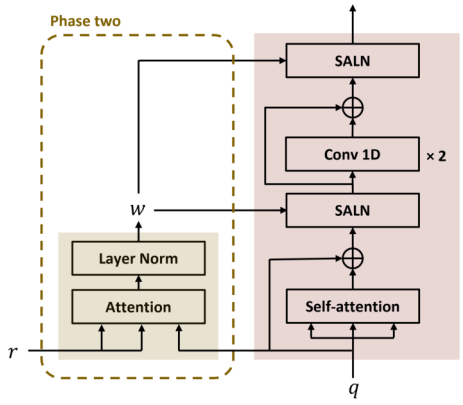

Figure 2. The SALN FFT block (right) with the Extractor (left). For the FFT blocks not using the SALN layers (e.g., the FFT blocks in E_p), the SALN layer is replaced by a layer normalization layer.

The processing of the extractor is illustrated in Figure 2. Let q be the input of one SALN FFT block of D and r be the latent representation of the target from its corresponding Conv Block of E_t. Each frame of q attempts to find local timbre/style patterns in r that have a similar structure to itself. And by aggregating them together, the extractor outputs an aligned local timbre/style representation w. More specifically, the attention mechanism is as follows:

$$Q = qW_q\,, \quad K = rW_k\,, \quad V = rW_v\,,$$
$$w = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V\,, \quad (6)$$

where W_q, W_k and W_v are learnable parameters.

It should be noted that for each generated result, its corresponding w and also r are usually learned from the concatenation of multiple target audios, which is a strategy for training speech synthesis and conversion systems [18, 19]. However, such a strategy still tends to generate discontinuous outputs in SVS and SVC, especially when deal-

ing with unseen singers. This is because singing exhibits much more diverse and combined expressions than speaking. Under this situation, an effective extractor should be able to recombine the features of various expressions from different audio recordings into a sentence. The softmax function in (6), however, fails to achieve this since it tends to give parsimonious attention maps.

To alleviate the situation mentioned above, we adopt the idea from video style transfer [28], which used an alternative way of computing the attention map by changing the Softmax function in Equation (6) to cosine similarity. The cosine similarity attention can be represented as:

$$M_{i,j} = \frac{S_{i,j}}{\sum_j S_{i,j}}, \quad S_{i,j} = \frac{Q_i \cdot K_j}{\|Q_i\|\|K_j\|} + 1, \\ w = MV, \tag{7}$$

where Q_i and K_j are the ith frame of Q and jth frame of K respectively, and $M_{i,j}$ is the (i,j)th element of the resulting new attention map M. Since the softmax function may overemphasize the contents of vocal fragments with similar phonetic structures [28], using cosine similarity ensures that the model combines much more diversified timbre structures and generates a smoother output.

2.5 Discriminator

Over-smoothing of the mel-spectrogram predictions is a common problem in singing voice generation due to the use of the reconstruction loss [29]. Like many previous studies, we add a discriminator (denoted as Disc) at the output stage of our model to alleviate this problem.

The discriminator architecture we use is similar to [30] but with two slight modifications. First, we remove the conditional projections that were intended to make predictions for multiple singers, and second, we divide the number of channels by 4 to balance the generator and discriminator performances. The input of the discriminator is a 128-frame mel-spectrogram segment randomly sampled from $\hat{\mathbf{x}}_{t,3}$, and the output is a single value. It should be noted that $\hat{\mathbf{x}}_{t,1}$ and $\hat{\mathbf{x}}_{t,2}$ are not taken as the input of the discriminator in our experimental setting. Based on the principle of the Least Square Generative Adversarial Network (LSGAN) [31], the proposed model is trained with the adversarial loss $\mathcal{L}_{\text{adv}}$ while the discriminator is trained with the discriminator loss $\mathcal{L}_{\text{Disc}}$ ($a = 0.3$ in this paper):

$$\mathcal{L}_{\text{adv}} := (\text{Disc}(\hat{\mathbf{x}}_{t,3}) - a)^2, \\ \mathcal{L}_{\text{Disc}} := (\text{Disc}(\hat{\mathbf{x}}_{t,3}))^2 + (\text{Disc}(\mathbf{x}_s) - a)^2. \tag{8}$$

2.6 Two-phase training process and inference

We adopt the two-phase training process [7] to train the whole system. During phase one, the source encoder and D are jointly trained. Since E_t and the extractors are inactive in this phase, the latent representation of target singer w is set to a fixed-size 1-dimensional vector, which is derived from a singer embedding lookup table followed by two linear layers with ReLU activation. The 1-dimensional w is then expanded to t_a frames and fed into the SALN module. The total loss during this phase is as follows:

$$\mathcal{L}_{\text{Phase-I}} := \mathcal{L}_{\text{recon}} + \lambda_{\text{enc}}\mathcal{L}_{\text{enc}} - \lambda_{\text{mle}}\mathcal{L}_{\text{mle}} - \mathcal{L}_C, \tag{9}$$

where the last two terms of the equation use a minus sign to represent maximization, and $\mathcal{L}_C$ is also used for updating C. We set the loss weights with $\lambda_{\text{enc}} = 0.5$ and $\lambda_{\text{mle}} = 0.1$.

During phase two, E_t, D, and the extractors are jointly trained, while the parameters of the source encoder are fixed. For the input of E_t, we randomly sampled 5 utterances from the source singer s and concatenate their log-scale mel-spectrogram along the time axis as $\mathbf{x}_t$. The total loss for updating the model during this phase is as follows:

$$\mathcal{L}_{\text{Phase-II}} := \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{adv}}, \tag{10}$$

and the discriminator Disc is updated by $\mathcal{L}_{\text{Disc}}$.

The process of randomly switching (see Equation (3)) is applied in both phases, and for the utterance without textual notation, we directly pass $\mathbf{z}_a$ as $\mathbf{z}$, letting the corresponding loss functions $\mathcal{L}_{\text{enc}}$ and $\mathcal{L}_{\text{mle}}$ not counted.

During inference, we specify either $\mathbf{z}_p^*$ or $\mathbf{z}_a$ as $\mathbf{z}$, depending on whether SVS or SVC is to be performed. $\mathbf{x}_t$ can be multiple utterances of a target singer. Since the input pitch $\mathbf{f}_s$ may not be within the target singer's pitch range, we shift $\mathbf{f}_s$ by the difference in median pitches:

$$\mathbf{f}_{\text{shift}} = \mathbf{f}_s - \text{median}(\mathbf{f}_s) + \text{median}(\mathbf{f}_t), \tag{11}$$

where $\mathbf{f}_t$ is the pitch sequence of $\mathbf{x}_t$ and $\mathbf{f}_{\text{shift}}$ is the shifted pitch for model input [15]. We take the last layer output $\hat{\mathbf{x}}_{t,3}$ of the model as the resulting mel-spectrogram.

Dataset	d	p	d/p	Type	Text
MPOP600	10	4	150	singing	✓
Musdb-V	2.3	86	1.6	singing	✗
NUS-48E	2.8	12	14	singing/speech	✓
VCTK	44	109	24.2	speech	✓

Table 1. The datasets employed in this paper. From left to right: dataset name, total duration (d, in hours), number of speaker/singer (p), average duration per speaker/singer (d/p, in minutes), type of content (singing, speech, or both), and the availability of text labels.

3. EXPERIMENTAL SETUP

3.1 Datasets

Four public datasets are used in our experiment. First, the MPOP600 [32] dataset contains 600 Mandarin pop songs sung by two male and two female vocalists. Second, the NUS-48E [33] dataset consists of 48 English popular songs performed by 12 singers. Third, the VCTK corpus [34] is a multi-speaker speech dataset for TTS and voice conversion tasks and has been widely used in many singing voice generation systems.Finally, Musdb-V is the vocal tracks manually collected from MUSDB18 [35], a dataset for music

source separation, containing 150 songs in different genres along with their isolated drums, bass, vocals, and other stems.Details of these datasets are listed in Table 1. The total duration of the data achieves 59.1 hours.

We randomly select 10 singers from Musdb-V as our test set; they are the so-called unseen singers during training. The remaining data are then partitioned into training and validation sets by 95:5 for each singer/speaker. All the audios are resampled to 24kHz, and the log-scale mel-spectrograms with 80 bins are computed by short-time Fourier transformation (STFT) using the size of Fast Fourier Transform, window size, and hop size of 2,048, 1,200, and 300, respectively. For MPOP600 and VCTK, we convert the text into phoneme sequence using pypinyin [2] and phonemizer [3], respectively. As for NUS-48E, we use the phoneme labels provided in the dataset.

3.2 Implementation details

The dimension of all hidden layers is set as 128 for the source encoder and 256 for both D and E_t. In each block, the kernel size in the two-layer 1D-convolution are set to 3 and 1, respectively, and the kernel sizes in the first 1D-convolution layer of E_a and E_t are both set to 3. The number of attention heads is set to 2. We use the AdamW optimizer with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, and follow the same learning rate schedule in [19]. Both phase one and phase two are trained for 250k steps with batch size being 8, and the discriminator joins the training process of phase two after 150k steps. We use Parallel WaveGAN (PWG) [36] to convert log-scale mel-spectrograms to waveforms. The PWG [4] is trained for 1M steps using MPOP600, NUS-48E and Musdb-V.

3.3 Evaluation methods

We conduct the evaluation with four scenarios: 1) **Unseen SVC**, conversion between singers in the test set; 2) **Seen SVC**, conversion between singers in the training set; 3) **Unseen SVS**, synthesizing singers in the test set from the utterance in the validation set; [5] and 4) **Seen SVS**, synthesizing singers in the training set from the utterance of another speaker/singer also in the training set. In addition, the VCTK and the speech part of NUS-48E are excluded from the evaluation. For each round of generation, one utterance of the source singer and five utterances of the target singer are randomly selected.

For objective evaluation, we use a speaker verification (SV) system [37] to evaluate the similarity between the target singer and the generated singing voice. The SV system [6] is trained from scratch using all four datasets for 500 epochs. The loss function and the model we use is AM-Softmax and ResNetSE34L, respectively. The resulting model takes an utterance as input and outputs a fix-dimensional embedding. We then compute the cosine similarity between the embedding of the generated singing voice and the average embedding of the five target utterances. The averaged cosine similarity of the randomly generated 1,000 utterances is then reported.

For subjective evaluation, we conduct a Mean Opinion Score (MOS) test. For each evaluation scenario, 50 utterances are randomly generated for evaluation. Each participant was asked to listen to the source utterance, target utterances, and the generated audios , and rated the generated singing voice in terms of two metrics: perceptual naturalness and similarity to the target. Both metrics are rated from 1 to 5, with 5 representing the best performance and 1 being the worst. We report the averaged scores with the 95% confidence intervals for each model.

Model	Unseen		Seen	
	SVC	SVS	SVC	SVS
Baseline SVC	0.059	–	0.213	–
Baseline SVS	–	0.084	–	–*
Fragment SVC/SVS	0.129	0.122	0.237	0.244
Proposed (S)	**0.294**	**0.232**	**0.536**	**0.479**
Proposed (S\Disc)	0.257	0.232	0.420	0.398
Proposed (C)	0.115	0.079	0.451	0.446

Table 2. Result of objective evaluation. *Result of seen Baseline SVS is not counted due to the inconsistency of training data (not supporting text-free training) compared to other seen SVS models.

Datasets	SVC	SVS
All	0.290	0.233
w/o MPOP600	0.292	0.225
w/o Musdb	0.178	0.145
w/o MPOP600 and w/o Musdb	0.194	0.149

Table 3. Objective similarity scores with reduced training data. All four settings are trained with Proposed (S) under the unseen-to-unseen scenario.

3.4 Models for comparison

Three variants of the proposed model are considered in the evaluation. They are the model using the softmax-based attention in the extractor (denoted as **Proposed (S)**), the model using cosine similarity attention (denoted as **Proposed (C)**), and the model without the discriminator while using softmax-based attention (denoted as **Proposed (S\Disc)**). These model variants are used to compare how the attention types and the discriminator affect the performance. Besides, three baseline models are considered and described as follows.

Fragment (SVC/SVS) is adapted from FragmentVC, a state-of-the-art voice conversion model which also employs the softmax attention in the extractors [19].In our implementation, we simply replace the pretrained Wav2Vec [38] source encoder in FragmentVC with our pretrained source encoder. The remaining architecture and the train-

[2] https://github.com/mozillazg/python-pinyin
[3] https://github.com/bootphon/phonemizer
[4] https://github.com/kan-bayashi/ParallelWaveGAN
[5] It should be noted that there is no text annotation in our test set.
[6] https://github.com/clovaai/voxceleb_trainer

Task	Model	Unseen singers		Seen singers	
		Similarity	Naturalness	Similarity	Naturalness
SVC	Baseline (SVC)	3.14 ± 0.17	3.29 ± 0.16	3.20 ± 0.18	3.21 ± 0.16
	Proposed (S)	$\mathbf{3.61 \pm 0.16}$	3.27 ± 0.17	3.63 ± 0.16	3.27 ± 0.16
	Proposed (C)	3.56 ± 0.17	$\mathbf{3.53 \pm 0.17}$	$\mathbf{3.70 \pm 0.16}$	$\mathbf{3.46 \pm 0.16}$
SVS	Baseline (SVS)	3.14 ± 0.18	2.98 ± 0.16	$\mathbf{3.88 \pm 0.15}$	$\mathbf{3.39 \pm 0.16}$
	Proposed (S) w/o Musdb-V	$\mathbf{3.18 \pm 0.17}$	$\mathbf{3.06 \pm 0.17}$	3.87 ± 0.15	3.32 ± 0.17

Table 4. Subjective test of both the SVC and SVS tasks. MOS and the 95% confidence interval are shown for each case.

ing process follow the original implementation.

Baseline (SVC) is adapted from [15], the state-of-the-art zero-shot SVC model, which is an adaptation of AutoVC [20] and uses the WORLD vocoder [39] to synthesize the singing voice. In our modification of [15], we replace the WORLD vocoder with the PWG vocoder by changing the output layer to generate mel-spectrogram and concatenating a pitch embedding with the same dimension of the content encoding. This is to ensure fair comparison since WORLD vocoder may lead to sound quality degradation in comparison with PWG [40]. During training and inference, the singer embedding is generated by feeding the five target utterances to the speaker encoder and averaging the resulting embeddings.

Baseline (SVS) is adapted from the singing model of DeepSinger, also a state-of-the-art SVS model [6]. In our implementation, we change the output layer of this model to generate mel-spectrograms, and the phoneme duration is extracted by a commonly used open source tool [41]. The major difference between Baseline (SVS) and our proposed model is that the baseline model needs phoneme duration provided by another speech-text alignment system, while our model estimates phoneme duration directly from our self-supervised source encoder.

For the objective test, the six models described above are all compared under the four scenarios. As for the subjective test, to reduce participants' loading, we only compare Baseline (SVC), Proposed (S), and Proposed (C) for SVC, and compare Baseline (SVS) and Proposed (S) (w/o Musdb-V) for SVS. it should be noted that in SVS, Proposed (S) is particularly trained without Musdb-V; this is again for a fair comparison since Baseline (SVS) does not support training with text-free data. [7]

4. RESULTS

Table 2 shows the objective evaluation results in terms of similarity score. The three proposed models generally outperform the baselines. The effectiveness of using the attention-based extractor is verified by comparing Baseline SVC/SVS to the other four models. Also, the advantage of the SALN layer is shown by comparing Fragment SVC/SVS and Proposed (S). The improvement using the discriminator is also shown. The softmax-based attention achieves higher similarity than cosine similarity attention.

Table 3 compares the objective similarity scores trained on reduced sizes of data for the unseen-to-unseen case. It shows that the Musdb-V dataset plays a crucial role in improving the similarity score, and implies that a training dataset with more singers might benefit from singing voice generation more than a large dataset in zero-shot singing voice generation.

Table 4 shows the MOS results responded by 62 subjects for both the SVC and SVS tasks. For the SVC task, both Proposed (S) and Proposed (C) outperform the Baseline (SVC) considerably in terms of perceptual similarity for both seen and unseen cases. As for naturalness, Proposed (C) outperforms both Baseline (SVC) and Proposed (S) substantially. The cosine similarity attention achieves better naturalness and verifies the argument that it can better combine timbre structures and generate smoother outputs, while softmax-based attention achieves better similarity, in line with objective evaluation (see Table 2).

For the SVS subjective test, our degenerated model (Proposed (S) without Musdb-V) shows results comparable to Baseline (SVS) in general, while it still slightly outperforms the baseline for both similarity and naturalness in the unseen-to-unseen case. Such a comparable result can be explained by the finding in Table 3: having a singer-diverse dataset (e.g., Musdb-V) is critical for zero-shot singing voice generation. Since lyrical annotations are not always available for such datasets, a unified model that supports both audio- and text-based training is apparently more flexible, and could also jointly improve the performance of the SVS and SVC models.

5. CONCLUSION

We have presented a unified model which jointly supports the zero-shot SVC and SVS tasks, and has achieved state-of-the-art performance on both tasks. Our experiments also show that the design of the attention mechanism determines the trade-off between perceptual similarity and naturalness, and that a dataset containing a large number of singers is critical in improving zero-shot SVS and SVC. These two directions are suggested for future research on zero-shot singing voice generation.

[7] Since our proposed model allows either audio or text input, we can surely incorporate the text-free Musdb-V set but train the model for the SVS task. In our pilot study, we also observed that incorporating Musdb-V did improve the perceptual quality of SVS results. Such improvement with Musdb-V is also seen in the ablation study (Table 3). However, in order not to take advantage of the Baseline (SVS) model, we opt to degenerate Propose (S) (i.e. without Musdb-V) in our subjective test.

6. REFERENCES

[1] J. Bonada and X. Serra, "Synthesis of the singing voice by performance sampling and spectral models," *IEEE Signal Process. Mag.*, vol. 24, no. 2, pp. 67–79, 2007.

[2] T. Nakano and M. Goto, "Vocalistener2: A singing synthesis system able to mimic a user's singing in terms of voice timbre changes as well as pitch and dynamics," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2011, pp. 453–456.

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

[4] M. Blaauw, J. Bonada, and R. Daido, "Data efficient voice cloning for neural singing synthesis," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6840–6844.

[5] S. Choi, W. Kim, S. Park, S. Yong, and J. Nam, "Korean singing voice synthesis based on auto-regressive boundary equilibrium gan," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7234–7238.

[6] Y. Ren, X. Tan, T. Qin, J. Luan, Z. Zhao, and T. Liu, "DeepSinger: Singing voice synthesis with data mined from the web," in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2020, pp. 1979–1989.

[7] J. Bonada and M. Blaauw, "Semi-supervised learning for singing synthesis timbre," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 7083–7087.

[8] E. Nachmani and L. Wolf, "Unsupervised singing voice conversion," *Interspeech*, pp. 2583–2587, 2019.

[9] A. Polyak, L. Wolf, Y. Adi, and Y. Taigman, "Unsupervised cross-domain singing voice conversion," in *Interspeech*, 2020, pp. 801–805.

[10] Y. Luo, C. Hsu, K. Agres, and D. Herremans, "Singing voice conversion with disentangled representations of singer and vocal technique using variational autoencoders," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 3277–3281.

[11] N. Takahashi, M. K. Singh, and Y. Mitsufuji, "Hierarchical disentangled representation learning for singing voice conversion," in *International Joint Conference on Neural Networks (IJCNN)*, 2021, pp. 1–7.

[12] S. Liu, Y. Cao, N. Hu, D. Su, and H. Meng, "Fastsvc: Fast cross-domain singing voice conversion with feature-wise linear modulation," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*, 2021, pp. 1–6.

[13] L. Wan, Q. Wang, A. Papir, and I. Lopez-Moreno, "Generalized end-to-end loss for speaker verification," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4879–4883.

[14] Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, J. Shen, F. Ren, Z. Chen, P. Nguyen, R. Pang, I. Lopez-Moreno, and Y. Wu, "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," in *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018 (NeurIPS)*, 2018, pp. 4485–4495.

[15] S. Nercessian, "Zero-shot singing voice conversion," in *Proceedings of the 21th International Society for Music Information Retrieval Conference (ISMIR)*, 2020, pp. 70–76.

[16] ——, "Improved zero-shot voice conversion using explicit conditioning signals," in *Interspeech*, 2020, pp. 4711–4715.

[17] L. Zhang, C. Yu, H. Lu, C. Weng, C. Zhang, Y. Wu, X. Xie, Z. Li, and D. Yu, "DurIAN-SC: Duration informed attention network based singing voice conversion system," in *Interspeech*, 2020, pp. 1231–1235.

[18] S. Choi, S. Han, D. Kim, and S. Ha, "Attentron: Few-shot text-to-speech utilizing attention-based variable-length embedding," in *Interspeech*, 2020.

[19] Y. Y. Lin, C.-M. Chien, J.-H. Lin, H.-y. Lee, and L.-s. Lee, "Fragmentvc: Any-to-any voice conversion by end-to-end extracting and fusing fine-grained voice fragments with attention," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5939–5943.

[20] K. Qian, Y. Zhang, S. Chang, X. Yang, and M. Hasegawa-Johnson, "AutoVC: Zero-shot voice style transfer with only autoencoder loss," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, vol. 97, 2019, pp. 5210–5219.

[21] E. Cooper, C. Lai, Y. Yasuda, F. Fang, X. Wang, N. Chen, and J. Yamagishi, "Zero-shot multi-speaker text-to-speech with state-of-the-art neural speaker embeddings," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6184–6188.

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

[23] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 161–165.

[24] Y. Ren, Y. Ruan, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu, "Fastspeech: Fast, robust and controllable text to speech," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[25] D. Min, D. B. Lee, E. Yang, and S. J. Hwang, "Meta-stylespeech: Multi-speaker adaptive text-to-speech generation," in *International Conference on Machine Learning*, 2021, pp. 7748–7759.

[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015, pp. 234–241.

[27] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Ryan, R. A. Saurous, Y. Agiomyrgiannakis, and Y. Wu, "Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.

[28] S. Liu, T. Lin, D. He, F. Li, M. Wang, X. Li, Z. Sun, Q. Li, and E. Ding, "Adaattn: Revisit attention mechanism in arbitrary neural style transfer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6649–6658.

[29] J. Chen, X. Tan, J. Luan, T. Qin, and T.-Y. Liu, "HiFiSinger: Towards high-fidelity neural singing voice synthesis," *arXiv preprint arXiv:2009.01776*, 2020.

[30] T. Kaneko, H. Kameoka, K. Tanaka, and N. Hojo, "StarGAN-VC2: Rethinking conditional methods for stargan-based voice conversion," in *Interspeech*, 2019.

[31] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.

[32] C. Chu, F. Yang, Y. Lee, Y. Liu, and S. Wu, "Mpop600: A mandarin popular song database with aligned audio, lyrics, and musical scores for singing voice synthesis," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2020, pp. 1647–1652.

[33] Z. Duan, H. Fang, B. Li, K. C. Sim, and Y. Wang, "The NUS sung and spoken lyrics corpus: A quantitative comparison of singing and speech," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. IEEE, 2013, pp. 1–9.

[34] J. Yamagishi, C. Veaux, and K. MacDonald, "CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit (version 0.92)," 2019.

[35] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," Dec. 2017.

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

[37] J. S. Chung, J. Huh, S. Mun, M. Lee, H. Heo, S. Choe, C. Ham, S. Jung, B. Lee, and I. Han, "In defence of metric learning for speaker recognition," in *Interspeech*, 2020, pp. 2977–2981.

[38] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, (NeurIPS)*, 2020.

[39] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. Inf. Syst.*, vol. 99-D, no. 7, pp. 1877–1884, 2016.

[40] J. Liu, C. Li, Y. Ren, F. Chen, P. Liu, and Z. Zhao, "DiffSinger: Diffusion acoustic model for singing voice synthesis," *CoRR*, vol. abs/2105.02446, 2021.

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