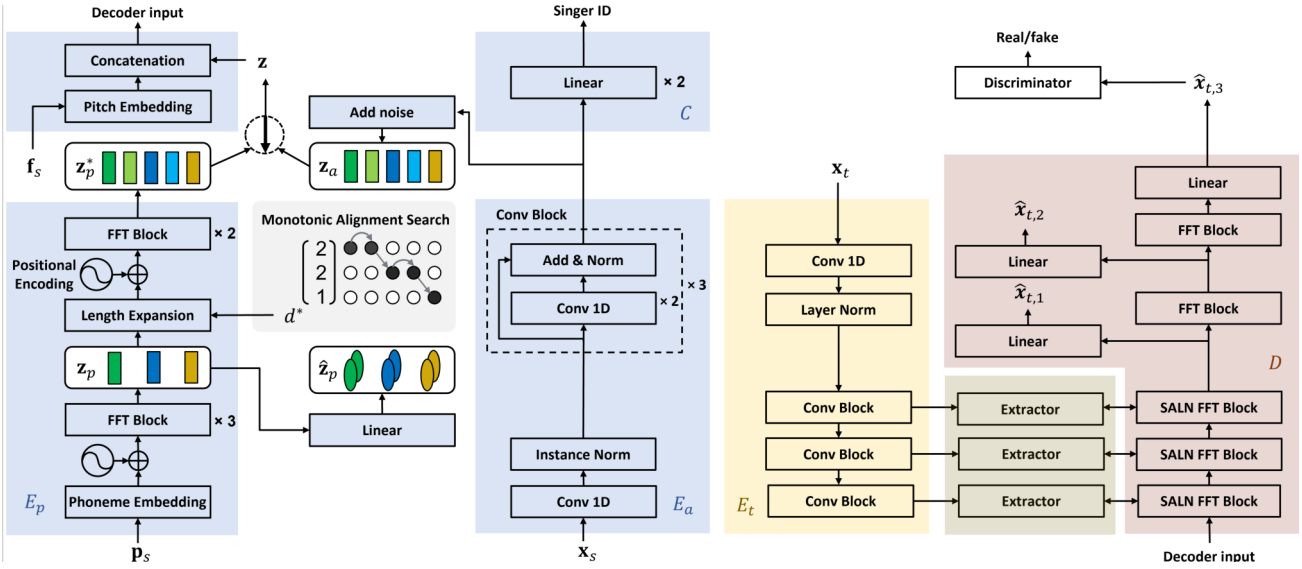

Figure 1. The architecture of the proposed model.

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.

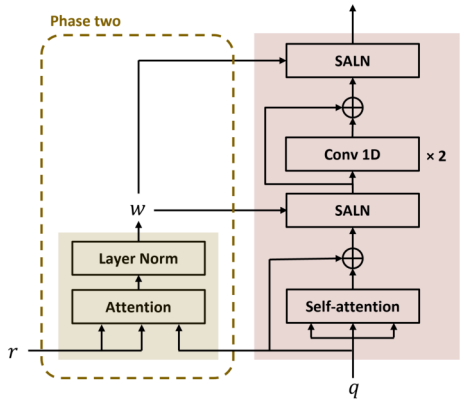

Figure 2. The SALN FFT block (right) with the Extractor (left). For the FFT blocks not using the SALN layers (e.g., the FFT blocks in E_p), the SALN layer is replaced by a layer normalization layer.

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.

SEMANTIC CONTROL OF GENERATIVE MUSICAL ATTRIBUTES

Stewart Greenhill **Majid Abdolshah** **Vuong Le**
Sunil Gupta **Svetha Venkatesh**
Applied Artificial Intelligence Institute, Deakin University, Australia
s.greenhill@deakin.edu.au

ABSTRACT

Deep generative neural networks have been successful in tasks such as composing novel music and rendering expressive performance. Controllability is essential for building creative tools from such models. Recent work in this area has focused on disentangled latent space representations, but this is only part of the solution. Efficient control of semantic attributes must handle non-linearities and holes that occur in latent spaces, whilst minimising unwanted changes to other attributes. This paper introduces SeNT-Gen, a neural traversal algorithm that uses a secondary neural network to model the complex relationships between latent codes and musical attributes. This enables precise editing of semantic attributes that adapts to context. We demonstrate the method using the dMelodies dataset, and show strong performance for several VAE models.

1. INTRODUCTION

Deep generative models show promise for music, with systems that can compose melodies and accompaniments, render expressive performances, or synthesise instrumental sounds and singing voices [1, 2]. While many of these work as "black boxes," *controllability* is an essential factor in creative tools for musicians. This raises important challenges such as controlling musically meaningful aspects of the composition, balancing creativity versus imitation, providing interactivity and refinement, and producing a convincing temporal structure that has a sense of direction [3].

In generative applications a neural network is trained to find a low-dimensional representation for complex data. A common approach uses a Variational Auto-Encoder (VAE) in which an *encoder* learns to transform the data space X into a simpler "latent space" Z. For music X is a representation of a score such as a piano-roll that defines the pitch and duration of notes over time. A *decoder* recovers the original representation from the latent samples. In learning the latent representation the network identifies the essential attributes needed to construct the melody. The dimensions of the latent space represent semantic attributes such as note density [4], syncopation [5], genre [6] or arousal [7]. Novel melodies can be generated by decoding samples drawn from the latent space. Moving in the latent space adjusts the semantic attributes, for example: to smoothly "morph" between two melodies by interpolating a path between their latent positions.

Recent work on control of generative music has focused on regularisation of the latent space, and disentanglement of the semantic attributes. The aim is to relate each semantic attribute of the melody to a unique dimension of the latent space, which allows attributes to be adjusted by adding an "attribute vector" [8, 9] in latent space. This approach involves some challenges. First, there is often a tradeoff between regularisation of the latent space and reconstruction quality. Second, effective disentanglement can only be achieved through supervision [10] and unsupervised approaches are sensitive to inductive biases in both the data set and learning model (such as network model, hyperparameters, random seeds). Third, the relationship between the latent dimension and corresponding semantic attribute value may be non-linear. Fourth, latent spaces may contain "holes" where the decoder produces invalid results [11] and these must be avoided when adjusting attributes, or interpolating paths in the latent space. This means that some additional work is required beyond disentanglement to properly control the values of semantic attributes.

This paper proposes a new method to efficiently traverse latent spaces frequently used in generative music. We introduce the Semantic Neural Latent Traversal method for Generative models (SeNT-Gen), which employs a secondary neural network to model the complex relationship between latent codes and semantic attributes. The SeNT-Gen traversal function predicts the new latent position given a starting position and attribute change, supporting non-linear relationships and adapting to the context of the traversal. We evaluate the performance of SeNT-Gen for musical control using the dMelodies [12] data set.

Our key contributions are:

1. A neural method to traverse the latent space, targeting precise changes to musical attributes, and supporting non-linear contextual relationships between the VAE latent space and semantic attributes;

2. Experiments and analysis of the proposed method using the established dMelodies data set. The method is independent of learning model, and shows best performance for strongly regularised models;

3. Objective quantitative metrics to measure performance of the method through targeted attribute changes, which improves on notions of "controllability" that are tied to interpolation heuristics.

2. RELATED WORK

Control of generative music is usually based on latent space representations. Some of these approaches are surveyed here, while alternative approaches for images do not directly translate between image and musical domains [12]. Music-VAE [5] uses a hierarchical recurrent VAE to model temporal structure in music. Attribute vectors are shown to partially control attributes such as *note density*, *melodic interval* and *rhythmic syncopation*. MIDI-VAE [6] learns songs using a a parallel VAE with shared latent space, and an attached style classifier enables a chosen *style* (classic, jazz, pop, Bach, Mozart) to be applied to the output. GLSR-VAE [4] uses a novel regularisation method to control *note density* of chorale-style melodies. Music FaderNets [7] proposes a Gaussian mixture VAE for learning piano performance, with control over the *arousal* or "energy level" based on rhythm and note density. AR-VAE [13] defines a supervised regularisation method which is applied to controlling *rhythmic complexity*, *pitch range*, *note density*, and *contour* for melodies trained from chorales and folk tunes. Kawai and colleagues [14] use a VAE with adversarial classifier-discriminator to condition the decoder on semantic attributes. This model is trained on folk tunes to control orthogonal attributes such as *number of notes*, *pitch variability*, *chromatic motion* and *amount of arpeggiation*.

Apart from latent space approaches, other methods generally require ad-hoc modifications to the probability distribution of a model. Transformers are used to generate music with long-term structure [15], but control is limited to providing a prompt stimulus for the system to extend. Pop Music Transformer [16] uses a Transformer-XL model to learn expressive piano performances. Control over *tempo* and *chord* is achieved by masking out corresponding event probabilities in the model output. Coconet [17] uses a convolutional model to generate chorales in the style of Bach. Cococo [18] adds "semantic sliders" for *conventional/surprising* and *happy/sad* output using "soft priors" to modify the model's sampling distribution.

Several factors make it difficult to compare the results of these studies. Firstly, there are a wide variety of model architectures, and each implements its own encoding of the training data into a form suitable for model training. This in turn depends on the data-sets used, which vary from simple scores in ABC notation, to traditional scores, to MIDI transcriptions of actual performances either recorded from a digital keyboard or automatically extracted from audio recordings. Secondly, while there is a focus on disentangled representations, these are not in themselves control methods. Even in a disentangled latent space, more work is required to accurately achieve a specific value of an attribute, dealing with potential non-linearities and holes in the latent space. In the absence of a particular control algorithm some works assess "controllability" through interpolation [7, 11, 14] though there is no consistency in the metrics used.

Feature	Values	Description
Tonic	12	Notes C, C#, D ..., B
Octave	3	Octave 4, 5, or 6
Scale	3	major, minor, or blues
Rhythm Bar 1, 2	28	$\binom{8}{6}$ codes for 6 note onsets
Arp Chord 1, 2, 3, 4	2	arpeggio direction up or down

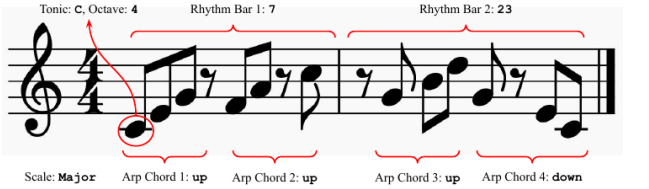

Figure 1. Each dMelodies two-bar melody is described by 9 attributes. *Tonic*, *Octave* and *Scale* apply to the entire melody. Separate *Rhythm* attributes apply to each bar. *Arpeggio* attributes apply to each of the 4 chords. The table (top) shows the number of values for each attributes, and an example from [12] is shown below.

The dMelodies dataset [12] has been proposed for evaluation of musical disentanglement learning, similar to dSprites in the image domain [19]. dMelodies includes 1,354,752 unique two-bar melodies, algorithmically constructed with independent factors of variation, with each comprised of 4 arpeggios in a I-IV-V-I chord pattern (see Table 1). Attributes are discrete, and most (except for *Tonic* and *Octave*) are categorical. dMelodies provides three unsupervised reference models including β-VAE [20], and a subsequent paper [11] adds three supervised models: I-VAE [21], AR-VAE [13], and S2-VAE [22]. Three measures of disentanglement are implemented [12]: *Mutual Information Gap* [23], *Modularity* [24], and *Separated Attribute Predictability* [25]. On these measures, supervised learning is shown to give better disentanglement, without sacrificing reconstruction quality [11].

2.1 Metrics

Suppose the VAE decoder $\mathcal{G}$ generates an object $x = \mathcal{G}(z)$ for latent sample z, and let $c = [c_1, c_2, ..., c_K]$ be the K semantic attributes of x. Disentanglement is formalised through the concepts of *consistency* and *restrictiveness* [26] and establishes a relationship between a latent dimension z_i and a corresponding semantic attribute c_i. *Consistency* says that when z_i is fixed, c_i of the generated x never changes. *Restrictiveness* says that when only z_i is changed, the change is restricted to c_i and there is no change to other attributes c_j for $j \neq i$.

Attributes may be controlled through a traversal function $\mathcal{T}_k(z, c_k, c_k^*)$ which predicts the latent value z^* required to change attribute k of z from c_k to c_k^*. The accuracy of $\mathcal{T}_k$ can be evaluated for a set of changes (z, c_k, c_k^*) by measuring the deviation of the result from the target, and any side effects on non-target attributes. Alternative approaches assess "controllabilty" in the absence of such a traversal function. Instead, interpolation is used to tra-