SCORE AND LYRICS-FREE SINGING VOICE GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models for singing voice have been mostly concerned with the task of "singing voice synthesis," i.e., to produce singing voice waveforms given musical scores and text lyrics. In this work, we explore a novel yet challenging alternative: singing voice generation without pre-assigned scores and lyrics, in both training and inference time. In particular, we experiment with three different schemes: 1) *free singer*, where the model generates singing voices without taking any conditions; 2) *accompanied singer*, where the model generates singing voices over a waveform of instrumental music; and 3) *solo singer*, where the model improvises a chord sequence first and then uses that to generate voices.

We outline the associated challenges and propose a pipeline to tackle these new tasks. This involves the development of source separation and transcription models for data preparation, adversarial networks for audio generation, and customized metrics for evaluation.

1 INTRODUCTION

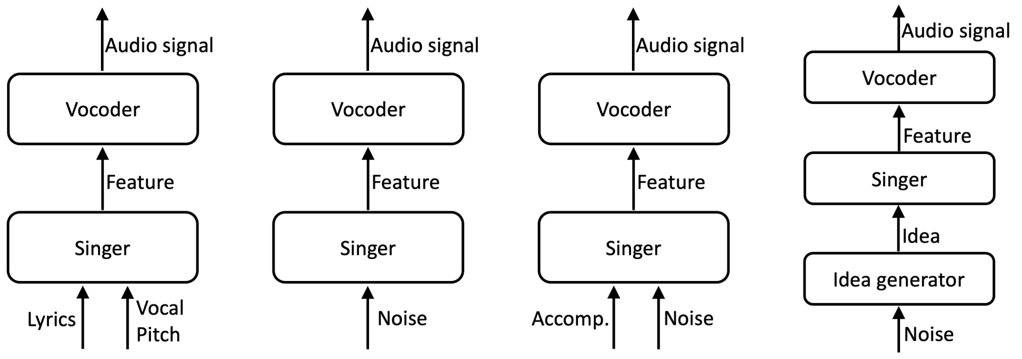
The task of computationally producing singing voices is usually referred to as singing voice *synthesis* in the literature (Cook, 1996). Most researchers assume that the note sequence and the lyrics of the waveform to be generated are given as the model input,[1] and aim to build synthesis engines that sound as natural and expressive as a real singer (Blaauw et al., 2019; Hono et al., 2019; Kaewtip et al., 2019; Lee et al., 2019; Tamaru et al., 2019). As such, the content of the produced singing voice is largely determined by the given model input, which is usually assigned by human. And, accordingly, progress in singing voice synthesis has followed closely with that in text-to-speech (TTS) synthesis (Shen et al., 2017; Gibiansky et al., 2017; Bonada & Serra, 2007).

However, we argue that singing according to a pre-assigned musical score and lyrics is only a part of the human singing activities. For human beings, singing can also be a spontaneous activity. We learn to spontaneously sing when we were children (Dowling, 1984). We do not need a score to sing when we are humming on the road or in the bathroom. The voices sung do not have to be intelligible. Jazz vocalists can improvise according to a chord progression, an accompaniment, or even nothing.

We aim to explore such a new task in this paper: teaching a machine to sing with a training collection of singing voices, but without the corresponding musical scores and lyrics of the training data. Moreover, the machine has to sing without pre-assigned score and lyrics as well even in the inference (generation) time. This task is challenging in that, as the machine sees no lyrics at all, it hardly has any knowledge of the human language to pronounce or articulate either voiced or unvoiced sounds. And, as the machine sees no musical scores at all, it has to find its own way learning the language of music in creating plausible vocal melodies. It is the uncertainty and artistic freedom of the learning and generation processes that motivate us to pursue such a possibly anti-anthropocentric approach for creating musical audio. It also makes the task different from TTS.

Specifically, we consider three types of such score- and lyrics-free singing voice *generation* tasks, as shown in Figures 1(b)–(d). A *free singer* sings with only random noises as the input. An *accompanied singer* learns to sing over a piece of instrumental music, which is given as an audio waveform (again without score information). Finally, a *solo singer* also sings with only noises as the input, but

[1]Sometimes together with other conditional signals such as the singer identity and expression controls (e.g., vibrato extent/rate and intended emotions) (Hono et al., 2018). We do not consider them in this work.



(a) Singing synthesis (b) Free singer (c) Accompanied singer (d) Solo singer

Figure 1: Schematic diagrams of different schemes of singing voice generation. In the latter three, the model is provided with neither scores nor lyrics during both training and inference time.

it uses the noises to firstly generate some kind of 'inner ideas' of what to sing. The three tasks may have different use cases. In the latter two, the machine has to create not only human-like sounds of good quality but also sounds that harmonize well with the assigned background music.

We outline below the challenges associated with the proposed tasks and the solutions we investigate.

First, the tasks are fairly unsupervised as we hardly provide any labeled data (such as annotations of phonemes, pitches, or onset times) for the training singing files. The machine has to learn the complex structure of music directly from audio signals. We explore the use of generative adversarial network (GAN) (Goodfellow et al., 2014) to address this issue, for its demonstrated effectiveness for singing voice synthesis (Hono et al., 2019) and pitch-conditioned instrument note synthesis (Engel et al., 2019). Specifically, we design a novel GAN-based architecture to learn to generate the mel-spectrogram of singing voice, and then use WaveRNN (Kalchbrenner et al., 2018), a single-layer recurrent neural network, as the vocoder to generate the audio waveform. Rather than considering the mel-spectrograms as a fixed-size image as done in recent work on audio generation (Engel et al., 2019; Marafioti et al., 2019), we use gated recurrent units (GRUs) (Cho et al., 2014) and dilated convolutions (van den Oord et al., 2016) in both the generator and discriminator, to model both the local and sequential patterns in music and to facilitate the generation of variable-length waveforms.

Second, for training the free singer, unaccompanied vocal tracks are needed. As for the accompanied singer, we need additionally an accompaniment track for each vocal track. However, public-domain multi-track music data is hard to find. We choose to implement a vocal source separation model with state-of-the-art separation quality (Liu & Yang, 2019) for data preparation. The proposed pipeline for training and evaluating an accompanied singer is illustrated in Figure 2. We note that, as we do not need information regarding the vocal pitches and sung lyrics of the training data, we do not need extra efforts to label or segment the training clips. The advantage of having a vocal separation model is that we can use as many audio files (either mono or stereo) as we have to compile the training data. The downside is that the singing voice generation models may suffer from the artifact (Cano et al., 2018) of the source separation model.

Third, for the scenario addressed by the accompanied singer, we note that there are plenty of valid ways to sing over an accompaniment track. In other words, there is no single "ground truth" and the relationship between the model input and output is one-to-many. For diversity and artistic freedom, we cannot ask the machine to generate any specific singing voice in response to an accompaniment track, even if we have paired data of vocal and accompaniment tracks. We investigate using conditional GAN (Mirza & Osindero, 2014) to address this issue.

Fourth, as the proposed tasks are new, there is no established ways for performance evaluation. According to our setting, we desire our machine to generate audio waveforms with high quality and diversity, vocal-like timbre, plausible pitch contour, emotion expression, and, for the accompanied singer, that are in harmony with the given accompaniment track. But, the singing does not have to be intelligible. We propose customized objective and subjective metrics to evaluate our models in these

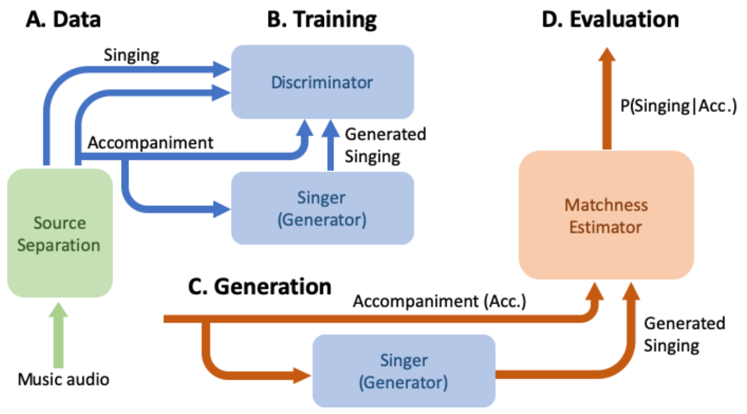


Figure 2: The proposed pipeline for building the accompanied singer. In step A, we apply source separation to professionally recorded audio files to get separated singing voice and accompaniment. In step B, we use the separated tracks to train the generators and discriminators in GAN. In step C, we can feed an unseen accompaniment to the trained singer model and let it "sing." One way to evaluate the result is to measure how the generated vocal is in harmony with the accompaniment.

aspects. For example, we adapt the melody harmonization model proposed by Lim et al. (2017) to measure the matchness between the generated vocal track and the given accompaniment track.

Finally, reproducibility is a major issue, especially for a subjective task. We intend to use publicly-available copyright-free instrumental music as the condition signals for testing the accompanied singer, so that other researchers can use the same testing conditions for model comparison in the future. We will also release the testing conditions for the solo singer, the generated singing voices for all our models, as well as open source our code through a public git repository [URL removed]. Samples of the generated result can be found at `http://bit.ly/2mIvoIc`.

2 Schemes of singing voice generation

In singing voice synthesis (Figure 1(a)), the musical scores and lyrics are provided to a singer model. Following the common approach in TTS (Shen et al., 2017; Gibiansky et al., 2017), the singer model tries to generate audio features such as the mel-spectrograms as the first step, which are then fed into a vocoder (Kalchbrenner et al., 2018; Prenger et al., 2018) to produce audio waveforms as the second step. As we will introduce below, the three proposed new voice generation tasks take no scores and lyrics, and they accordingly features higher, and different, degrees of freedom.

2.1 Free Singer

A free singer takes no conditions at all as the input. We want it to sing freely. The singing voices from a free singer may not even sound good, but they should sound like singing voice. A free singer is like we are freely humming or singing on the road walking or in the bathroom taking a shower. We may not even know what we are singing and likely there is no underlying musical score.

From the viewpoint of a generative model, training a free singer amounts to modeling a distribution $P(\mathbf{Y})$, where $\mathbf{Y} \in \mathbb{R}^{K \times T}$ is a matrix representing a sequence of K-dimensional features and T is the number of time frames. A free singing is sampled from this distribution without conditions.

2.2 Accompanied Singer

An accompanied singer takes as the input a sequence of accompaniment-derived features. An accompanied singer tries to generate singing voices that match the accompaniment track in some way. It is similar to the case of Karaoke, where a backing accompaniment track is played from a speaker, the lyrics and a video are displayed on a screen, and a user tries to sing according to the lyrics and the backing track. The difference is that, this time the user is a trained model and we do not ask it to follow the lyrics or the exact pitch contour of the accompaniment. The note sequence found in the singing has to be in harmony with, but not a duplicate of, that in the backing track.

Training an accompanied singer amounts to modeling a distribution $P(\mathbf{Y}|\mathbf{A})$, where $\mathbf{Y} \in \mathbb{R}^{K \times T}$ represents a feature sequence of the vocal track, and $\mathbf{A} \in \mathbb{R}^{H \times T}$ is a feature sequence of the given accompaniment track. In our implementation, we use the mel-spectrograms for $\mathbf{Y}$ in compliance with the need of the vocoder. For $\mathbf{A}$, different features can be tried, and we investigate using a transcription model (Hawthorne et al., 2018) to extract pitch features. See Section 4.1 for details.

2.3 Solo Singer

A solo singer is similar to a free singer in that both takes no conditions as the input. However, a solo singer would generate an 'inner idea' first, and then sing according to that. In other words, it learns a joint distribution $P(\mathbf{Y}|\mathbf{I})Q(\mathbf{I})$, where $\mathbf{I} \in \mathbb{R}^{J \times T}$ is a matrix representing the idea sequence. The singer first samples $\mathbf{I}$ from a distribution Q, and then uses that to conditionally sample $\mathbf{Y}$ from P. The inner idea $\mathbf{I}$ can take several forms. In this work, we instantiate this scheme with $\mathbf{I}$ being a chord progression (namely a sequence of chord labels). The distribution Q is modeled by an auto-regressive recurrent network we build for chord progression generation (with a network architecture adapted from that in (Waite et al., 2016)), as described in Section 3.3.

Alternatively, we can think of a solo singer as a combination of an idea generator and an an accompanied singer. For an accompanied singer, the information extracted from the given accompaniment track can take several forms such as transcribed pitches and chord progressions. A solo singer learns to generate such information on its own, without reference to an actual accompaniment track.

3 Models

To account for the absence of supervised data and the highly complicated spatio-temporal patterns in audio spectrograms, we propose a new adversarial net that features heavy use of GRUs (Cho et al., 2014; Chung et al., 2014), dilated convolutions (van den Oord et al., 2016; Sercu & Goel, 2016), and feature grouping to build our singer models. We provide the algorithmic details below.

3.1 Block of GRU-Grouped Dilated Convolution-Group Normalization

Network architectures with stacked blocks of GRUs and dilated convolutions have been used to attain state-of-the-art performance in blind musical source separation (Liu & Yang, 2019). In a source separation task, a model learns to decompose, or unmix, different sources (e.g., vocal, piano, bass, drum) from a mixture signal (Rafii et al., 2018). This requires the abilities to model the relationships between different sources as well as the relationships between neighboring time frames. The output spectrograms are also expected to be distortion-less and of high audio quality. We like to investigate whether such blocks can help generating plausible singing voices in an unsupervised way. Especially, we want the singer models to also consider accompaniment information.

Specifically, one such block we adopted in our models is a stack of GRU, dilated convolution with feature grouping, and group normalization (Wu & He, 2018). The input to the GRU, the output of the GRU, and the output of the group normalization are summed to form the output of the block. We note that the original 'D2 block' used in (Liu & Yang, 2019) uses dilated GRU and uses weight normalization (Salimans & Kingma, 2016) for the dilated convolution layers. However, empirically we find that it is easier for the singer models to converge by replacing weight normalization with group normalization, and using plain GRUs is as good as using dilated GRUs. We refer to our blocks as GRU-grouped dilated convolution-grouped normalization block ('G3 block').

3.2 Singer Models with BEGAN, G3 blocks and Frame-wise Noises (G3BEGAN)

The accompanied singers and solo singers have to take conditions as part of their input. One desirable property of the models is the ability to generate voices with arbitrary length, as the conditional signal can be of variable length. Besides, the model has to deal with the one-to-many issue mentioned in Section 1, and the absence of supervisory signals. With these issues in mind, we design a

GAN architecture for score and lyrics-free voice generation. In particular, we pay special attention to the following three components: 1) the network architecture, 2) the input noises for GAN, and 3) the loss function of the discriminator.

Let us first take a look at two existing GAN models for audio generation: (Engel et al., 2019) and (Donahue et al., 2019). Their generators and discriminators are both based on 2D convolutions, transposed 2D convolutions and dense (linear) layers. The generators take a vector $\mathbf{z} \in \mathbb{R}^U$ as the input noise and use transposed convolutions to expand $\mathbf{z}$ so that a temporal dimension emerges in the expanded intermediate matrices. The number of temporal frames in the final output depends on the total strides used in all the transposed convolutions. The discriminators take the output of the generators or the real signal as the input, and compress the input matrix with convolution layers until the output becomes a single value represents the prediction of true (real) or false (generated) data.

We can see that a main cause that prevents existing models from generating variable-length output is the need to expand $\mathbf{z}$ by transposed convolution layers. We remedy this by using an architecture consisting of the proposed G3 blocks, and convolutions without strides, for both the generators $G(\cdot)$ and discriminators $D(\cdot)$. Moreover, instead of using a single noise vector, our models take as input a sequence of noise vectors, denoted as $\mathbf{Z} \in \mathbb{R}^{U \times T}$, that has the same temporal length as the desired output $\mathbf{Y}$. Each column of $\mathbf{Z}$ is sampled indepedently from a Gaussian distribution $Normal(0, 1)$. At the first glance, it might feel unnatural to have one noise vector per frame as that may result in fast oscillations in the noises. However, we note that the output of $G(\cdot)$ for the t-th frame depends not only on the t-th column of $\mathbf{Z}$ (and $\mathbf{C}$ or $\mathbf{I}$), but the entire $\mathbf{Z}$ (and the condition matrices), because of the recurrent GRUs in the architecture. We expect that the GRUs in the discriminator $D(\cdot)$ would force $G(\cdot)$ to generate consistent consecutive frames. Therefore, the effect of the frame-wise noises might be introducing variations to the generation result (e.g., by adjusting the modes of the generated frame-wise features).

As for the loss function of $D(\cdot)$, our pilot experiments show that the strategy adopted by Engel et al. (2019) and Donahue et al. (2019), i.e., compressing the output of $G(\cdot)$ into a single true/false value, does not work well for our tasks. Our conjecture is that, as the output of $G(\cdot)$ is a sequence of vectors of variable length (rather than a fixed-size image), compressing the output of $G(\cdot)$ may have lost information important to the task.

We seek to remedy this by adopting the "auto-encoder style" discriminator loss proposed in boundary equilibrium GAN (BEGAN) (Berthelot et al., 2017). We adapt the objective of BEGAN, which is originally for generating images, for generating sequences. As an example, for the accompanied singer, the loss functions l_D and l_G for the discriminator and generator become respectively:

$$l_D = L(\mathbf{X}, \mathbf{C}) - \tau_s L(G(\mathbf{Z}, \mathbf{C}), \mathbf{C}) \,, \tag{1}$$

$$l_G = L(G(\mathbf{Z}, \mathbf{C}), \mathbf{C}) \,, \tag{2}$$

where $\mathbf{X} \in \mathbb{R}^{K \times T}$ is the feature sequence of a real vocal track sampled from the training data, $G(\mathbf{Z}, \mathbf{C}) \in \mathbb{R}^{K \times T}$ is the feature sequence for the generated vocal track, and $L(\cdot)$ is a function that measures how well the discriminator $D(\cdot)$, implemented as an auto-encoder, reconstructs its input:

$$L(\mathbf{M}, \mathbf{C}) = \frac{1}{WT} \sum_{w,t} |D(\mathbf{M}, \mathbf{C})_{w,t} - M_{w,t}| \,, \quad \text{for an arbitrary } W \times T \text{ matrix } \mathbf{M} \,, \tag{3}$$

where we use $M_{w,t}$ to denote the (w, t)-th element of a matrix $\mathbf{M}$ (and similarly for $D(\mathbf{M}, \mathbf{C})_{w,t}$). Moreover, the variable τ_s in Eq. (1) is introduced by BEGAN to balance the power of $D(\cdot)$ and $G(\cdot)$ during the learning process. It is dynamically set to be $\tau_{s+1} = \tau_s + \lambda(\gamma L(\mathbf{X}, \mathbf{C}) - L(G(\mathbf{Z}, \mathbf{C}), \mathbf{C}))$, for each training step s, with $\tau_s \in [0, 1]$. λ and γ are manually-set hyperparameters.

We can replace C in the above equations by an empty matrix for the case of training the free singer, and by I for the solo singer. For the accompanied singer, we note that $D(\cdot)$ has two ways to discriminate true and fake data: by examining whether $G(\mathbf{Z}, \mathbf{C})$ sounds like real singing voice, and by examining whether $G(\mathbf{Z}, \mathbf{C})$ matches the given condition $\mathbf{C}$.

3.3 Chord Progression Generator

We also use auto-regressive RNN to build a chord progression generator for implementing the solo singer, as mentioned in Section 2.3. Our chord generator is trained on the Wikifonia dataset,[2] a

[2] http://www.wikifonia.org/

collection of 6,670 songs in the leadsheet format (i.e., with separated symbolic melody and chord tracks). Its chord vocabulary covers 612 different chords. We set the harmonic rhythm of the chord generator such that a chord change may occur every beat. We desire the chord generator to freely generate chord progressions across different tempo values, time signatures, and keys. Moreover, the generated chord progression has to be rhythmically correct. In achieving so, we encode the tempo, time signatures, and key information (which are available in the Wikifonia dataset) as the initial hidden state of the RNN, and concatenate the chord vector from last time step with the beat position (e.g., 1, 2, 3, 4) of that time step as the input to the RNN. For data augmentation, we transpose the chord progressions found in Wikifonia to 24 possible keys.

Once trained, we use the chord generator to create chord progressions for testing the solo singer. With a one-hot encoding, each column of the condition matrix $\mathbf{I}$ would have dimension $J = 660$.

4 Experiments

4.1 Experimental Setup and Datasets

In our implementation, we use 80-dimensional mel-spectrograms as the acoustic features modeled and generated by the singer models (i.e., $K = 80$). We use the python package librosa (McFee et al., 2015), with default settings, to compute the mel-spectrograms from audio. A mel-spectrogram is passed to a WaveRNN vocoder (Kalchbrenner et al., 2018) to generate an audio signal from mel-spectrograms. Our implementation of the WaveRNN vocoder is based on the code from Fatchord.[3] Instead of using off-the-shelf pre-trained vocoders, which are typically trained for TTS, we train our vocoder from scratch with a set of 3,500 vocal tracks we separate (by a separation model) from an in-house collection of music that covers diverse musical genres.

One main difficulty in conducting this research is to get vocal tracks for training our singer models. Existing multitrack datasets that contain clean vocal tracks are usually diverse in musical genre and the singing timbre. As we will not inform our singer models with information regarding the identity of the singers, using such multitrack datasets cannot generate singing voices with consistent timbre. We therefore opt for collecting our own vocal tracks with limited timbre variations. In doing so, a source separation model is needed.

We implement our source separation model following the architecture proposed by Liu & Yang (2019), which represents the state-of-the-art as evaluated on the MUSDB benchmark (Rafii et al., 2017). Part of our training data for the source separation model is also from MUSDB. This dataset (and many other publicly available mutli-track music datasets) contains clean vocal and accompaniment tracks. However, for constructing the condition matrix $\mathbf{A}$ of our accompanied singer, we desire to have separated piano tracks as well (we will explain why shortly). We therefore collect 4.5 hours of clean piano playing audio on our own and use them to augment our training set for source separation. As a result, our source separation model can isolate out not only the vocal track but also the piano track from an arbitrary audio mixture.

We choose to focus on Jazz singers in this work and collect 17.4 hours of singing from female singers (with similar timbre) and 7.6 hours of singing from male singers (again with similar timbre). We use the aforementioned pre-trained separation model to get the vocal tracks. For batched training, we divide the tracks into 10-second sub-clips. Sub-clips that contain less than 40% vocals, as measured from energy, are removed. This leads to 9.9-hour and 5.0-hour training data for female and male Jazz vocals, respectively. 200 and 100 sub-clips are reserved from the training set as the validation set for female singing and male singing, respectively. Each model is trained for 500 epochs and the parameters of the epoch with the best BEGAN convergence are used for evaluation.

For the accompanied singer, we experiment with extracting pitch-related information from the accompaniment track to form the matrix $\mathbf{A}$ that conditions the generation of the vocal track. The assumption here is that whether the generated vocal track is in harmony with the accompaniment track can be largely determined by pitch-related information. For this purpose, we implement a piano transcription model to transcribe the separated piano track, leading to 88-dimensional transcribed frame-wise pitch as the accompaniment condition (i.e., $H = 88$, as there are 88 piano notes). We implement a piano transcription model with the G3 blocks introduced in Section 3.1, following the

[3] `https://github.com/fatchord/WaveRNN`

training procedure of (Hawthorne et al., 2018). The performance of our piano transcription model is slightly worse than that proposed by Hawthorne et al. (2018) (as evaluated on benchmark datasets), but according to our observation is strong enough for training accompanied singers.

Note that the clips in the training set do not necessarily contain piano playing. Even if a clip contains piano playing, the piano does not play across the entire clip. Therefore, the singer models have to learn to sing either with or without the accompaniment.

For performance evaluation, we collect 5.3 hours of Jazz music from Jamendo,[4] a platform for sharing copyright-free music. As mentioned in Section 1, this test set is meant to be public. We apply source separation to the audios, divide them into 20-second sub-clips,[5] and remove those that do not contain piano. This results in 402 20-second sub-clips for evaluation. 402 chord progressions are generated by the chord generator to evaluate the solo singer and compute the matchness.

Some more details of the G3BEGAN and the chord generator can be found in the appendix.

4.2 Objective Metrics and Objective Evaluation Result

The best way to evaluate the performance of the singer models we built is perhaps by listening to the generated results. As a result, we encourage readers of the paper to listen to the audio files provided in the supplementary material. However, objective evaluation remains desirable, either for model development or for gaining insights into the generation result. It is in particular important for a new task such as the ones tackled in this work.[6]

We propose to use the following objective metrics for our task.

- **Vocalness** measures whether a generated singing voice audio sounds like vocals. We use the singing voice detection tool proposed by Leglaive et al. (2015) and made available by Lee et al. (2018).[7] Given an audio signal, the tool outputs a value between 0 and 1 with 1 being most likely a vocal.
- **Average pitch**: We use the state-of-the-art monophonic pitch tracker CREPE (Kim et al., 2018)[8] to compute the pitch (in Hz) for each frame. The average pitch is computed by averaging the pitches of all the frames.
- **Singing-accompaniment matchness**: To objectively measure matchness, we build a melody harmonization RNN by adapting the chord generator described in Section 3.3. Given a pair of melody and chord sequences, the model can compute the likelihood of observing that chord sequence as the output when taking the melody sequence as the model input. We use the summation of the log likelihood across time frames as the matchness score. As the harmonization model considers symbolic sequences, we use CREPE (Kim et al., 2018) to transcribe the generated vocal, and Madmom (Böck et al., 2016) to recognize the chord sequence from the accompaniment track. Please see the appendix for details of this harmonization model.

The objective scores of our singer models are tabulated in Table 1. First we can see that all the singing voices generated by the models have high vocalness ranging from 84% to 95%. The average pitch of the generated female singing voices are higher than that of the generated male singing voices, and their values are also close to the average pitches of the singing voices from the female and male training data (311.6 and 263.3, respectively).

The accompanied singer achieves –9.25 in matchness, which are better than the random track baseline (–13.16) and the free singer baseline. From this comparison, we may say that the accompanied singers indeed learn to sing over the accompaniment to some degree.

[4]`https://www.jamendo.com`

[5]We note that this is longer than the 10-second sub-clips we used to train the singer models. This is okay as our model can generate variable-length output

[6]However, even in relatively well-studied music generation tasks such as performance generation, objective metrics that can replace subjective evaluation are yet to be developed (Chacón et al., 2018).

[7]`https://github.com/kyungyunlee/ismir2018-revisiting-svd/tree/master/leglaive_lstm`

[8]`https://github.com/marl/crepe`

Table 1: Result of objective evaluation for our singer models. *The matchness scores of the free singers are computed by pairing them with the 402 test clips.

Model	**Average pitch (Hz)**	**Vocalness**	**Matchness**
Free singer (female)	288 ± 28	0.84 ± 0.10	–13.28 ± 3.80*
Accompanied singer (female)	313 ± 18	0.91 ± 0.08	–9.25 ± 3.13
Solo singer (female)	302 ± 17	0.93 ± 0.06	–9.30 ± 3.11
Free singer (male)	248 ± 39	0.95 ± 0.10	–13.29 ± 3.19*
Accompanied singer (male)	207 ± 14	0.90 ± 0.09	–9.31 ± 3.16
Solo singer (male)	213 ± 14	0.90 ± 0.07	–9.30 ± 3.13

Table 2: Mean opinion scores (and standard deviations) with respect to four evaluation criteria collected from the user study, for three different versions of accompanied singer (female). The scores are in 5-point Likert scale and are from 1 to 5; the higher the better.

Model (epochs trained)	**Sound quality**	**Vocalness**	**Expression**	**Matchness**
G3BEGAN (20 epochs)	1.59 ± 0.82	1.93 ± 0.99	1.98 ± 0.88	2.18 ± 1.08
G3BEGAN (240 epochs)	2.24 ± 0.93	2.66 ± 1.01	2.60 ± 1.01	2.58 ± 1.05
G3BEGAN (final)	2.38 ± 0.96	2.98 ± 1.02	2.85 ± 1.00	2.74 ± 1.04

Examples of the generated spectrograms of our models can be found in the appendix. From visually inspecting the spectrograms and listening to the result, it seems the female singer models learn better than the male singer models.

4.3 User Study and Subjective Evaluation Result

We conduct a user study to subjectively evaluate the accompanied singer, the female one. We intend to compare the 'final model (with number of epochs selected according to a validation set) against two early versions of the model trained with less epochs. An online, non-paid questionnaire is established, inviting people to rate the generated singing voices of the three versions of the female accompanied singer for three different accompaniment tracks (each 20 seconds), one accompaniment track per page. The subjects are informed the purpose of our research (i.e., score and lyrics-free singing voice generation) and the user study (to compare three computer models), and are asked to listen to the generated vocals with the accompanied track in a quiet environment with proper microphone volume. No post-processing (e.g., noise removal, EQ adjustment) is applied to the audio files. The ordering of the result of the three models is randomized.

Table 2 shows the average ratings from 39 participants. We can see from the table that 1) the model indeed learns better with more epochs, 2) the final model performs relatively worse with respect to sound quality, but works fine for the other three criteria. We consider these as promising result, given the challenges associated with the task. Yet, certainly there is room for future improv

5 Related work

Singing voice synthesis is often considered as a variant of TTS as both require the conversion from a text stream to an audio stream. While early approaches are mainly based on digital signal processing techniques (Cook, 1996; Bonada & Serra, 2007; Umbert et al., 2015), machine learning approaches offer greater flexibility and have been shown to generate audio with high fidelity in recent years. Nishimura et al. (2016) firstly demonstrated singing voice synthesis with improved naturalness by replacing hidden Markov models with deep neural nets. Many neural network models for singing voice synthesis have been proposed since then (e.g., (Lee et al., 2019)). However, to our best knowledge, the case of score and lyrics-free singing voice generation has not been tackled before.

Using neural nets for score-conditioned instrumental audio generation have also been investigated in recent years. However, existing work is mostly concerned with the generation of single notes of, for example, 4-second long (Défossez et al., 2018; Donahue et al., 2019; Engel et al., 2019). A deep neural network that is capable of generating variable-length audio (e.g., a "recurrent generator") as the proposed singer models do, to our knowledge, has not been much studied.

6 Conclusion

In this paper, we have introduced a novel task of singing voice generation that does not use musical scores and lyrics. Specifically, we proposed three singing schemes with different input conditions: free singer, accompanied singer, and solo singer. We have also proposed a BEGAN based architecture that uses GRUs and grouped dilated convolutions to learn to generate singing voices in an adversarial way. For evaluating such models, we proposed several objective metrics and implemented a model to measure the compatibility between a given accompaniment track and the generated vocal track. The evaluation shows that the audio quality of the generated voices still leave much room for improvement, but in terms of humanness and emotion expression our models work fine. We consider this as promising result since our approach is fairly anti-anthropocentric in that the models are trained from scratch with little knowledge of human language.

Score and lyrics-free singing voice generation is a new task, and this work represents only a first step tackling it. There are many interesting ideas to pursue. For example, we have chosen to extract pitch-related information only from the accompaniment track for the accompanied singer, but a more interesting way is to let the model learns to extract relevant information itself. Improving the source separation model by artifact removal, or pre-training our singer models on manually-separated tracks might help improve the quality of the generated sounds. In the near future, we plan to experiment with other network architectures (e.g., progressive GAN (Karras et al., 2018; Engel et al., 2019) or mutli-band design (Wang & Yang, 2019)), and to investigate advanced settings that allow for timbre and expression control. We also like to investigate using the generated sounds in various genres to create musical artworks, such as Jazz Hiphop.

References

David Berthelot, Thomas Schumm, and Luke Metz. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv preprint*, 2017.

Merlijn Blaauw, Jordi Bonada, and Ryunosuke Daido. Data efficient voice cloning for neural singing synthesis. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6840–6844, 2019.

Sebastian Böck, Filip Korzeniowski, Jan Schlüter, Florian Krebs, and Gerhard Widmer. madmom: a new python audio and music signal processing library. *CoRR*, abs/1605.07008, 2016. URL `http://arxiv.org/abs/1605.07008`.

Jordi Bonada and Xavier Serra. Synthesis of the singing voice by performance sampling and spectral models. *IEEE Signal Processing Magazine*, 24(2):67–79, 2007.

Estefanía Cano, Judith Liebetrau, Derry Fitzgerald, and Karlheinz Brandenburg. The dimensions of perceptual quality of sound source separation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 601–605, 2018.

Carlos Eduardo Cancino Chacón, Maarten Grachten, Werner Goebl, and Gerhard Widmer. Computational models of expressive music performance: A comprehensive and critical review. *Frontiers in Digital Humanities*, 2018, 2018.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In *Proc. Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *Proc. NIPS Workshop on Deep Learning*, 2014.

Perry R. Cook. Singing voice synthesis: History, current work, and future directions. *Computer Music Journal*, 20(3):38–46, 1996.

Alexandre Défossez, Neil Zeghidour, Nicolas Usunier, Léon Bottou, and Francis Bach. SING: Symbol-to-instrument neural generator. In *Proc. Advances in Neural Information Processing Systems*, pp. 9041–9051, 2018.

Chris Donahue, Julian McAuley, and Miller Puckette. Adversarial audio synthesis. In *Proc. International Conference on Learning Representations*, 2019.

Walter Jay Dowling. Development of musical schemata in children's spontaneous singing. *Advances in Psychology*, 19:145–163, jan 1984.

Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. GANsynth: Adversarial neural audio synthesis. In *Proc. International Conference on Learning Representations*, 2019.

Andrew Gibiansky, Sercan Arik, Gregory Diamos, John Miller, Kainan Peng, Wei Ping, Jonathan Raiman, and Yanqi Zhou. Deep Voice 2: Multi-speaker neural text-to-speech. In *Proc. Advances in Neural Information Processing Systems 30*, pp. 2962–2970, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proc. Advances in Neural Information Processing Systems*, pp. 2672–2680. 2014.

Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription. In *Proc. International Society for Music Information Retrieval Conference*, pp. 50–57, 2018.

Yukiya Hono, Shumma Murata, Kazuhiro Nakamura, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda. Recent development of the DNN-based singing voice synthesis systemSinsy. In *Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pp. 1003–1009, 2018.

Yukiya Hono, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda. Singing voice synthesis based on generative adversarial networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6955–6959, 2019.

Kantapon Kaewtip, Fernando Villavicencio, Fang-Yu Kuo, Mark Harvilla, Iris Ouyang, and Pierre Lanchantin. Enhanced virtual singers generation by incorporating singing dynamics to personalized text-to-speech-to-singing. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6960–6964, 2019.

Nal Kalchbrenner, Erich Elsen, Karen Simonyan, Seb Noury, Norman Casagrande, Edward Lockhart, Florian Stimberg, Aaron van den Oord, Sander Dieleman, and Koray Kavukcuoglu. Efficient neural audio synthesis. In *Proc. International Conference on Machine Learning*, pp. 2410–2419, 2018.

Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *Proc. International Conference on Learning Representations*, 2018.

Jong Wook Kim, Justin Salamon, Peter Li, and Juan Pablo Bello. CREPE: A convolutional representation for pitch estimation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 161–165, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Juheon Lee, Hyeong-Seok Choi, Chang-Bin Jeon, Junghyun Koo, and Kyogu Lee. Adversarially trained end-to-end Korean singing voice synthesis system. In *Proc. INTERSPEECH*, 2019.

Kyungyun Lee, Keunwoo Choi, and Juhan Nam. Revisiting singing voice detection: a quantitative review and the future outlook. In *Proc. International Society for Music Information Retrieval Conference*, 2018.

Simon Leglaive, Romain Hennequin, and Roland Badeau. Singing voice detection with deep recurrent neural networks. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 121–125, 2015.

Hyungui Lim, Seungyeon Rhyu, and Kyogu Lee. Chord generation from symbolic melody using BLSTM networks. In *Proc. International Society for Music Information Retrieval Conference*, pp. 621–627, 2017.

Jen-Yu Liu and Yi-Hsuan Yang. Dilated convolution with dilated GRU for music source separation. In *Proc. International Joint Conference on Artificial Intelligence*, pp. 4718–4724, 2019.

Andrés Marafioti, Nicki Holighaus, Nathanaël Perraudin, and Piotr Majdak. Adversarial generation of time-frequency features with application in audio synthesis. In *Proc. International Conference on Machine Learning*, 2019.

Brian McFee, Colin Raffel, Dawen Liang, Daniel P. W . Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. librosa: Audio and music signal analysis in Python. In *Proc. Python in Science Conf.*, pp. 18–25, 2015. [Online] https://librosa.github.io/librosa/.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Masanari Nishimura, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda. Singing voice synthesis based on deep neural networks. In *Proc. INTERSPEECH*, pp. 2478–2482, 2016.

Ryan Prenger, Rafael Valle, and Bryan Catanzaro. WaveGlow: A flow-based generative network for speech synthesis. *arXiv preprint arXiv:1811.00002*, 2018.

Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner. The MUSDB18 corpus for music separation, 2017.

Zafar Rafii, Antoine Liutkus, Fabian-Robert Stter, Stylianos Ioannis Mimilakis, Derry FitzGerald, and Bryan Pardo. An overview of lead and accompaniment separation in music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(8):1307–1335, 2018.

Tim Salimans and Diederik P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. *Proc. Advances in Neural Information Processing Systems*, pp. 901–909, 2016.

Tom Sercu and Vaibhava Goel. Dense prediction on sequences with time-dilated convolutions for speech recognition. *arXiv preprint arXiv:1611.09288*, 2016.

Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, R. J. Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4779–4783, 2017.

Hiroki Tamaru, Yuki Saito, Shinnosuke Takamichi, Tomoki Koriyama, and Hiroshi Saruwatari. Generative moment matching network-based random modulation post-filter for DNN-based singing voice synthesis and neural double-tracking. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7070–7074, 2019.

Martí Umbert, Jordi Bonada, Masataka Goto, Tomoyasu Nakano, and Johan Sundberg. Expression control in singing voice synthesis: Features, approaches, evaluation, and challenges. *IEEE Signal Processing Magazine*, 32(6):55–73, 2015.

	Details	Input dim	Output dim
Input	1DConv (kernel=3, dilation=1)		512
	Group normalization (group=4)		
	Leaky ReLU (0.01)		
G3 Block 1	GRU	512	512
	Grouped 1DConv (kernel=3, dilation=2, group=4)	512	512
	Group normalization (group=4)		
	Leaky ReLU (0.01)		
G3 Block 2	GRU	512	512
	Grouped 1DConv (kernel=3, dilation=2, group=4)	512	512
	Group normalization (group=4)		
	Leaky ReLU (0.01)		
Output	1DConv (kernel=3, dilation=1)	512	80

Table 3: Network architecture of the generator of the proposed G3BEGAN model. It uses two G3 blocks introduced in Section 3.1. For the noise input, we set the dimension U to 20.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. WaveNet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Elliot Waite, Douglas Eck, Adam Roberts, and Dan Abolafia. Project Magenta: Generating long-term structure in songs and stories, 2016. `https://magenta.tensorflow.org/blog/2016/07/15/lookback-rnn-attention-rnn/`.

Bryan Wang and Yi-Hsuan Yang. PerformanceNet: Score-to-audio music generation with multi-band convolutional residual network. In *Proc. AAAI Conference on Artificial Intelligence*, 2019.

Yuxin Wu and Kaiming He. Group normalization. *arXiv preprint arXiv:1803.08494*, 2018.

A Appendix: Training setting and model details

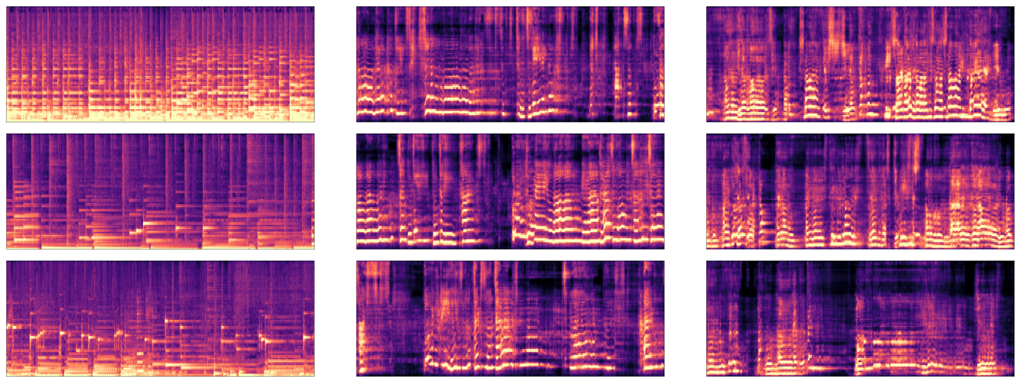
A.1 G3BEGAN

Singer models with G3BEGAN are trained with Adam (Kingma & Ba, 2014) with 0.0001 learning rate and mini-batch size being 5. Each model is trained with 500 epochs. Gradient norm clipping with magnitude 3 is used.

The generator in G3BEGAN is implemented with a stack of two G3 blocks. Please see Table 3 for details of the network architecture.

A.2 Chord Generator

The chord generator is aimed to generate chord progressions freely under some given conditions. It supports 12 major and 12 minor keys, 10 tempo options from 60 to 240 BPM, 6 time signature options, and 51 chord qualities (612 chords in total). The conditions, key, tempo, and time signatures, are encoded into one-hot representation and concatenated together as a 40-dimension vector. The model mainly consists with 3 stacked GRU layers, each with 512 hidden variables. The input of each time step is a 524-dimensional vector consisting of a chord embedding and a beat-related one-hot positional encoding (to encourage the model to follow certain rhythmical pattern. This input array passes through a fully-connected layer to 512-dimension and is used as the input of the GRUs. The training data are the leadsheets from the Wikifonia dataset. We augmented the data by rotating the keys, leading to in total 80,040 leadsheets for training.



(a) The given instrumental, accompaniment track

(b) The voices generated by the accompanied singer (female)

(c) The voices generated by the accompanied singer (male)

Figure 3: Samples of spectrograms generated by our accompanied singers: (left) the given accompaniment tracks, the voices generated by (middle) the female singer and (right) the male singer.

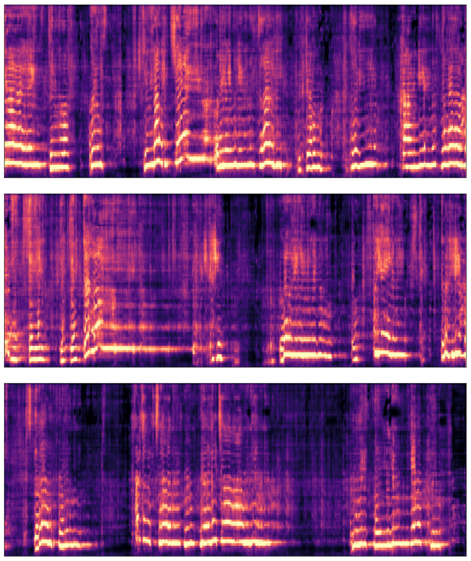
A.3 Melody Harmonization Model

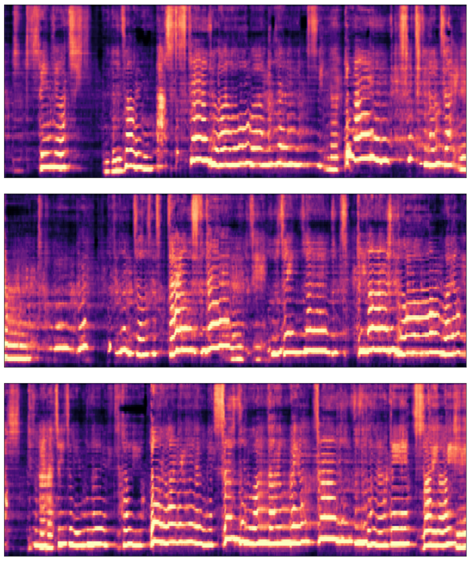
The melody harmonization model is modified from the chord generator described in Appendix A.2, using additionally the melody tracks found in the Wikifonia dataset. Specifically, the model intends to generate a chord sequence given a melody sequence. Such a model can be learned by using the pairs of melody and chord tracks in Wikifonia. We add the chroma representation of the melody with window size of a quarter-note to the input vector.

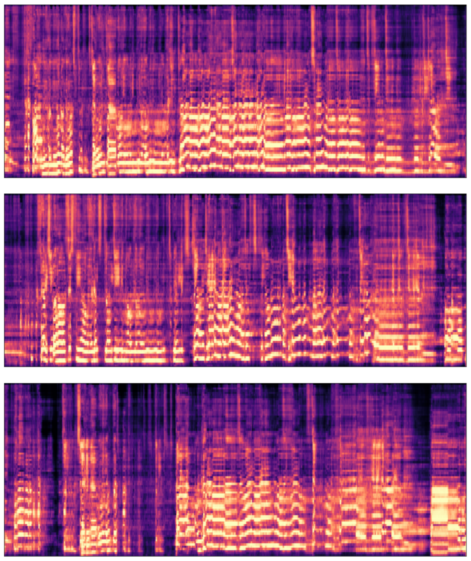
The matchness of the real pairs of vocal and accompaniment tracks in Wikifonia is –7.04±2.91. If we pair a vocal track with a randomly selected accompaniment track, the score becomes –13.16±3.72.

B Appendix: Examples of the Spectrogram of the Generated Singing Voices

Examples of the spectrograms of the generated singing voices can be found in Figures 3 and 4.


(a) Free singer (female)


(b) Solo singer (female)


(c) Free singer (male)

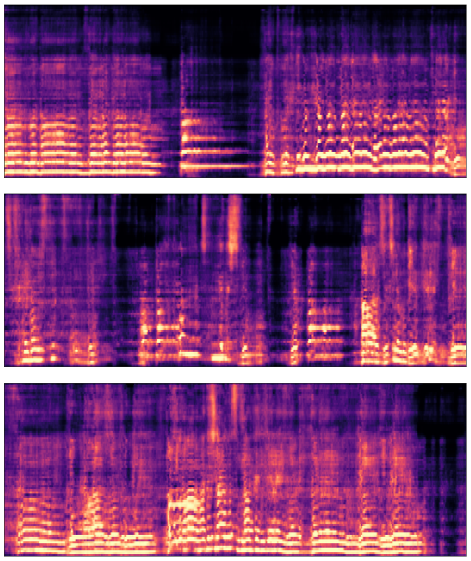

(d) Solo singer (male)

Figure 4: Samples of spectrograms generated by our (left) free singers and (right) solo singers. We can see salient pitch contour in the spectrograms. Moreover, the pitches sung by the male singers seem on average lower than those sung by the female singers.