CHAPTER V

LEARNING TIMBRE SPACE FOR MUSIC SYNTHESIS

In this chapter, we consider the problem of music synthesis. Although seemingly unconnected to music transcription, studying music synthesis models can provide an interesting perspective in understanding the music transcription problem: viewing synthesis as an inverse problem of analysis. One important outcome of this approach is that it allows us to computationally model the variations of musical timbre, a perceptual quality of sound that is less rigorously defined compared to other notions such as pitch and loudness. In light of this, we develop in this chapter a music synthesis model that can be used not only for synthesizing music, but also for better understanding the relation between audio signals and the qualities of musical sounds such as timbre and pitch.

The recent success of raw audio waveform synthesis models like WaveNet motivates a new approach for music synthesis, in which the entire process — creating audio samples from a score and instrument information — is modeled using generative neural networks. In this chapter, we develop a neural music synthesis model, which consists of a recurrent neural network conditioned on a learned instrument embedding followed by a WaveNet vocoder. The synthesis model is capable of learning a timbre embedding space that successfully captures the diverse variations in timbres within a large dataset. The model can therefore seamlessly connect the note sequence input and the timbre information to the audio signal, serving as the synthesizer component in Figure 2. The content of

this chapter is largely based on the work presented at IEEE ICASSP 2019 (J. W. Kim et al., 2019).

1 Introduction

Musical synthesis, most commonly, is the process of generating musical audio with given control parameters such as instrument type and note sequences over time. The primary difference between synthesis engines is the way in which *timbre* is modeled and controlled. In general, it is difficult to design a synthesizer that both has dynamic and intuitive timbre control and is able to span a wide range of timbres; most synthesizers change timbres by having presets for different instrument classes or have a very limited space of timbre transformations available for a single instrument type.

In this chapter, we present a flexible music synthesizer named Mel2Mel, which uses a learned, non-linear instrument embedding as timbre control parameters in conjunction with a learned synthesis engine based on WaveNet. Because the model has to learn the timbre – any information not specified in the note sequence – to successfully reconstruct the audio, the embedding space spans over the various aspects of timbre such as spectral and temporal envelopes of notes. This learned synthesis engine allows for flexible timbre control, and in particular, timbre morphing between instruments, as demonstrated in our interactive web demo.

2 Background

2.1 Timbre Control in Musical Synthesis

Methods for music synthesis are based on a variety of techniques such as FM synthesis, subtractive synthesis, physical modeling, sample-based synthesis, and granular synthesis (Pejrolo & Metcalfe, 2017). The method of controlling timbre and the level of flexibility depends on the parameters of the exact method used, but in general, there is a trade-off between flexible timbre control over synthetic sounds (e.g. FM or subtractive synthesis) and a limited timbre control in more "realistic" sounds (e.g. sample-based, physical model-based, or granular synthesis). Our work is aimed at achieving the best of both worlds: flexibly controlling a variety of realistic-sounding timbres.

2.2 Timbre Morphing

'Morphing' of a sound can be generally described as making a perceptually gradual transition between two or more sounds (Caetano & Rodet, 2010). A common approach is to use a synthesis model and define sound morphing as a numerical interpolation of the model parameters. Sinusoidal models can directly interpolate between the energy proportions of the partials (Osaka, 1995; Boccardi & Drioli, 2001). Other models use parameters characterizing the spectral envelope (Slaney et al., 1996; Ezzat et al., 2005) or psychoacoustic features for perceptually linear transition (Caetano & Rodet, 2013). A limitation of these approaches is that morphing can only be applied among the range of timbres covered by a certain synthesis model, whose expressiveness or parameter set may be limited. To

overcome this, we employ a data-driven approach for music synthesis that is generalizable to all timbres in the dataset.

2.3 Timbre Spaces and Embeddings

Timbre is often modeled using a timbre space (Peeters et al., 2011), in which similar timbres lie closer than dissimilar timbres. In early work in psychoacoustics, multidimensional scaling (MDS) was used to obtain a timbre space which preserves the timbral dissimilarities measured in perceptual experiments (Grey, 1977; Wessel, 1979). Meanwhile, in music content analysis, timbre similarity is measured using computed features such as the Mel-frequency cepstral coefficients (MFCCs) (Logan, 2000), descriptors of the spectral envelope (A. Agostini & Ghisi, 2013), or hidden-layer weights of a neural network trained to distinguish different timbres (Humphrey et al., 2011). A recent method (Esling et al., 2018) used a variational autoencoder (Kingma & Welling, 2014) to obtain a timbre space, and unlike the above embeddings, the method is able to generate monophonic audio for a particular timbre embedding but does not consider the temporal evolution of notes such as attacks and decays. In our work, we generate a timbre embedding as a byproduct of polyphonic synthesis, which can utilize both spectral and temporal aspects of timbre.

2.4 Neural Audio Synthesis using WaveNet

WaveNet (van den Oord, Dieleman, et al., 2016) is a generative audio synthesis model that is able to produce realistic human speech. WaveNet achieves this by learning an autoregressive distribution which predicts the next audio sample from the previous samples in its receptive field using a series of dilated convolutions.

Tacotron (Shen et al., 2018) and Deep Voice (Ping et al., 2018) are WaveNet-based text-to-speech models which first predict a Mel spectrogram from text and use it to condition a WaveNet vocoder.

There are also a few notable applications of WaveNet in music, including NSynth (Engel et al., 2017), an autoencoder architecture which separately encodes monophonic pitch with learned timbral features, and the universal music translation network (Mor et al., 2019) which uses a denoising autoencoder architecture that can extract and translate between musical styles while preserving the melody. In Hawthorne et al. (2019), a WaveNet is used for music synthesis conditioned directly on note sequences but only supports piano sounds. Our model is similarly built around WaveNet for its synthesis capability but uses a learned embedding space to flexibly control the timbre of polyphonic music.

3 Method

The neural network shown in Figure 14, dubbed Mel2Mel, concerns the task of synthesizing music corresponding to given note sequences and timbre. The note sequences are supplied as a MIDI file and converted to a piano roll representation, which contains the note timings and the corresponding note velocities for each of the 88 piano keys. We use a fixed step size in time, and the piano roll representation is encoded as a matrix by quantizing the note timings to the nearest time step. The input to the neural network is a concatenation of two 88-dimensional piano roll representations, one for onsets and one for frames, comprising 176 dimensions in total:

$$
\begin{aligned}
\mathbf{X} &= [\,\mathbf{X}^{\text{onset}}\,;\,\mathbf{X}^{\text{frame}}\,] \\
\mathbf{X}^{\text{onset}}_{p,t} &= v_{\text{the active note}} \cdot \mathbb{1}_{\text{a note at pitch } p \text{ is } \textit{first} \text{ active at time } t} \\
\mathbf{X}^{\text{frame}}_{p,t} &= v_{\text{the active note}} \cdot \mathbb{1}_{\text{a note at pitch } p \text{ is active at time } t}
\end{aligned}
$$

where $\mathbb{1}$ is the indicator function, and v denotes the MIDI velocity scaled to $[0, 1]$. This input representation is inspired by (Hawthorne et al., 2018) which showed that jointly training on onsets and frames performs better than using frame information only; similarly, we want the network to maximally utilize the onsets which have the most relevant information on the attack sounds, while still receiving the frame information. Another reason for using both onsets and frames is that, because of the time quantization, repeated notes become indistinguishable only using $\mathbf{X}^{\text{frame}}$ when an offset is too close to the subsequent onset.

The input goes through a linear 1x1 convolution layer, which is essentially a time-distributed fully connected layer, followed by a FiLM layer, to be described in the following subsection, which takes the timbre embedding vector and transforms the features accordingly. After a bidirectional LSTM layer and another FiLM layer for timbre conditioning, another linear 1x1 convolution layer produces the Mel spectrogram prediction. The resulting Mel spectrogram is then fed to a separately trained WaveNet vocoder to produce the music; Mel spectrograms compactly convey sufficient information for audio synthesis and have been successfully used for conditioning WaveNet (Shen et al., 2018; Ping et al., 2018). The use of bidirectional LSTM is appropriate because Mel spectrograms are constructed using a larger window than the step size, making it non-causal. The only nonlinearities

in the network are in the LSTM, and there are no time-domain convolutions except in WaveNet.

3.1 Timbre Conditioning using FiLM Layers

A FiLM layer (Perez et al., 2017) is a neural network that can take side information; it first learns functions f and h, which are linear layers mapping the timbre embedding $\mathbf{t}$ to $\gamma = f(\mathbf{t})$ and $\beta = h(\mathbf{t})$. The affine transformation, or FiLM-ing, of intermediate-layer features $\mathbf{F}$ is then applied using feature-wise operations:

$$\text{FiLM}(\mathbf{F} \mid \gamma, \beta) = \gamma\mathbf{F} + \beta = f(\mathbf{t})\mathbf{F} + h(\mathbf{t}). \tag{19}$$

We can think of the model architecture as a multi-step process that shapes a piano roll into its Mel spectrogram, by applying appropriate timbre given as side information. A FiLM layer is a suitable choice for this task, because it can represent such action of shaping using an affine transformation of intermediate-layer features. For each instrument to model, its timbre is represented in an embedding vector $\mathbf{t}$, implemented as a learned matrix multiplication on one-hot encoded instrument labels. The values of instrument embedding vectors do not vary in time and are learned jointly with the rest of the model.

At a higher level, the affine transformations learned by the FiLM layers are nonlinearly transformed by the recurrent and convolutional layers to respectively form temporal and spectral envelopes, which are two important aspects that characterize instrumental timbre. Using the first FiLM layer is essential because the recurrent layer needs to take timbre-dependent input to apply the temporal dynamics according to the timbre, and the second FiLM layer can apply additional spectral envelope on the recurrent layer's output.

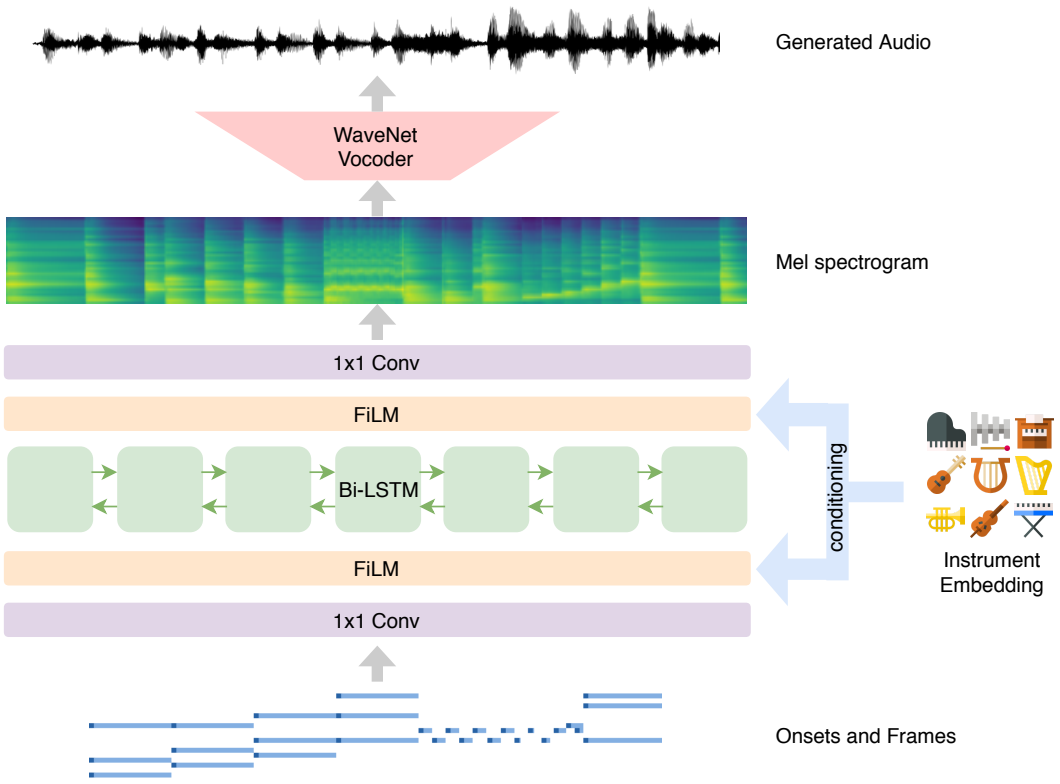

Figure 14: The overall architecture of the proposed Mel2Mel model. The note sequences and instrument embeddings are combined to predict the Mel spectrogram, which is then fed to the WaveNet vocoder.

3.2 Model Details

The sampling rate of 16 kHz and μ-law encoding with 256 quantization levels are used for all audio as in the original WaveNet paper (van den Oord, Dieleman, et al., 2016). The predicted Mel spectrograms are defined using 80 area-normalized triangular filters distributed evenly between zero and the Nyquist frequency in the Mel scale. The STFT window length of 1,024 samples and the step size of 128 samples are used, which translate to 64 milliseconds and 8 milliseconds, respectively. Unless specified otherwise, we use 256 channels in all hidden layers and a two-dimensional embedding space for timbre conditioning.

For Mel spectrogram prediction, an Adam optimizer with the initial learning

rate of 0.002 is used, and the learning rate is halved every 40,000 iterations. The model is trained for 100,000 iterations, where each iteration takes a mini-batch of 128 sequences of length 65,536, or 4.096 seconds. Three different loss functions are used and compared; for linear-scale Mel spectrograms S_{true} and S_{pred}:

$$\textbf{abs MSE} = \mathbb{E}\left[\left(S_{\text{true}} - S_{\text{pred}}\right)^2\right] \tag{20}$$

$$\textbf{log-abs MSE} = \mathbb{E}\left[\left(\log S_{\text{true}} - \log S_{\text{pred}}\right)^2\right] \tag{21}$$

$$\textbf{tanh-log-abs MSE} = \mathbb{E}\left[\left(\tanh \tfrac{1}{4} \log S_{\text{true}} - \tanh \tfrac{1}{4} \log S_{\text{pred}}\right)^2\right] \tag{22}$$

All logarithms above are natural, and the spectrogram magnitudes are clipped at -100 dB. Prepending tanh gives a soft-thresholding effect where the errors in the low-energy ranges are penalized less than the errors close to 0 dB.

For the WaveNet vocoder, we used nv-wavenet[1], a real-time open-source implementation of autoregressive WaveNet by NVIDIA. This implementation limits the recurrent channel size at 64 and the skip channels at 256, because of the GPU memory capacity. A 20-layer WaveNet model was trained with the maximum dilation of 512, and the Mel spectrogram input is upsampled using two transposed convolution layers of window sizes 16 and 32 with strides of 8 and 16, respectively. An Adam optimizer with the initial learning rate of 0.001 is used, and the learning rate is halved every 100,000 iterations, for one million iterations in total. Each iteration takes a mini-batch of 4 sequences of length 16,384, i.e. 1.024 seconds.

[1] `https://github.com/NVIDIA/nv-wavenet`

4 Experiments

4.1 Datasets

While it is ideal to use recorded audio of real instruments as the training dataset, the largest multi-instrument polyphonic datasets available such as MusicNet (Thickstun et al., 2017) is highly skewed, contains a limited variety of solo instrument recordings, and is expected to have a certain degree of labeling errors as the authors suggest. Instead, we used synthesized audio for training and collected MIDI files from `www.piano-midi.de`, which are also used in the MAPS Database (Emiya et al., 2010); these MIDI files are recorded from actual performances and contain expressive timing and velocity information. We have selected 10 General MIDI instruments shown in Figure 16 covering a wide variety of timbres, and 334 piano tracks are synthesized for each instrument using FluidSynth with the default SoundFont from MuseScore 3. The 334 tracks are randomly split into 320 for training and 14 for validation. The total size of the synthesized dataset is 3,340 tracks and 221 hours.

For later experiments, we also generate a similar dataset using 100 manually selected instrument classes using a high-quality collection of SoundFonts, which contains a wide variety of timbres.

4.2 Ablation Study on Model Design

In this series of experiments, we examine how slight variations in the model architecture affect the performance and show that the proposed model achieves the best performance in accurately predicting Mel spectrograms. The first two variations use either the frame data or the onset data only as the input. The

Variations	Train loss ($\times 10^3$)	Validation loss ($\times 10^3$)
Proposed	4.09 ± 0.30	$\mathbf{4.75 \pm 0.05}$
Frame input only	5.58 ± 0.32	5.92 ± 0.08
Onset input only	5.88 ± 0.37	6.97 ± 0.06
First FiLM only	4.55 ± 0.32	4.99 ± 0.06
Second FiLM only	7.65 ± 0.34	8.76 ± 0.08
Forward LSTM only	5.70 ± 0.43	5.56 ± 0.09
ReLU activation	3.97 ± 0.35	5.04 ± 0.06
3x1 convolutions	3.66 ± 0.28	5.12 ± 0.08
5x1 convolutions	3.49 ± 0.30	5.06 ± 0.08
2-layer LSTM	2.98 ± 0.20	4.96 ± 0.12

Table 3: Train and validation losses as defined in Equation 22 with respect to different variations on the architecture, either limiting or increasing the model capacity. The proposed architecture has the lowest validation loss, indicating that it predicts the most accurate Mel spectrograms while not being prone to overfitting.

next three omit an architectural component: one of the two FiLM layers or the backward LSTM. The last four increase the network's capacity by adding the ReLU nonlinearity after the first convolution, using kernel sizes of 3 or 5 time steps in convolutions, or adding another LSTM layer. The train and validation losses as defined in Equation 22 are shown[1] in the table above for each variation. Using both onsets and frames is indeed more effective than using only one of them in the input. The first FiLM layer plays a more crucial role than the second, because only the first can help learn a timbre-dependent recurrent layer. As expected, removing the backward LSTM also hurt the performance.

On the other hand, any variations increasing the model capacity make the model overfit and fail to generalize to validation data. This implies that the proposed model has the optimal architecture among the tested variations, and

[1] The means and standard deviations over the model checkpoints in 90k-100k iterations are reported, to minimize the variability due to SGD.

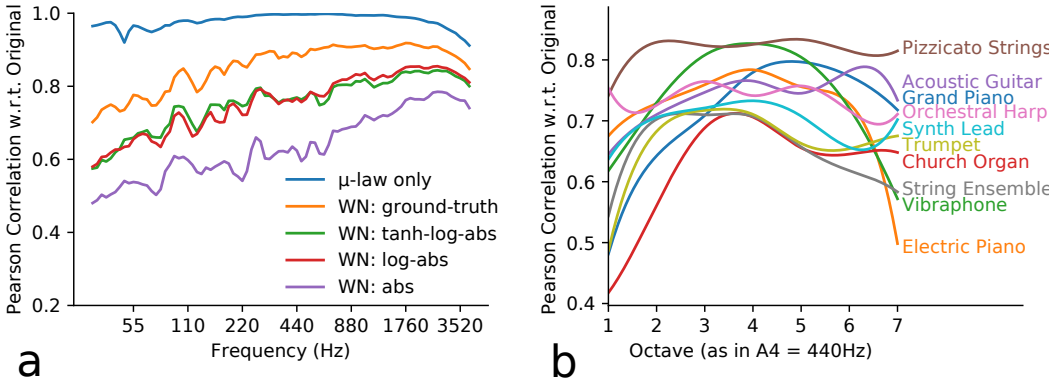

Figure 15: Pearson correlations between the reconstructed and original audio. (a) each stage of degradation. (b) per-instrument breakdown of the green curve on the left. The curves are smoothed and drawn with respect to each octave for readability.

more specifically, having the nonlinearity only in the single recurrent layer helps the model better generalize in predicting Mel spectrograms from unseen note sequences. A possible interpretation is that the increased capacity is being used for memorizing the note sequences in the training dataset, as opposed to learning to model the timbral features independent of specific notes.

4.3 Synthesis Quality

4.3.1 Numerical Analysis of Audio Degradation

The model goes through several stages of prediction, and each stage incurs a degradation of audio quality. There necessarily exists some degradation caused by the μ-law quantization, and WaveNet adds additional degradation due to its limited model capacity. The generated audio is further degraded when imperfect Mel spectrogram predictions are given. As an objective measure of audio quality degradation at each stage and for each instrument, we plot the Pearson correlations between the synthesized and original audio in Figure 15. To calculate and visualize the correlations with respect to evenly spaced octaves, we use 84-bin log-magnitude

constant-Q transforms with 12 bins per octave starting from C1 (≈ 32.70Hz) and 512-sample steps. For ideal synthesis, the Pearson correlation should be close to 1, and lower correlations indicate larger degradation from the original.

Figure 15a shows the correlations for each stage of degradation and for different loss functions used for training the model. The degradations are more severe in low frequencies in general, where the WaveNet model sees fewer periods of a note within its fixed receptive field length. The orange curve showing the correlations for WaveNet synthesis using ground-truth Mel spectrograms already exhibits a significant drop from the top curve; this defines an upper bound of Mel2Mel's synthesis quality. The lower three curves correspond to the loss functions in Equations 20-22, among which the abs MSE loss clearly performs the worse than the other two which have almost identical Pearson correlation curve, indicating that the MSE loss is more effective in the log-magnitude scale.

Figure 15b shows the breakdown of the curve corresponding to Equation 22 into each of the 10 instruments. There are rather drastic differences among instruments, and most instruments have low Pearson correlations in low pitches except pizzicato strings. The implications of these trends are discussed in (J. W. Kim et al., 2019), in comparison with the subjective audio quality test.

4.4 The Timbre Embedding Space

To make sense of how the learned embedding space conveys timbre information, we construct a 320-by-320 grid that encloses all instrument embeddings and predict the Mel spectrogram conditioned on every pixel in the grid. The spectral centroid and the mean energy corresponding to each pixel are plotted in Figure 16, which are indicative of the two main aspects of instrumental timbres: the spectral and

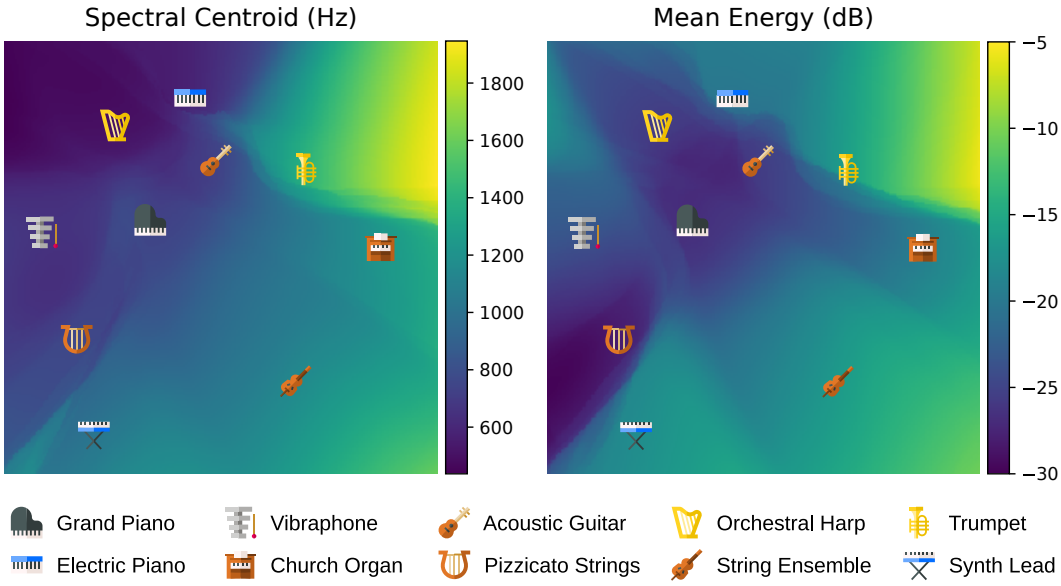

Figure 16: Visualization of the embedding space trained in 2 dimensions, using spectral centroids and mean energy. The continuous colors are obtained for each pixel in the 320-by-320 grid and are related to the spectral and temporal envelopes of the timbres.

temporal envelopes. For a fixed f0, a higher spectral centroid signifies stronger harmonic partials in high frequency, while a lower spectral centroid indicates that it is closer to a pure sine tone. Similarly, higher mean energy implies a more sustained tone, and low mean energy means that the note is transient and decays rather quickly. The points corresponding to the 10 instruments are annotated with instrument icons.[1] These plots show that the learned embedding space forms a continuous span over the timbres expressed by all instruments in the training data. This allows us to use the timbre embedding as a flexible control space for the synthesizer, and timbre morphing is possible by interpolating along curves within the embedding space.

To illustrate how the model scales with more diverse timbre, we train the Mel2Mel model with 100 instruments using a 10-dimensional embedding, and we

[1] The icons are made by Freepik and licensed by CC 3.0 BY.

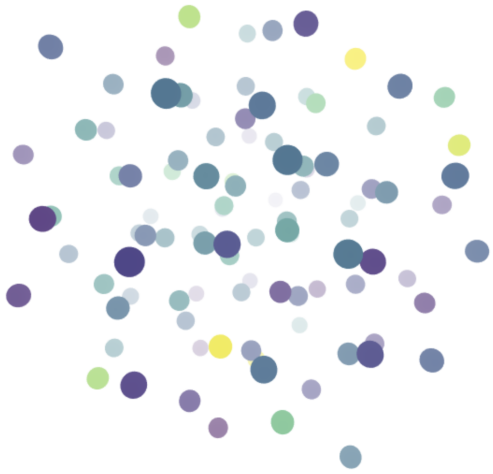

Figure 17: A t-SNE visualization of the 10-dimensional timbre embedding space learned using 100 instruments, color-coded according to the spectral centroid. In the web demo, users can rotate the dots to navigate and click on the dots to play the corresponding audio segments.

refer the readers to the web demo[2] for an interactive t-SNE visualization (van der Maaten & Hinton, 2008) of the embedding space. Figure 17 shows a screenshot of the web demo. The 10-dimensional embedding space also contains a locally continuous timbre distribution of instruments, as in Figure 16, implying that the Mel2Mel model is capable of scaling to hundreds of instruments and to a higher-dimensional embedding space.

In addition to the audio samples used in the experiments, the interactive web demo showcases the capability of flexible timbre control, where the Mel2Mel model runs on browser to convert preloaded or user-provided MIDI files into Mel spectrograms using a user-selected point in the embedding space.

[2] `https://neural-music-synthesis.github.io`

5 Conclusions and Future Directions

We showed that it is possible to build a music synthesis model by combining a recurrent neural network and FiLM conditioning layers, followed by a WaveNet vocoder. It successfully learns to synthesize musical notes according to the given note sequence and timbre embedding in a continuous timbre space, providing the ability of flexible timbre control for music synthesizers.

The capacity of the WaveNet, such as the number of residual channels and the number of layers, is limited due to the memory requirements of the nv-wavenet implementation, and the degradation from μ-law quantization is also apparent in the experiments. These limitations can be overcome by Parallel WaveNet (van den Oord et al., 2018) or WaveGlow (Prenger et al., 2019), which does not require a specialized CUDA kernel for fast synthesis and uses a continuous probability distribution for generation, thereby avoiding the quantization noise. Our earlier experiments on continuous emission failed to stably perform autoregressive sampling due to teacher forcing, and the future work includes investigating this phenomenon comparing with (Hawthorne et al., 2019), which used a mixture of logistics distributions to produce high-quality piano sounds.

A notable observation is that the WaveNet vocoder is able to synthesize plausible polyphonic music from Mel spectrograms containing only 80 frequency bins, which are not even aligned to the tuning of the audio files. This implies that Mel spectrograms contain a close-to-sufficient amount of information for what we perceive from polyphonic music. While more information available from the increased bins should help synthesize more accurate audio, predicting the higher-dimensional representation becomes more compute-intensive and inaccurate, making 80 bins a sweet spot for use with WaveNet. Introducing

an adversarial loss function for predicting high-resolution images (Ledig et al., 2017) can be a viable direction for predicting more accurate and realistic Mel spectrograms for conditioning WaveNet.

To summarize, we have demonstrated in this chapter that a MIDI-to-audio synthesizer can be learned directly from audio, and that this learning allows for flexible timbre control through interpolation in the learned timbre embedding space. While it is certainly a possible future research direction to improve the synthesis quality through the usage of a better vocoder component and a larger and more realistic dataset, the main takeaway of having the synthesis model in the context of this thesis is that it is an end-to-end model that can convert semantic information of music into audio signals. This end-to-end approach not only allows for a simpler and streamlined process of synthesizing music, but also enables using the synthesis model as a differentiable component in a larger neural network. We will examine this possibility in Chapter VII later in this thesis, by combining a music synthesis model with a multi-pitch transcription model introduced in Chapter VI.