An End to End Model for Automatic Music Generation: Combining Deep Raw and Symbolic Audio Networks

Rachel Manzelli*, **Vijay Thakkar***, **Ali Siahkamari, and Brian Kulis**
Department of Electrical and Computer Engineering
Boston University
Boston, MA 02215 USA
{manzelli, thakkarv, siaa, bkulis}@bu.edu

Abstract

We develop an approach to combine two types of music generation models, namely symbolic and raw audio models. While symbolic models typically operate at the note level and are able to capture long-term dependencies, they lack the expressive richness and nuance of performed music. Raw audio models train directly on raw audio waveforms, and can be used to produce expressive music; however, these models typically lack structure and long-term dependencies. We describe a work-in-progress model that trains a raw audio model based on the recently-proposed WaveNet architecture, but that incorporates the notes of the composition as a secondary input to the network. When generating novel compositions, we utilize an LSTM network whose output feeds into the raw audio model, thus yielding an end-to-end model that generates raw audio outputs combining the best of both worlds. We describe initial results of our approach, which we believe to show considerable promise for structured music generation.

Introduction

Deep learning has become an indispensable tool in the field of automated music generation. A number of deep learning architectures have been studied for performing tasks such as pop music generation (Chu, Urtasun, and Fidler 2017) and for creating novel melodies that are similar to classical compositions (Hadjeres, Pachet, and Nielsen 2017). Tools now exist for helping artists write music in order to augment the creative process,[1] and there exist various commercial music generation systems.[2]

Existing generation tools can be broadly classified into two types: symbolic and raw audio models. Symbolic methods train and operate at the note level—based on melodies and notes from existing training data, these methods produce novel melodies in the form of MIDI or related outputs. A popular approach in this space is to use recurrent neural networks, and in particular Long Short Term Memory networks (LSTMs). An advantage of this approach is that the LSTMs are able to capture medium-scale melodic structure in music fairly well (Johnson 2017). They are also typically fast to generate novel melodies, but have the disadvantage that the resulting melodies must be interpreted by a human or synthesizer.

With the recent success of speech synthesis models such as WaveNet (van den Oord et al. 2016), there has been increased interest in utilizing raw audio models for music generation. Raw audio models operate directly on the audio waveforms, both at training and generation times. WaveNet, for instance, is a model that attempts to predict the next sample of audio. The waveforms are typically sampled at 16 kHz, resulting in a computationally expensive model that requires many predictions per second of audio. Once the model has been trained on existing audio, novel audio can be generated, though the generation time for the original WaveNet model is slow. The advantage of raw audio models, however, is that they have the ability to produce considerable richer and more nuanced musical outputs that can capture emotion, mood, etc. In initial work, the model was shown to produce realistic-sounding piano music. Other advantages of these methods include the ability to produce novel sounds, such as combining multiple instruments together (Simon and Oore 2017), as well as generating vocals (Blaauw and Bonada 2017). Further, recent work on WaveNet has developed faster generation methods (Le Paine et al. 2016). Unfortunately, existing raw audio models are limited in that they fail to capture any medium- or long-term effects in the music; most generated music sounds improvisational, with no clear strucuture.

We believe that combining these two approaches will yield new possibilities for music generation, and open the door to a host of new music generation tools. In particular, we propose to train a biaxial LSTM to create symbolic melodies, and then treat these melodies as local conditioning of a WaveNet model. Thus, the LSTM model allows us to create long-term melodic structure in the music, while the WaveNet component interprets and expands upon the generated melodic structure.

The research presented in this paper is work-in-progress, but we discuss some initial experimental results. We first discuss tuning of the unconditioned WaveNet model to produce classical piano music. Once we have tuned this model, we then discuss our extensions to the conditioned case. We first

*Equal contribution.

[1]https://www.ampermusic.com

[2]https://www.jukedeck.com

generate raw audio on pre-specified melodies (i.e. Happy Birthday and Ode to Joy) after training on the MusicNet dataset; then we demonstrate preliminary results of training both the LSTM and the WaveNet on the training data and generating a realistic raw audio melody by using the output of the LSTM generation as a local conditioning time series to the WaveNet model.

Background

We elaborate on two prevalent deep learning models for music generation, namely raw audio models and symbolic models.

Generative Raw Audio Models

Initial efforts to generate raw audio involved models used primarily for image generation, such as char-rnn and LSTMs. Raw audio generations from these networks are often noisy and unstructured; they are limited in their capacity to abstract higher level representations of raw audio, mainly due to problems with overfitting (Briot, Hadjeres, and Pachet 2018).

In 2016, DeepMind introduced WaveNet (van den Oord et al. 2016), a generative model for general raw audio, designed mainly for speech applications. At a high level, WaveNet is a deep learning architecture that operates directly on a raw audio waveform. In particular, for a waveform modeled by $x = \{x_1, ..., x_T\}$ (representing speech, music, etc.), the joint probability of the entire waveform can be factorized as a product of conditional probabilities, namely

$$p(x) = \prod_{t=1}^{T} p(x_t | x_1, ..., x_{t-1}). \tag{1}$$

The waveforms in WaveNet are typically represented as 8-bit audio, meaning that each x_i can take on one of 256 possible values. The WaveNet model uses a deep learning architecture to model the conditional probabilities $p(x_t | x_1, ..., x_{t-1})$. The model is trained by predicting values of the waveform at step t and comparing them to the true value x_t, using cross-entropy as a loss function; thus, the problem simply becomes a multi-class classification problem (with 256 classes) for each step in the waveform. The architecture of the conditional probabilities utilizes causal convolutions, similar to masked convolutions used in PixelRNN and similar image generation networks (van den Oord, Kalchbrenner, and Kavukcuoglu 2016). Causal convolutions ensure that the prediction for time step t only depends on the predictions for previous time steps. Furthermore, the causal convolutions are dilated; this is a convolution where the filter is applied over an area larger than its length by skipping particular input values, as shown in Figure 1. In addition to the dilated causal convolutions, each layer features gated activation units and residual connections, as well as skip connections to the final output layers.

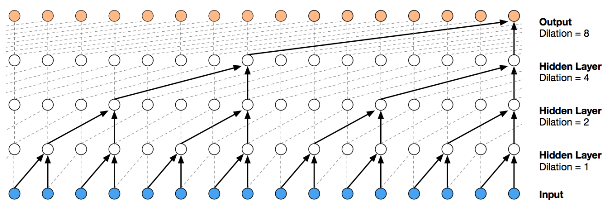

Figure 1: A stack of dilated causal convolutions as used by WaveNet, reproduced from (van den Oord et al. 2016).

Generative Symbolic Audio Models

Most deep learning systems for automatic music generation are based on symbolic representations of the music. One of the most popular of these is MIDI (Musical Instrument Digital Interface)[3], which is a standard format and protocol for electronic music. Other representations that have been utilized include the piano roll representation (Huang and Wu 2016)—inspired by player piano music rolls—text representations (e.g., ABC notation[4]), chord representations (e.g., Chord2Vec (Madjiheurem, Qu, and Walder 2016)), and lead sheet representations. A typical scenario for producing music in such models is to train and generate on the same type of representation; for instance, one may train on a set of MIDI files that encode melodies, and then generate new MIDI melodies from the learned model.

There are now a very large number of existing deep learning symbolic music systems (Briot, Hadjeres, and Pachet 2018), including models that are based on recurrent neural networks (RNNs), many of which use Long Short Term Memory (LSTM) components. Some of the models using RNNs include the Blues Melody Generation System (Eck and Schmidhuber 2002), DeepBach (Hadjeres, Pachet, and Nielsen 2017), the CONCERT system (Mozer 1994), the Celtic Melody Generation system (Sturm et al. 2016), the biaxial LSTM model (Johnson 2017), and several methods that combine RNNs with restricted Boltzmann machines (Boulanger-Lewandowski, Bengio, and Vincent 2012; Goel, Vohra, and Sahoo 2014; Chung et al. 2014; Lyu et al. 2015).

Architecture

We first discuss our symbolic method for generating unique melodies, then detail the modifications to the raw audio model for compatibility with these generations. Modifying the architecture involves working with both symbolic and raw audio data in harmony.

Melody Generation with LSTM Networks

A traditional approach to learning temporal dependencies in data is to use a Recurrent Neural Network (RNN). The general topology of a recurrent network is such that the output of the previous time step is fed as input to the next, allowing the network to learn sequential dependencies in the data.

[3]https://www.midi.org/specifications

[4]http://abcnotation.com

However, in practice, RNNs do not perform well when the learning sequences have long term temporal dependence due to issues such as exploding gradients (Y. Bengio and Frasconi 1994). This is especially true for music, as properties such as key signature and time signature may be constant throughout a piece of music.

Long-Short Term Memory models are designed to overcome this issue by providing a mechanism that is able to both store and discard the information saved about the previous steps, limiting the accumulated error using Constant Error Carousels (Hochreiter and Schmidhuber 2006). This error carousel serves as a flow of cell state through the recurrent cells of the network, encoding the dependence on previous time steps. This state is modified by each unit through a gated structure, allowing the network to selectively retain important dependence data in the cell state.

Recently, applications of LSTMs specific to music generation, such as the biaxial LSTM, have been explored that utilize a pair of tied, parallel networks (Johnson 2017). This approach imposes an LSTM in the temporal dimension, and another in the pitch dimension at each time step. This allows the model to not only learn the overall structure of the music, but also captures the inter-dependence of the notes being played at any given timestep.

We explore the sequential combination of the symbolic and raw audio models to produce structured raw audio output. We train a biaxial LSTM model on the training data of a particular genre, and then feed the generations from this trained model into the raw audio network.

Conditioning with Raw Audio Models

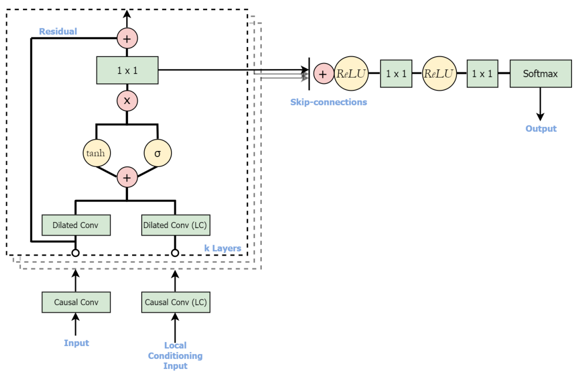

Figure 2: An overview of the model architecture, showing the local conditioning time series as an extra input.

Once a learned symbolic melody is obtained, we treat it as a second time series within our raw audio model (analogous to using a second time series with a desired text to be spoken in the speech domain). In particular, in the WaveNet model, each layer features a gated activation unit. If x is the raw audio input vector, then at each layer k, it passes through the following gated activation unit:

$$z = \tanh(W_{f,k} * x) \odot \sigma(W_{g,k} * x), \qquad (2)$$

Instrument	Minutes	Labels
Piano	1,346	794,532
Violin	874	230,484
Cello	621	99,407
Solo Piano	917	576,471
Solo Violin	30	8,837
Solo Cello	49	10,876

Table 1: Statistics of the MusicNet dataset. (Thickstun, Harchaoui, and Kakade 2017)

where $*$ is a convolution operator, $\odot$ is an elementwise multiplication operator, $\sigma(\cdot)$ is the sigmoid function, and the $W_{f,k}$ and $W_{g,k}$ are learnable convolution filters. Following WaveNet's use of local conditioning, we can introduce a second time series y (in this case from the LSTM model, to capture the long-term melody), and instead utilize the following activation, effectively incorporating y as an extra input:

$$z = \tanh(W_{f,k} * x + V_{f,k} * y) \odot \sigma(W_{g,k} * x + V_{g,k} * y), \qquad (3)$$

where V are learnable linear projections. By conditioning on an extra time series input, we effectively guide the raw audio generations to require certain characteristics; y influences the output at all timestamps.

In our modified WaveNet model, the y timeseries is the upsampled MIDI embedding timeseries. In particular, LC embeddings are 128-dimensional binary vectors where ones correspond to note indicies that are being played at the current time step. As with the audio timeseries, the LC embeddings first go through a layer of causal convolution to reduce the number of dimensions from 128 down to 16, which are then used in the dilation layers as the conditioning samples. This reduces the computational requirement for the dilation layers without reducing the note state information, as most of the embeddings are zero for most timestamps.

Evaluation

As any generated piece of music can generally only be subjectively evaluated by human listeners, it is challenging to quantitatively evaluate the generations from our model. For instance, to compare the outputs of two systems, one can have a human listen to excerpts from both, and rate which of the two sounds more natural (or whichever parameter is desired to be evaluated). We plan to use such crowdsourced evaluation techniques (specifically, Amazon Mechanical Turk[5]) to compare our outputs with other systems; however, for now, we evaluate our results in terms of the complexity and structure of the musical outputs we have obtained so far. Example results of generations, including results from our follow-up ISMIR paper, are posted on our web page.[6]

Training Datasets

At training time, in addition to raw training audio, we must also incorporate its underlying symbolic melody, perfectly

[5]https://www.mturk.com/mturk/

[6]http://people.bu.edu/bkulis/projects/music/index.html

aligned with the raw audio at each timestep. The problem of melody extraction in raw audio is still an active area of research; due to a general lack of such annotated music, we have experimented with multiple datasets, which we enumerate here.

We have utilized the Melodia algorithm (Salamon and Gomez 2012) to generate training data. By automatically detecting the pitch of a melody, the algorithm is able to piece together symbolic data given a raw audio input. The algorithm is centered around the idea of detecting pitch contours, which are groups of pitch candidates arranged sequentially using auditory streaming cues. However, the algorithm only detects a song's main melody; thus, the resulting symbolic alignments were very sparse when multiple notes were played at once in a song.

Experiments were also performed using the Saarland Music Data (Müller et al. 2011). This dataset contains 50 pieces of Western piano music with perfectly aligned MIDI sequences. Due to limited resources and our choice to focus solely on cello in this work, extensive results from training on this dataset were not generated.

We are currently exploring the use of the MusicNet database for training (Thickstun, Harchaoui, and Kakade 2017), as this data features both raw audio as well as melodic annotations. Other metadata is also included, such as the composer of the piece, the instrument with which the composition is played, and each note's position in the metrical structure of the composition. The music is separated by genre; there are over 900 minutes of solo piano alone, which has proven to be very useful in training on only one instrument. The details of this dataset are enumerated in Table 1.

Unconditioned Music Generation with WaveNet

The WaveNet model was designed mainly for speech applications, as it was deployed in the Google Assistant. Thus, we preface our evaluation by first acknowledging the fact that we originally tuned WaveNet for unstructured music generation. We tuned the model to generate music trained on solo piano inputs (about 50 minutes of Chopin Nocturnes, from the YouTube-8M dataset (Abu-El-Haija et al. 2016)), as well as 350 songs of various genres of electronic music, obtained from No Copyright Sounds[7].

Upon training the model with multi-track electronic music, we found the generations to be very noisy and usually only containing information from a single instrument, such as the bass line or drums. With complex, diverse training data of this type, WaveNet does not generalize well. The mix of lyrics and many different types of sounds in the music was not consistent over the training data, which led to poor generations. There is a tradeoff between versatility of the trained networks in the genres it can produce and the quality of said generations.

In the case of training on single instruments, such as cello and piano, we discovered that WaveNet models are capable of producing complex musical generations without losing instrumental quality as the number of generated samples increases. The network is able to learn short-range dependencies, including hammer action and simple chords. We found that the generations maintain consistent emotion throughout. However, they are unstructured and do not contain any long-range temporal dependencies. Results that showcase these techniques and attributes are available on our webpage.

[7]https://www.youtube.com/user/NoCopyrightSounds

Structure in Raw Audio Generations

Though it is challenging to quantitatively evaluate the structuring ability of our conditioned model, we may attempt to visualize a given output against its conditioning sequence via a peak frequency spectrogram, as shown in Figure 3.

We note that for a given timestep, both the symbolic input and the raw audio generation have similar power at their respective frequencies, e.g. the color in the spectrogram intensifies at similar timesteps throughout the song. This indicates that our model is predicting samples in time with the rhythm of the given raw audio.

The relative heights between peak frequencies are similar for each piece of music, indicating that the frequency of the output is changing in a similar way throughout the piece. This indicates that the raw audio output closely follows the dynamics of the conditioning signal.

The difference in overall vertical height of the spectrograms is due to the fact that the symbolic input was interpreted with a different instrument than the raw audio output, and was played at a different overall frequency. We may also note that even though multiple notes are being played at the same time throughout the symbolic input, the raw audio model does indicate sound at these frequencies at similar timesteps (more softly, though, since the color is not as intense).

This analysis generalizes to all of the pieces generated with our model; we have successfully been able to transform an unstructured model with little long-range dependency to one with guided generations that exhibit certain characteristics.

We are currently experimenting with and optimizing the output of the model with the creative melody generation from the LSTM. The results, further detailed in our follow-up ISMIR paper, are posted on our webpage.

Conclusions and Future Work

In conclusion, we focus on developing a work-in-progress model to combine raw and symbolic audio models for the improvement of automatic music generation. Combining two prevalent models allows us to take advantage of both of their features; in the case of raw audio models, this is the realistic sound and feel of the music, and in the case of symbolic models, it is the complexity, structure, and long-range dependency of the generations.

Our next steps to effectively move toward a hands-free, creative raw audio music generator include successfully using the unique LSTM generations as a second time series input to the WaveNet model. We showcase results of this in our follow-up ISMIR paper (Manzelli et al. 2018). An additional future modification of our approach could be to merge the LSTM and WaveNet models to a coupled architecture. This would eliminate the need to synthesize MIDI

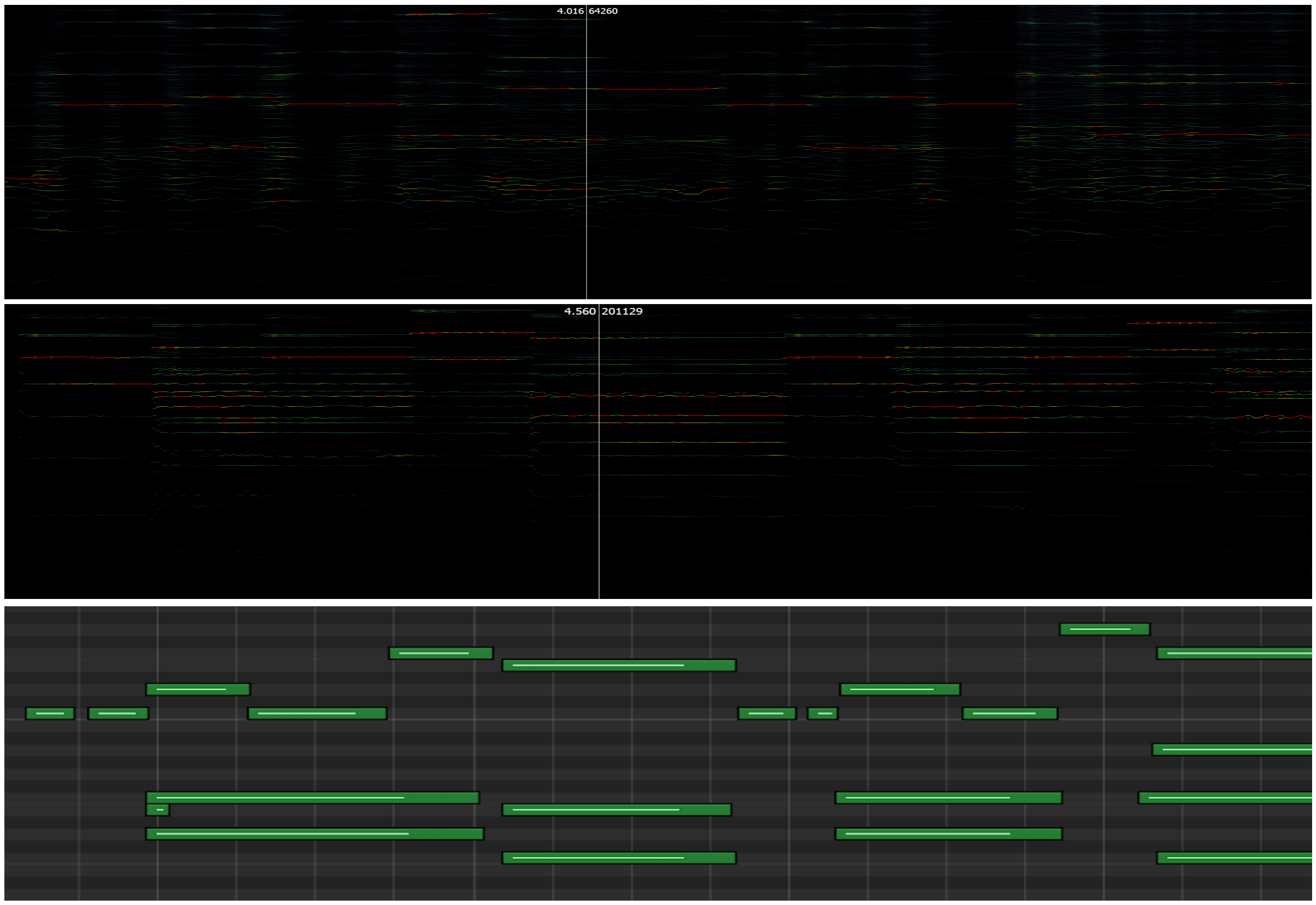

Figure 3: The peak frequency spectrograms for the raw audio output of a cello model (top) and its conditioning signal, which is a portion of the Happy Birthday melody played on chimes (middle). Despite the two waveforms sounding different, the figure shows the similar structure of each spectrogram to the conditioning sequence (bottom). The audio files are posted at https://ismir2018submissio.wixsite.com/music/music.

files, as well as the need for MIDI labels associated with the raw audio training data.

The architecture proposed in this paper allows for a modular approach to automated music generation; our model as it stands can inspire many future applications. For example, multiple different instances of our model can be trained on different genres of music, and generate based on a single local conditioning series split among these networks. As a result, the same melody can be reproduced in different genres or instruments, and can be strung together to create effects such as a quartet or band. The key application here is that this type of synchronized effect can be achieved without awareness of the other networks, avoiding model interdependence.

Additionally, the combination of our model with multiple audio domains could also be implemented; specifically, this would involve the integration of speech audio with music to produce lyrics sung in tune with our realistic melody.

References

Abu-El-Haija, S.; Kothari, N.; Lee, J.; Natsev, P.; Toderici, G.; Varadarajan, B.; and Vijayanarasimhan, S. 2016. YouTube-8M: A large-scale video classification benchmark. *ArXiv preprint 1609.08675*.

Blaauw, M., and Bonada, J. 2017. A neural parametric singing synthesizer. In *Proc. Interspeech*.

Boulanger-Lewandowski, N.; Bengio, Y.; and Vincent, P. 2012. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proc. International Conference on Machine Learning (ICML)*.

Briot, J.; Hadjeres, G.; and Pachet, F. 2018. *Deep Learning Techniques for Music Generation*. Springer International Publishing.

Chu, H.; Urtasun, R.; and Fidler, S. 2017. Song from PI: A musically plausible network for pop song generation. In *Proc. International Conference on Learning Representations (ICLR), Workshop Track*.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *ArXiv preprint 1412:3555.*

Eck, D., and Schmidhuber, J. 2002. A first look at music composition using LSTM recurrent neural networks. Technical Report IDSIA-07-02, IDSIA/USI-SUPSI.

Goel, K.; Vohra, R.; and Sahoo, J. K. 2014. Polyphonic music generation by modeling temporal dependencies using a RNN-DBN. In *Proc. International Conference on Artificial Neural Networks*.

Hadjeres, G.; Pachet, F.; and Nielsen, F. 2017. DeepBach: a stterable model for Bach chorales generation. In *International Conference on Machine Learning (ICML)*.

Hochreiter, S., and Schmidhuber, J. 2006. Long short term memory. *Neural Computation* 9(8):1735–1780.

Huang, A., and Wu, R. 2016. Deep learning for music. *ArXiv preprint 1606:04930*.

Johnson, D. D. 2017. Generating polyphonic music using tied parallel networks. In *International Conference on Evolutionary and Biologically Inspired Music and Art*.

Le Paine, T.; Khorrami, P.; Chang, S.; Zhang, Y.; Ramachandran, P.; Hasegawa-Johnson, M. A.; and Huang, T. S. 2016. Fast WaveNet generation algorithm. *ArXiv preprint 1611.09482*.

Lyu, Q.; Wu, Z.; Zhu, J.; and Meng, H. 2015. Modelling high-dimensional sequences with LSTM-RTRBM: Application to polyphonic music generation. In *Proc. International Artificial Intelligence Conference (AAAI)*.

Madjiheurem, S.; Qu, L.; and Walder, C. 2016. Chord2Vec: Learning musical chord embeddings. In *Advances in Neural Information Processing Systems (NIPS)*.

Manzelli, R.; Thakkar, V.; Siahkamari, A.; and Kulis, B. 2018. Conditioning deep generative raw audio models for structured automatic music. In *Proc. International Society for Music Information Retrieval Conference (ISMIR)*.

Mozer, M. C. 1994. Neural network composition by prediction: Exploring the benefits of psychophysical constraints and multiscale processing. *Connection Science* 6(2–3):247–280.

Müller, M.; Konz, V.; Bogler, W.; and Arifi-Müller, V. 2011. Saarland music data (SMD). http://resources.mpi-inf.mpg.de/smd/index.html.

Salamon, J., and Gomez, E. 2012. Melody extraction from polyphonic music signals using pitch contour characteristics. *IEEE Transactions on Audio, Speech and Language Processing* 20(6):1759–1770.

Simon, I., and Oore, S. 2017. Performance RNN: Generating music with expressive timing and dynamics. *Magenta Blog*. https://magenta.tensorflow.org/performance-rnn.

Sturm, B. L.; Santos, J. F.; Ben-Tal, O.; and Korshunova, I. 2016. Music transcription modelling and composition using deep learning. In *Conference on Computer Simulation of Musical Creativity*.

Thickstun, J.; Harchaoui, Z.; and Kakade, S. M. 2017. Learning features of music from scratch. In *International Conference on Learning Representations (ICLR)*.

van den Oord, A.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. WaveNet: A generative model for raw audio. *ArXiv preprint 1609.03499*.

van den Oord, A.; Kalchbrenner, N.; and Kavukcuoglu, K. 2016. Pixel recurrent neural networks. In *Proc. International Conference on Machine Learning (ICML)*.

Y. Bengio, P. S., and Frasconi, P. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* 5(2):157–166.