A HIERARCHICAL RECURRENT NEURAL NETWORK FOR SYMBOLIC MELODY GENERATION

Jian Wu[1], *Changran Hu*[2], *Yulong Wang*[1], *Xiaolin Hu*[1], *Jun Zhu*[1]

[1]Department of Computer Science and Technology, Tsinghua University
[2]Department of Electronic Engineering, Tsinghua University
{wuj16@mails, hcr14@mails, wang-yl15@mails, xlhu@mail, dcszj@mail }.tsinghua.edu.cn

ABSTRACT

In recent years, neural networks have been used to generate music pieces, especially symbolic melody. However, the long-term structure in the melody has posed great difficulty for designing a good model. In this paper, we present a hierarchical recurrent neural network for melody generation, which consists of three Long-Short-Term-Memory (LSTM) subnetworks working in a coarse-to-fine manner. Specifically, the three subnetworks generate bar profiles, beat profiles and notes in turn, and the output of the high-level subnetworks are fed into the low-level subnetworks, serving as guidance for generating the finer time-scale melody components. Two human behavior experiments demonstrate the advantage of this structure over the single-layer LSTM which attempts to learn all hidden structures in melodies. In the third human behavior experiment, subjects are asked to judge whether the generated melody is composed by human or computer. The results show that 33.69% of the generated melodies are wrongly classified as human composed.

Index Terms— melody generation, recurrent neural network

1. INTRODUCTION

Music has been an artistic domain where deep learning can play a significant role. For example, WaveNet can generate realistic raw audio musical fragments [1]. Instead of generating the acoustic output, in this study, we focus on symbolic melody generation, which requires learning from sheet music.

Many music genres such as pop music consist of melody and harmony. And in this study, we only focus on melody generation, similar to some recent studies [2][3][4]. One reason is that beautiful harmonies can be ensured by using legitimate chord progressions which have been summarized by musicians. This greatly simplifies the melody generation problem.

Melody is a linear succession of musical notes along time. It has both short time scale such as notes and long time scale such as phrases and movements, which makes the melody generation a challenging task. Existing methods generate pitches and rhythm simultaneously [2] or sequentially [5] using Recurrent Neural Networks (RNNs), but they usually work on the note scale without explicitly modeling the larger time-scale components such as rhythmic patterns. It is difficult for them to learn long-term structures in melody.

Theoretically, an RNN can learn the temporal structure of any length in the input sequence, but in reality, as the sequence gets longer it is very hard to learn long-term structure. Different RNNs have different learning capability, e.g., LSTM [6] performs much better than the simple Elman network. But any model has a limit for the length of learnable structure, and this limit depends on the complexity of the sequence to be learned. To enhance the learning capability of an RNN, one approach is to invent a new structure. In this work we take another approach: increase the granularity of the input. Since each symbol in the sequence corresponds to longer segment than the original representation, the same model would learn longer temporal structure.

To implement this idea, we propose a Hierarchical Recurrent Neural Network (HRNN) for learning melody. It consists of three LSTM-based sequence generators — Bar Layer, Beat Layer and Note Layer. The Bar Layer and Beat Layer are trained to generate bar profiles and beat profiles, which are designed to represent the high-level temporal features of melody. The Note Layer is trained to generate melody conditioned on the bar profile sequence and beat profile sequence output by the Bar Layer and Beat Layer. By learning on different time scales, the HRNN can grasp the general regular patterns of human composed melodies in different granularities, and generate melody with realistic long-term structures. This method follows the general idea of granular computing [7], in which different resolutions of knowledge or information is extracted and represented for problem solving. With the shorter profile sequences to guide the generation of note sequence, the difficulty of generating note sequence with well-organized structure is alleviated.

We trained the HRNN on a dataset which consists of 3,864 lead sheets. To evaluate the effect of the hierarchical architecture, HRNN was compared with the single-layer LSTM model in human behavior experiments. The results showed that HRNN significantly outperformed the single-layer LSTM model.

2. RELATED WORK

2.1. Melody Generation with Neural Networks

There is a long history of generating melody with RNNs. A recurrent autopredictive connectionist network called CONCERT is used to compose music [8]. With a set of composition rules as constraints to evaluate melodies, an evolving neural network is employed to create melodies [9]. As an important form of RNN, LSTM [6] is used to capture the global music structure and improve the quality of the generated music [10]. Lookback RNN and Attention RNN are proposed to tackle the problem of creating melody's long-term structure [2]. The Lookback RNN introduces a handcrafted lookback feature that makes the model repeat sequences easier while the Attention RNN leverages an attention mechanism to learn longer-term structures. Inspired by convolution, two variants of RNN are employed to attain transposition invariance [11]. To model the relation between rhythm and melody flow, a melody is divided into pitch sequence and duration sequence and these two sequences are processed in parallel [12]. This approach is further extended in [4]. A 4-layer LSTM is employed to produce the key, press, chord and drum of pop music seperately [5]. This is clearly different from our model because all layers in our model are used to generate notes.

More recently, Generative Adversarial Networks (GANs) have also been used to generate melodies. For example, RNN-based GAN [13] and CNN-based GAN [3] are employed to generate melodies, respectively. However, currently, a good approach to training GANs with long sequences is still lacked, and their generated melodies also face the problem of lacking long-term realistic structures.

2.2. Hierarchical and Multiple Time Scales Networks

The idea of hierarchical or multiple time scales has been used in neural network design, especially in the area of natural language processing. The Multiple Timescale Recurrent Neural Network (MTRNN) realizes the self-organization of a functional hierarchy with two types of neurons "fast" unit and "slow" unit [14]. Then it is shown that the MTRNN can acquire the capabilities to recognize, generate, and correct sentences in a hierarchical way: characters grouped into words, and words into sentences [15]. An LSTM auto-encoder is trained to preserve and reconstruct paragraphs by hierarchically building embeddings of words, sentences and paragraphs [16]. To process inputs at multiple time scales, the Clockwork RNN is proposed, which partitions the hidden layers of RNN into separate modules. Different from the Clockwork RNN, we integrate the prior knowledge of music in constructing the hierarchical model and feed multiple time scales of features to different layers.

3. MUSIC CONCEPTS AND REPRESENTATION

We first introduce the melody representation and then the handcrafted rhythmic profiles for HRNN. For readers who do not have a music background, some basic music concepts are introduced in the **Supplementary Text and Fig. S1**.

3.1. Melody Representation

We only chose musical pieces with a time signature of 4/4, since it is a common and widely-used time signature. According to the statistics on the Wikifonia dataset described in Section 5, about 99.83% of notes have pitches between C3 and C6. Thus, all notes are octave-shifted to this range. Then there are 36 options for a pitch of a note (3 octaves and each octave has 12 notes). To represent duration, we apply event messages used in the Midi standard. When a note is pressed, a note-on event with the corresponding pitch happens; and when the note is released, a note-off event happens. For a monophonic melody, if two notes are adjacent, the note-on event of the latter indicates the note-off event of the former, and the note-off event of the former is therefore not needed. In this study, every bar was discretized into 16 time steps. At every time step, there are 38 kinds of events (36 note-on events, one note-off event and one no-event), which are exclusive. One example is shown in **Supplementary Fig. S2**. In this way, note-on events mainly determine the pitches in the melody and no-events mainly determine the rhythm as they determine the duration of the notes. So a 38-dimensional one-hot vector is used to represent the melody at every time step.

3.2. Rhythmic Patterns and Profiles

Rhythmic patterns are successions of durations of notes which occur periodically in a musical piece. It is a concept on a larger time scale than the note scale and is important for melodies' long-term structure. Notice that in this model we do not encode the melody flow because it is hard to find an appropriate high-level representation of it.

Two features named *beat profile* and *bar profile* are designed, which are high-level representations of a whole bar and beat, respectively. Compared with individual notes, the two profiles provide coarser representations of the melody. To construct the beat profile set, all melodies are cut into melody clips with a width of one beat and binarized at each time step with 1 for an event (note-on events and note-off event) and 0 for no-event at this step. Then we cluster all these melody clips into several clusters via the K-Means algorithm and use the cluster centers as our beat profiles. Given a one beat melody piece, we can binarize it in the same manner and choose the closest beat profile as its representation. The computation of bar profile is similar, except that the width of melody clip is changed to one bar. Based on the well-known elbow method, the numbers of clusters for beat profiles and

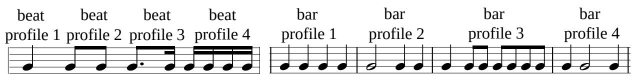

Fig. 1. Samples of beat profiles and bar profiles. Here we use notes with same pitch to illustrate rhythm in beat and bar.

bar profiles are set to be 8 and 16 respectively. In Fig. 1, some frequently appeared profiles are shown with notes.

4. HIERARCHICAL RNN FOR MELODY GENERATION

4.1. Model Architecture

HRNN consists of three event sequence generators: Bar Layer, Beat Layer and Note Layer, as illustrated in Fig. 2. These layers are used to generate bar profile sequence, beat profile sequence and note sequence, respectively.

The lower-level generators generate sequences conditioned on the sequence output by the higher-level generators. So to generate a melody, one needs to first generate a bar profile sequence and a beat profile sequence in turn. Consider that we want to generate a melody piece with the length of one bar, which is represented as $n_t, ..., n_{t+15}$. First, the Bar Layer generates a bar profile B_t with the last bar profile B_{t-16} as input. Then the Beat Layer generates 4 beat profiles $b_t, ..., b_{t+12}$ with b_{t-4} as input conditioned on the bar profile B_t. To generate the notes $n_t, n_{t+1}, ...n_{t+3}$, the Note Layer is conditioned on both B_t and b_t; to generate the notes $n_{t+4}, ..., n_{t+7}$, the Note Layer is conditioned on both B_t and b_{t+4}; and so on. In this way, each bar profile is a condition for the 16 generated notes and each beat profile is a condition for the 4 generated notes.

All of the three layers use LSTM but the granularity of the input is different. Theoretically, the Beat Layer and Bar Layer can learn 4 and 16 times longer temporal structure than the Note Layer, respectively. Note that it is difficult to quantify the length of temporal structure learned in a model, since "temporal structure" is an abstract concept and its characterization is still an open problem. We could only probe the difference in length produced by different models indirectly by measuring the quality of the generated sequences using behavior experiments (See Section 5).

To explicitly help RNN memorize recent events and potentially repeat them, a Lookback feature was proposed for *the Lookback RNN* [2]. A user study suggested that the RNN with Lookback feature outperforms basic RNN [3] so we also use it in our model[1]. The lookback distance is 2 and 4 for the Bar Layer, 4 and 8 for the Beat Layer, 4 and 8 for the Note Layer. Therefore, the Note Layer without the condition of the Beat layer and Bar layer is equivalent to *the Lookback RNN*.

[1]For fair comparison in experiments, all models were equipped with this feature.

4.2. LSTM Based Event Sequence Generator

Bar profiles, beat profiles and notes can be abstracted as events. We use the same LSTM based event sequence generator for the Bar Layer, Beat Layer and Note Layer.

The event sequence generator G_θ was trained by solving the following optimization problem:

$$\max_{\theta} \sum_{y \in \mathcal{Y}} \sum_{t=1}^{\text{len}(y)} \log p(y_t | y_0, ..., y_{t-1}, c_t) \tag{1}$$

where θ are the parameters of the generator, y is a sequence sampled in the event sequences dataset $\mathcal{Y}$. And y_t denotes the t-th event in y, c_t denotes the condition for y_t.

To predict the conditional probability in Eq. (1), we use a well-known recurrent neural network, LSTM, which involves input gates, output gates and forgetting gates [6].

During generation, given a primer sequence as an initial input sequence, the LSTM network generates the distribution p_0 over all candidate events. The next event was chosen by sampling over p_0. The successive events are generated according to $p(y_t | y_0, ..., y_{t-1}, c_t)$.

5. EXPERIMENTS

Evaluating the performance of the models for melody generation is difficult. The main reason is that measuring the quality of the generated melodies is subjective and it is hard to find an objective metric.

We evaluated three generative models, HRNN-1L, HRNN-2L and HRNN-3L mainly based on behavioral experiments. HRNN-3L is the model we described in the previous section. HRNN-2L is the HRNN-3L without the Bar Layer while HRNN-1L is the HRNN-3L without the Bar Layer and the Beat Layer. All of the music pieces generated by HRNN were not post-processed.

5.1. Implementation Details

All LSTM networks used in experiments had two hidden layers and each hidden layer had 256 hidden neurons. They were trained with Adam algorithm [17] and initial learning rate was 0.001. The minibatch size was 64. To avoid over-fitting, validation-based early stopping (see Fig. 3) and dropout with ratio 0.5 were adopted.

In each generation trial, primer sequences (both profiles and melodies) were randomly picked from the validation dataset. For the Bar Layer and Beat Layer, one profile is given as the primer. For the Note Layer, the length of the primer sequence is 1 beat. Beam search with a beam of size 3 was used in all experiments.

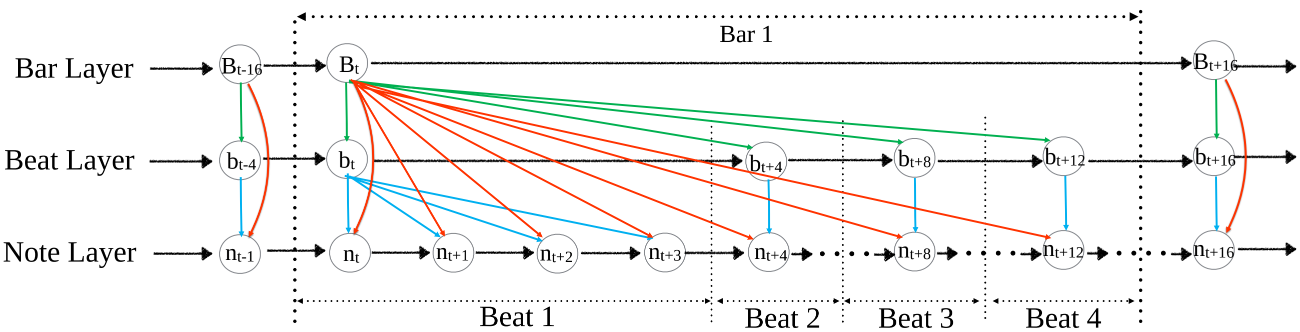

Fig. 2. Architecture of HRNN. From top to bottom are Bar Layer, Beat Layer and Note Layer respectively. Inner layer connections along time are shown with black lines. Connections between layers are shown with green lines, blue lines and red lines.

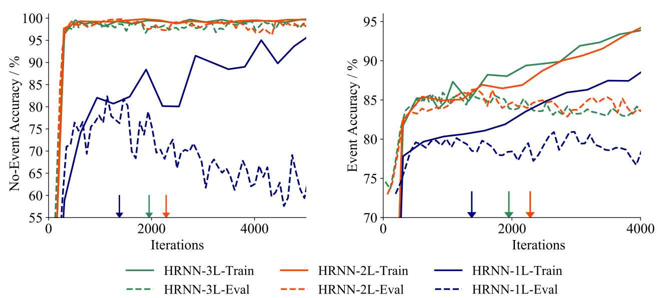

Fig. 3. The accuracy curves of Note Layer for training and validation dataset. **Left**: no-event accuracy curves. **Right**: event (both note-on event and note-off event) accuracy curves. Arrows indicate iterations at which training stopped to prevent over-fitting.

5.2. Dataset

We collected 3,864 lead sheets with the time signature of 4/4 in MusicXML format from `http://www.wikifonia.org`. 90% of the lead sheets were used as training set and the other 10% were used as validation set. The speed of most music pieces in the dataset is 120 beats per minute.

5.3. Guiding Effect of Profiles

To verify whether the beat and bar profiles can guide the generation of melody, we plotted the Note Layer's accuracy curves of in Fig. 3. Here both event accuracy (accuracy of the note-on event and note-off event; chance level is $1/37$) and no-event accuracy (chance level is $1/2$) are plotted.

With beat and bar profiles, the Note Layer learned the pattern of no-event quickly and easily. For models with profiles, the accuracy of no-event increased to nearly 100% at about 200 iterations while the model without profile converged slowly and over-fitting started after about 2000 iterations. Since rhythm is encoded by no-event (see Section 3.1), this showed that the Note Layer successfully utilized the rhythm provided by the beat and bar profiles. The accuracy of note-on and note-off events also improved, which means models with profiles not only did a good job in predicting rhythm, but also in predicting pitch.

(a) Beat profiles: "1, 1, 1, 2, ..., 1, 1, 1, 2"

(b) Bar profiles: "2, 1, 2, 1, ..., 2, 1, 2, 1"

Fig. 4. Melodies generated with given beat profile sequences and bar profile sequences.

With given profile sequences, the Note Layer will generate melodies with rhythm represented by profile sequence. To show this, we used handcrafted profile sequences to guide the generation of the Note Layer. Fig. 4 shows generated melodies given beat profile sequence "1, 1, 1, 2, ..., 1, 1, 1, 2" to HRNN-2L and bar profile sequence "1, 2, 1, 2, 1, 2, 1, 2" to HRNN-3L (profile index in Fig. 1). The results verified that the generated melodies are strongly constrained by the given profile sequence patterns. The same conclusion can be obtained using fixed beat profiles and bar profiles extracted from existing melodies (see **Supplementary Fig. S3**).

5.4. Qualitative Comparison

The strong guiding effect of profiles implies that the Note Layer could output good melodies if the higher layers could generate good profile sequences. Since note sequences are much longer than their corresponded profile sequences, learning the latter should be easier than learning the former us-

Fig. 5. Melodies generated by HRNN-1L, HRNN-2L and HRNN-3L.

ing the same type of model. Thus, compared to HRNN-1L, melodies generated by HRNN-2L and 3L model should be more well-organized and keep better long-term structures. The qualitative comparison verified this point. Three typical music pieces generated by HRNN-1L, HRNN-2L, HRNN-3L with the same primer note were shown in Fig. 5. The melody generated by HRNN-1L has basic rhythm, but also irregular rhythmic patterns. And the melodies generated by HRNN-2L and HRNN-3L contain less irregular rhythmic patterns.

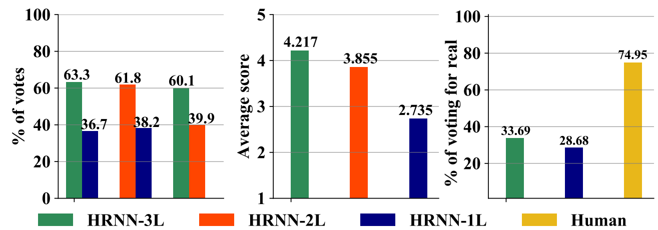

Fig. 6. Results of the 2AFC experiment (**left**), the melody score experiment (**middle**), and the music turing test (**right**).

5.5. Human Behavior Experiments

Three human behavior experiments were conducted to evaluate melodies generated by models. For this propose, we built an on-line website where people could listen to melodies and give their feedback. Listen to **Supplementary Audios** for samples of melodies evaluated in these experiments.

5.5.1. Two-Alternative Forced Choice Experiment

We randomly provided subjects pairs of melodies with the length of 16 bars (about 32 seconds) and asked them to vote which melody sounded better in every pair. This is the two-alternative forced choice (2AFC) setting.

Three types of pairs were compared: HRNN-3L versus HRNN-1L, HRNN-2L versus HRNN-1L and HRNN-3L versus HRNN-2L. Each model generated a set of 15 melodies and in every trial two melodies were randomly sampled from the two corresponding sets. Different types of pairs were mixed and randomized in the experiment.

Call for participants advertisement was spread in a social media WeChat. 1637 trials were collected from 103 IP addresses. The results are shown in Fig. 6. In nearly two-thirds of trials, melodies generated by hierarchical models were favored (Pearson's chi-squared test, $p = 3.96 \times 10^{-10}$ for HRNN-3L versus HRNN-1L and $p = 2.70 \times 10^{-8}$ for HRNN-2L versus HRNN-1L). In addition, subjects voted more for melodies generated by HRNN-3L than by HRNN-2L ($p = 3.38 \times 10^{-6}$)

5.5.2. Melody Score Experiment

To quantitatively measure the quality of melodies generated by different models and verify the conclusion obtained in the online experiment, we invited 18 subjects between the ages of 18 and 25 to score these melodies.

Again we generated 15 melodies with the length of 16 bars for HRNN-3L, HRNN-2L and HRNN-1L, respectively. Every subject was asked to score every melody with 5 levels: 5 the best and 1 the worst. It took each subject about 24 minutes to finish the experiment.

We calculated the average score of every melody in Fig. 6. The results verified that the two additional layers improved the quality of melodies generated by the single-layer model.

5.5.3. Music Turing Test

To compare the quality of the melodies generated by the models and the quality of melodies composed by human composers, a music "Turing test" was carried out. Only two models, HRNN-3L and HRNN-1L, were tested. We found that without chord, it was difficult for the models to generate melodies that could fool human. So chords were added as a condition of the Beat Layer and Note Layer in training and generation. Chords and primer sequences used in the generation of a melody were extracted from the same music piece in the validation set.

A total of 50 musical pieces containing 8 bars were randomly chosen from the validation set as human composed music. Then the HRNN-1L and the HRNN-3L both generated 25 melodies with the length of 8 bars. We provided subjects music pieces from these 100 examples and asked them to judge if they were composed by human. Feedback about the correctness of the choice was provided immediately after the subjects made the choice in every trial. Then the subjects had a chance to learn to distinguish human-composed and machine-composed melodies, which made it hard for the models to pass the Turing Test.

In this experiment, we collected 4185 trials from 659 IP addresses, among which 1018 music pieces were generated by HRNN-1L, 1003 by HRNN-3L and 2164 by human. As shown in Fig. 6, 33.69% of music pieces generated by HRNN-3L were thought of human composed (or real), which is higher than the proportion of HRNN-1L.

It is seen that not all music pieces sampled from the original dataset were thought to be composed by humans (only 74.95% were correctly classified). This implies that some music pieces generated by the models sounded better than human composed pieces.

6. DISCUSSIONS

In this paper, we present a hierarchical RNN model to generate melodies. We designed two high-level rhythmic features, beat profile and bar profile, to represent rhythm at two different time scales respectively. The human behavior experiment results show that the proposed HRNN can generate more well-organized melodies than the single-layer model.

There are a number of limitations with the current model. First, the dataset only contains musical pieces with 4/4 time signature. More time signatures should be taken into consideration to improve the power of the model. Second, since we quantized a bar into 16 time step, the encoding could not represent triplet or other types of rhythm. Third, the profiles presented in this paper are only about beats and bars. An appropriate representation for melody's features at larger time scales is demanded.

References

[1] 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.

[2] Elliot Waite, Douglas Eck, Adam Roberts, and Dan Abolafia, "Project magenta: Generating long-term structure in songs and storie," 2016.

[3] Li-Chia Yang, Szu-Yu Chou, and Yi-Hsuan Yang, "Midinet: A convolutional generative adversarial network for symbolic-domain music generation," in *Proceedings of the 18th International Society for Music Information Retrieval Conference, Suzhou, China*, 2017.

[4] Florian Colombo, Alexander Seeholzer, and Wulfram Gerstner, "Deep artificial composer: A creative neural network model for automated melody generation," in *International Conference on Evolutionary and Biologically Inspired Music and Art*. Springer, 2017, pp. 81–96.

[5] Hang Chu, Raquel Urtasun, and Sanja Fidler, "Song from pi: A musically plausible network for pop music generation," *arXiv preprint arXiv:1611.03477*, 2016.

[6] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] Andrzej Bargiela and Witold Pedrycz, *Granular computing: an introduction*, vol. 717, Springer Science & Business Media, 2012.

[8] Michael C Mozer, "Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing," *Connection Science*, vol. 6, no. 2-3, pp. 247–280, 1994.

[9] C-CJ Chen and Risto Miikkulainen, "Creating melodies with evolving recurrent neural networks," in *Neural Networks, 2001. Proceedings. IJCNN'01. International Joint Conference on*. IEEE, vol. 3, pp. 2241–2246.

[10] Douglas Eck and Juergen Schmidhuber, "A first look at music composition using lstm recurrent neural networks," *Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale*, vol. 103, 2002.

[11] Daniel D Johnson, "Generating polyphonic music using tied parallel networks," in *International Conference on Evolutionary and Biologically Inspired Music and Art*. Springer, 2017, pp. 128–143.

[12] Florian Colombo, Samuel P Muscinelli, Alexander Seeholzer, Johanni Brea, and Wulfram Gerstner, "Algorithmic composition of melodies with deep recurrent neural networks," *arXiv preprint arXiv:1606.07251*, 2016.

[13] Olof Mogren, "C-rnn-gan: A continuous recurrent neural network with adversarial training," in *Constructive Machine Learning Workshop (CML) at NIPS 2016*.

[14] Yuichi Yamashita and Jun Tani, "Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment," *PLoS computational biology*, vol. 4, no. 11, pp. e1000220, 2008.

[15] Wataru Hinoshita, Hiroaki Arie, Jun Tani, Hiroshi G Okuno, and Tetsuya Ogata, "Emergence of hierarchical structure mirroring linguistic composition in a recurrent neural network," *Neural Networks*, vol. 24, no. 4, pp. 311–320, 2011.

[16] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky, "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015.

[17] D Kinga and J Ba , "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.