Composing Music with Grammar Argumented Neural Networks and Note-Level Encoding

Zheng Sun*, Jiaqi Liu*, Zewang Zhang*, Jingwen Chen*, Zhao Huo†, Ching Hua Lee‡ and Xiao Zhang*§
* Sun Yat-sen University, Guangzhou, China
† China University of Political Science and Law, Beijing, China
‡ Institute of High Performance Computing, Singapore
§ Corresponding author, E-mail: zhangxiao@mail.sysu.edu.cn

***Abstract*—Creating aesthetically pleasing pieces of art, including music, has been a long-term goal for artificial intelligence research. Despite recent successes of long-short term memory (LSTM) neural networks in sequential learning, LSTM neural networks have not, by themselves, been able to generate natural-sounding music conforming to music theory. To transcend this inadequacy, we put forward a novel method for music composition that combines the LSTM with Grammars motivated by music theory. The main tenets of music theory are encoded as grammar argumented (GA) filters on the training data, such that the machine can be trained to generate music inheriting the naturalness of human-composed pieces from the original dataset while adhering to the rules of music theory. Unlike previous approaches, pitches and durations are encoded as one semantic entity, which we refer to as note-level encoding. This allows easy implementation of music theory grammars, as well as closer emulation of the thinking pattern of a musician.**

I. Introduction

The creation of all forms of art, including music, has been a long-term pursuit of artificial intelligence (AI) research. Broadly speaking, music generation by artificial intelligence (AI) is based on the principle that musical styles are in effect "complex systems of probability relationships", as defined by the musicologist Leonard B. Meyer. In the early years, symbolic AI methods were popular and specific grammars describing a set of rules drive the composition[1], [2]. These methods were later much improved by evolutionary algorithms in various ways[3], as embodied by the famous EMI project[4]. More recently, statistical models such as Markov chains and the Hidden Markov model (HMM) became popular in algorithmic composition[5], [6]. Parallel to these developments was the rapid rise of neural network (NN) approaches, which have made remarkable progress in fields like signal and image recognition, as well as music composition[7]. At present, the cutting-edge approaches to generative modeling of music are based on Recurrent Neural Networks (RNN)[8], [9], [10], [11] like the Long Short-Term Memory (LSTM) neural networks[12], [13], [14].

While RNN and LSTM networks perform well in modeling sequential data, they suffer from a few significant shortcomings when applied to music composition. The music generated is often drab and dull without any discernible theme, consisting of notes that either sound either too repetitive or too random. It is thus desirable to have a machine that can learn to generate music adhering to the principles of music theory, although that is beyond the capabilities of ordinary neural networks or usual grammatical methods.

In this work, we hence improvise an LSTM with an original method known as the Grammar Argumented (GA) method, such that our model combines a neural network with grammars. The abovementioned procedure is summarized in Fig. 2. Another novel feature of our model is our note-level encoding method. This combines the duration and pitch of each note as a single semantic entity, which is closer to how human composers think. Our results indicate that our GA model possess markedly superior performance in music generation compared to its non-GA version, according to metrics based on music theory like the percentages of notes in the diatonic scale and chords, and pitch intervals within an octave.

II. Methods

A. Note-Level Encoding

Although machine-learning methods have made significant progress in music composition, so far none has managed to closely simulate how human composers create music. In particular, human composers regard the pitch and duration of each note as attributes of a single entity, which in turn forms the building block of more complex musical motifs. By contrast, existing approaches either analyze pitches and note durations separately in separate neural networks[15], [16], or represent music as quantized time series[8], [10], [17], [18], [13], [19]. In this work, we shall attempt to more closely emulate human composers by combining the pitches and durations of musical notes into one entity, which we shall call as note-level encoding. Very importantly, this encoding allows the natural implementation of the rules of music theory as grammars, which act on notes and not merely fixed durations. This will be elaborated in Section II-C.

Our training data is derived from the MIDI sequences of 106 piano pieces by contemporary musicians like Joe Hisaishi, Yiruma, Yoko Kanno and Shi Jin. For consistency, we transpose all pieces to start with C major/A minor, only include pieces with 4/4 time signature, and retain only the melody such that the resultant music is monophonic. This entails omitting music accompaniments, grace notes and intensity changes. In particular, only the highest note, which typically carries the melody, is retained when simultaneous notes occur. This leaves us with a sequence of "Note On Events" and

"Note Off Events", which can then be directly encoded as a sequence of one-hot vectors containing duration and pitch information, like Fig. 1. Each one-hot vector consist of a 59-bit segment representing pitch semitones that occur in dataset, concatenated with a 30-bit segment representing durations from a semiquaver to a breve. Indeed, by including both pitch and duration within a single vector, our note-level encoding method enables the machine to "learn" music composition by regarding the notes as fundamental building blocks, just like with human composers.

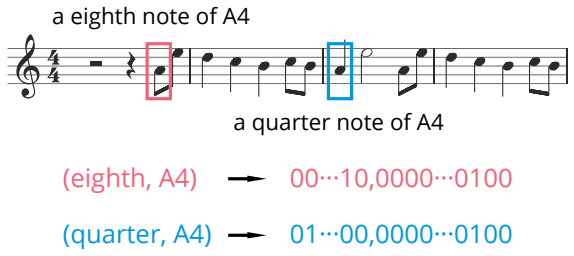

Fig. 1. The duration and pitch of each note, as extracted from MIDI files, are encoded in a single (binary) one-hot vector, This is illustrated by the quarter and eighth notes above, both of A4 pitch.

B. LSTM Neural Networks

Simple RNNs are inadequate for music composition as they do not perform well with long-term dependency due vanishing gradients[20]. This long-term dependency is necessary for understanding musical motifs which often last beyond several time steps. Our solution is to employ a more advanced type of RNNs known as a long short-term memory (LSTM) neural network, which also possess a memory cell with potentially longer-term storage of data controlled by various gates. The training details of our LSTM neural network will be discussed in Section III.

C. Grammar Argumented Method

One problem plaguing neural network approaches to music composition is that the music generated largly do not conform to basic principles of music theory. For instance, they often have too many chromatic notes (excessive chromaticity), overly large pitch intervals, and unharmonious melodies.

We propose a novel approach called the Grammar Argumented (GA) method that can significantly alleviate this problem without any manual intervention (Fig. 2). The idea is to augment the training data such that it also includes machine-generated music that perfectly satisfies the principles of music theory. To do so, the music generation is broken into two phases, the first for generating training data that perfectly conforms to criteria derived from music theory, and the second for the actual musical output. In the first phase, a GA filtering step is applied to the output, such that only melodies satisfying the three grammatical rules described below can pass (as amended data). The residual nonconforming data will be abandoned by resampling. Next the amended data will be added to the training data for retraining the machine before the second phase of generation produces the actual output.

Inspired by music theory, we put forward three specific rules for the GA filtering. The first rule is that the notes (after translation to C major) must belong to the C major diatonic scale (DIA). While occassional chromaticity (presence of chromatic notes C#, D#, F#, G#, and A#) can add extra color to a musical piece, LSTM generated music without GA argumentation contains too many chromatic notes and consequently sound random and devoid of structure. The second rule is that the pitch interval between two consecutive notes do not exceed an octave, i.e. that of short pitch interval (SPI). Large jumps in pitch usually sound disruptive and unlyrical, and we leave their artful implementation to future work. The third rule is that any three consecutive notes must belong to a triad (TRI). Triads are pairs of pitch intervals representing chords, which are of fundamental importance in musical harmony. There are four types of triads, namely the major, minor, augmented and diminished triads, each inducing a different emotional response. Triads are furthermore the building blocks of all seventh chords, which add sophistication to the composition.

III. Experiments

Our model consists of one LSTM layer and one fully connected layer. The LSTM layer includes 128 cells with input dimension 89, the length of each note's binary representation. There are 89 nodes in the fully connected layer, which is also the output layer. We build and train this model with $Keras$[21], a high-level neural networks library.

This model was first trained with the original dataset with the length of seed phrase set to 7 (notes). The loss stopped decreasing after 400 epochs, and we label the resultant weights as Orig. In the first phase of generation, we used Orig to generate 100k notes for each GA rule and obtained 5759, 5217 and 7931 amended notes respectively. Each group of amended data was then mixed with the original dataset to produce three different sets of new training data. A fourth set of new training data was obtained by mixing all these three groups of data (MIX) with the original training data. The model was then retrained with these new data, yielding four new sets of weights labeled DIA, SPI, TRI, and MIX, based on the GA rules they conform to. For statistics analysis, we used a public random seed to generate 100k notes with all five sets of weights, including Orig. Finally, the second phase of generation was performed with these five sets of weights to produce the actual output music.

IV. Results and Evaluation

We first look at a representative segment from machine's full composition generated in MIX mode, which encompasses all three GA rules. The music score of this segment is shown in Fig.3. Evidently, the machine prefers notes in the C major diatonic scale, with only one chromatic note (E-flat). There is also some rudimentary use of repeating motifs, as appearing in bars 3-4 and 10-12. The machine has also employed variations

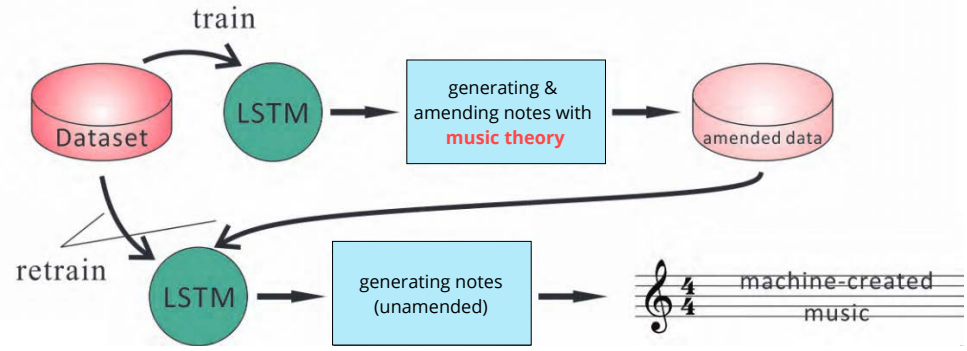

Fig. 2. The GA method. First, we train the LSTM neural network with the original dataset (top). In the first phase of generation, each note is evaluated with the GA rules from music theory, and each nonconforming note is replaced by a conforming note. The resultant amended data is next mixed with the original dataset, and used to retrain the LSTM network. This network then composes the machine-created music output in the second phase of generation.

of rhythm in bars 3, 4, 6 and 12, reminiscent of actual songs. On the whole, the segment is generally lyrical, consistent with music in the dataset.

To quantitatively evaluate the music generated, we put forward three metrics motivated by the GA rules based on music theory (Section II-C). They are the percentage of notes in the diatonic scale (p_{dia}), percentage of pitch intervals within one octave (p_{SPI}) and percentage of triads (p_{tri}). These metrics are generically applicable for all types of music, and not just those defined by note-level encoding.

A. p_{dia}

TABLE I
p_{dia} (%) OF DATASET (DS) AND OUTPUTS FROM 5 MODES

	DS	Orig	DIA	SPI	TRI	MIX
C	8.9	6.6	11.7	8.6	6.2	10.8
D	7.8	6.4	12.1	7.9	4.9	9.4
E	9.1	7.5	14.5	9.2	7.8	11.7
F	7.6	7.3	8.2	7.9	7.4	7.4
G	7.0	5.2	10.0	7.3	5.0	8.3
A	6.6	5.4	9.8	7.1	4.7	8.5
B	8.0	7.5	8.6	8.1	6.9	7.9
Total (p_{dia})	54.8	45.9	**75.0**	56.1	42.9	**64.0**

Our results show that music generated in the DIA mode indeed possess significantly more notes adhering to the diatonic scale. From Table I, which displays the percentages of each of the seven tones in C major diatonic scale, p_{dia} is 29.1 percentage points higher in the DIA mode than in Orig, where the DIA grammar rules have not been applied. Indeed the DIA GA method can significantly decrease the occurence of chromatic notes, even if the original dataset contain key changes and depart significantly from the original diatonic scale (as seen from its relatively low p_{dia}). Incidentally, the tonic note C is observed to have one of the highest occurrences, in line with expectations from more advanced music theory beyond the GA rules.

B. p_{SPI}

TABLE II
$1 - p_{SPI}$ (%) OF DATASET (DS) AND OUTPUTS FROM 5 MODES

	DS	Orig	DIA	SPI	TRI	MIX
$1 - p_{SPI}$	12.9	14.2	12.3	**9.4**	13.2	**10.2**

In Table II, we tabulate the percentage of pitch intervals within an octave for all the various mode outputs. A high p_{SPI} percentage corresponds to a more lyrical composition. Evidently, the SPI and MIX modes produces music with the highest p_{SPI}, such that there are about 30 percent fewer pitch jumps larger than one octave than that of Orig mode, whose data was generated before any GA rule has been applied.

C. p_{tri}

TABLE III
p_{tri} (%) OF DATASET (DS) AND OUTPUTS FROM 5 MODES

	DS	Orig	DIA	SPI	TRI	MIX
Major	2.3	2.2	2.3	2.4	7.9	5.8
Minor	2.1	2.0	2.3	2.1	7.7	5.6
Augmented	0.0	0.1	0.1	0.2	0.5	0.4
Diminished	0.2	0.3	0.4	0.5	1.3	1.1
Total (p_{tri})	4.6	4.6	5.1	5.3	**17.4**	**12.9**

Our results in Table III show that the music composed in the TRI and MIX modes indeed contain more triads than the other modes. p_{tri}, the percentage of triads, is computed by counting the total proportion of 3 consecutive notes assuming one of the four types of triads. The TRI mode, in particular, generates music within an almost fourfold increase in the number of triads.

Note that the music composed in MIX mode perform well under all three metrics. This suggests that the three GA rules are not conflicting, but rather are complementary ingredients for lyrical music. In addition, the MIX mode music has higher p_{dia}, p_{SPI} and p_{tri} than DS music. The high percentage of

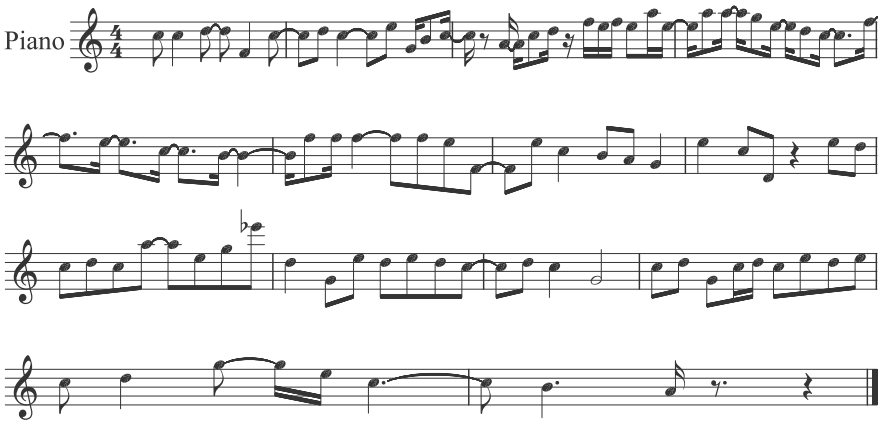

Fig. 3. An approximately 100-note segment of the machine's composition. It was generated in MIX mode, which encompasses all three GA rules.

amended data (15.9%) can account for this result and we will discuss it in future work. We emphasize that although some of the training data satisfy these metrics perfectly by construction, the final output music is generated purely by machine learning, and without human intervention.

V. Conclusion

By themselves, simple LSTM neural networks cannot generate music that are appealing from the standpoint of music theory. We addressed this problem through grammar argumented method. Since the GA filters are applied to the training data and not directly to the output, the latter is still generated by a completely bona-fide machine learning approach. Our note-level encoding method also allows a more authentic emulation of human composers, as well as provide a natural platform for implementing our grammar argumented method. The generated music generally adhere well to music theory according to the three major criteria we proposed.

Acknowledgment

The authors will like to thank Chuangjie Ren and Qingpei Liu for helpful discussions on neural networks. This work was supported by the Guangdong Science and Technology Innovation Youth Talent Program (Grant No.2016TQ03X688).

References

[1] Gary M Rader, "A method for composing simple traditional music by computer," *Communications of the ACM*, vol. 17, no. 11, pp. 631–638, 1974.

[2] Jose David Fernánd Ndez and Francisco Vico, "Ai methods in algorithmic composition: a comprehensive survey," *Journal of Artificial Intelligence Research*, vol. 48, no. 48, pp. 513–582, 2013.

[3] THYWISSEN and KURT, "Genotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition," *Organised Sound*, vol. 4, no. 2, pp. 127–133, 1999.

[4] David Cope, "Computer modeling of musical intelligence in emi," *Computer Music Journal*, vol. 16, no. 16, pp. 69–87, 1992.

[5] Moray Allan, "Harmonising chorales in the style of johann sebastian bach," *Master's Thesis, School of Informatics, University of Edinburgh*, 2002.

[6] Yushi Ueda, Yuki Uchiyama, Takuya Nishimoto, Nobutaka Ono, and Shigeki Sagayama, "Hmm-based approach for automatic chord detection using refined acoustic features," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 5518–5521.

[7] D Silver, A. Huang, C. J. Maddison, A Guez, L Sifre, den Driessche G Van, J Schrittwieser, I Antonoglou, V Panneershelvam, and M Lanctot, "Mastering the game of go with deep neural networks and tree search.," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[8] Peter M Todd, "A connectionist approach to algorithmic composition," *Computer Music Journal*, vol. 13, no. 4, pp. 27–43, 1989.

[9] 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.

[10] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," *arXiv preprint arXiv:1206.6392*, 2012.

[11] Stefan Wermter, Cornelius Weber, Wlodzislaw Duch, Timo Honkela, and Petia Koprinkovahristova, "Artificial neural networks and machine learning icann 2014," *Lecture Notes in Computer Science*, vol. 8681, 2014.

[12] Judy A. Franklin, "Recurrent neural networks for music computation," *Informs Journal on Computing*, vol. 18, no. 3, pp. 321–338, 2006.

[13] Douglas Eck and Jasmin Lapalme, "Learning musical structure directly from sequences of music," *University of Montreal, Department of Computer Science, CP*, vol. 6128, 2008.

[14] Natasha Jaques, Shixiang Gu, Richard E Turner, and Douglas Eck, "Tuning recurrent neural networks with reinforcement learning," *arXiv preprint arXiv:1611.02796*, 2016.

[15] 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.

[16] Judy A Franklin, "Recurrent neural networks for music computation," *INFORMS Journal on Computing*, vol. 18, no. 3, pp. 321–338, 2006.

[17] Kratarth Goel, Raunaq Vohra, and JK Sahoo, "Polyphonic music generation by modeling temporal dependencies using a rnn-dbn," in *Artificial Neural Networks and Machine Learning–ICANN 2014*, pp. 217–224. Springer, 2014.

[18] Douglas Eck and Juergen Schmidhuber, "Finding temporal structure in music: Blues improvisation with lstm recurrent networks," in *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*. IEEE, 2002, pp. 747–756.

[19] Qi Lyu, Zhiyong Wu, and Jun Zhu, "Polyphonic music modelling with lstm-rtrbm," in *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*. ACM, 2015, pp. 991–994.

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

[21] François Chollet et al., "Keras," https://keras.io, 2015.