Generating Globally Coherent Original Melodies Using LSTMs and music21

Dina Ginzburg

December, 15, 2017

Abstract

Machine learning algorithms have increasingly been implemented in a variety of ways within the music domain. In this paper, a Long Short Term Memory Network (LSTM) was used to solve the music-related problem of composition. The network was trained on "Ryan's Mammoth Collection of Fiddle Tunes" which was obtained from the music21 corpus. Previous attempts to generate melodies using neural networks have suffered from a lack of "global coherency". All networks were able to learn key. The 256 time step network produced melodies that were the most globally coherent, however the computer-generated melodies were not judged to be as globally coherent as the melodies in the training set.

1 Introduction

Researchers have long been interested in using machine learning techniques to compose original melodies. This task is deceptively difficult. The choice of music representation scheme, the quality and quantity of the data the network is trained on, and defining "good" melodies are all non-trivial aspects of the music-generating system which must be considered during the design process. This paper will detail one such music-generating system.

Most research on computer-generated melodies uses recurrent neural networks (RNNs), since they are specifically designed to "contemplate feedbacks units and delay operators, which allow the incorporation of nonlinearity and dynamical aspects to the model" [1]. This means that RNNs are especially good methods for learning temporal relationships in sequences .Mozer developed the CONCERT architecture, an Elman-style recurrent neural network to generate original melodies based on the genre of training data. His system transformed all of its training data to be in one key and used a complex representation scheme involving relatives pitches. Mozer was able to generate melodies that sounded pleasant to naive listeners. However, he did not find that the network was able to learn "more global phrase structure" [4]. Rather it was limited to only inducing local relationships among notes.

Later work continued to address this problem of "global coherence": Eck and Schmidbauer proposed the use of a specific type of recurrent neural network, an LSTM, since "Long Short Term Memory (LSTM) has succeeded in similar domains where other RNNs have failed, such as timing and counting, and learning of context sensitive languages" [3].This is because LSTMs are specifically designed to deal with the problem of long-term dependencies, which standard RNNs were not able to address. Their music representation scheme was also far simpler than Mozer's, simply assigning one input/target unit per note, with a 1 representing on, and a 0 representing off. Not only was

this representation simpler, it also allowed there to be no artificial distinction between notes and chords. Eck and Schmidbauer were able to show that an "LSTM is able to play the blues with good timing and proper structure as long as one is willing to listen" [3].

More recent work in music composition has introduced "chaotic inspiration" to LSTM-based systems, which involves giving the network a chaotic melody as "initial inspiration" [1]. Researchers have also attempted to define "melodiousness" in some sort of quantifiable way, despite the subjective nature of musical "quality". The system proposed in this paper will use an LSTM network, a simple binary music representation scheme, a certain amount of randomness when generating melodies, and a semi-subjective measure of "global coherency" which relies on human judges, to address some of the issues raised in the field.

2 Methods

Training data was obtained from the music21 corpus and preprocessed using music21's object-oriented toolkit. Three LSTM networks varying in input-size were trained on this data and then used to generate five original melodies each.

2.1 music21

music21 is an object-oriented python toolkit for searching, analyzing, and manipulating music scores. Scores are represented as stream objects which can be nested and contain notes, rests, and other symbolic data. All elements are timed, based on offset from a starting point. music21 also comes with a large virtual corpus containing many works of classical music, as well as a large variety of folk songs [2]. For the purposes of this project, "Ryan's Mammoth Collection of Fiddle Tunes" - containing over one thousand melodies - was chosen since it was the largest single-genre collection in the corpus. Ryan's contained melodies in varied keys which were kept as is and not converted to one single key.

2.2 Music Representation

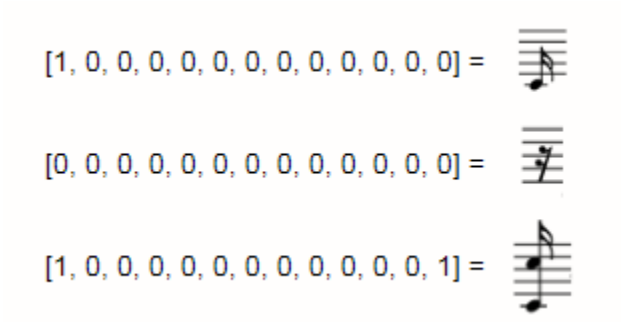

Figure 1: Each array represents the melody at a certain offset.

Each melody in Ryan's collection was transformed into a sequence of arrays. This was done using the music21 methods **Flatten** and **NotesAndRests** which reduced each score to a list of

note and rest elements, each assigned an offset value. The method **GetElementsAtOffset** was then used to check what elements were playing at each sixteenth note. The notes and rests at each offset were then encoded into an array with 13 entries, each of which represents note from C4 to C5. If an entry is set to 1, then that note is playing at that offset. If it is set to 0, it is not playing at that offset. This representation scheme is quite simple. Chords can easily be represented by simply setting more than one entry in an array to 1. However, the biggest drawback of representing melodies this way is that there is no way to discern repeating sixteenth notes and notes of longer duration. This drawback was overlooked for the sake of time, although improvements to representation will be discussed further in subsequent sections. Figure 1 shows examples of this representation. Each melody was padded with a certain amount of rests at the beginning in order to give the network an opportunity to learn how to begin melodies as well.

2.3 LSTM

RNNs address the limitations of standard feed-forward models by including loops. This allows information to persist. RNNs' chain-like structure makes it a particularly suitable model for dealing with lists and sequences. However standard RNN architectures were unable to handle long-term dependencies, or when relevant information is not recent in the sequence. This is where LSTMs come in.

LSTMs were specifically designed to address this shortcoming of RNNs. Rather than the single layer module used in standard RNNS, LSTMs have four layers in each module. These layers allow each block of the LSTM to add, remove, or pass on information from its memory. LSTMs have proven to be enormously successful at a variety of tasks and have popularized RNNs as a reliable method for learning sequences.

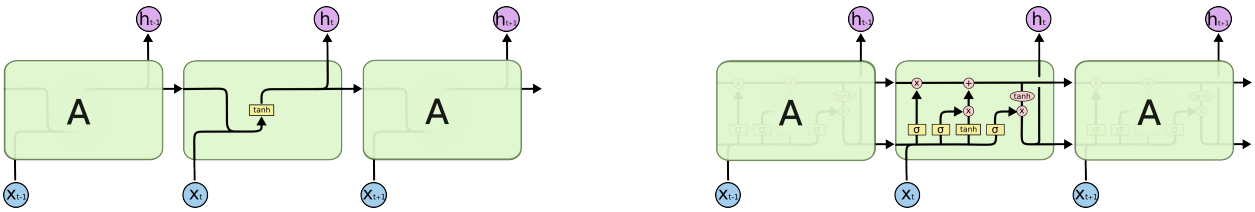

Figure 2: Left: The repeating module in a standard RNN contains a single layer. Right: The repeating module in an LSTM contains four interacting layers.

2.4 Training

The keras deep learning library was used to construct the network architectures, specifically the LSTM layer and the merge layer. Three LSTM networks were trained on the same training data. These networks only differed in the number of time-steps into the past the network considered. 64 time steps correspond to one 4/4 measure, so 64, 128, and 256 time step input sizes were selected as parameters to the network (corresponding to 1, 2, and 4 measures). Otherwise, all parameters were set to the default parameters as determined by keras. Each network was trained for 10 epochs with a batch size of 32, achieving an accuracy rates of 45%, 47.5%, and 43% respectively. In the context of music composition, the low accuracy rate should not be cause for worry. Since we want

the network to generate original melodies, we want the network to learn general rules for music composition rather than learn to predict the melodies within the training data perfectly.

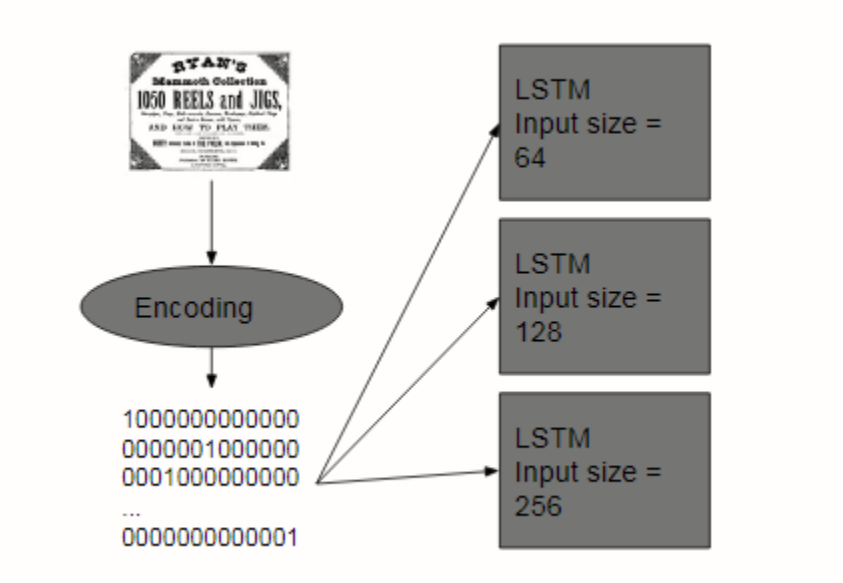

Figure 3: Three LSTMs were trained on encoded versions of melodies from Ryan's Mammoth Collection of Fiddle Tunes.

3 Experiments

Once all three networks were trained, the music-generation process began. Each network was given five sets of a starting input consisting of only rests. This was passed through the networks which generated a probability distribution for every possible next note. The next note was chosen not by choosing the note with the highest probability, but rather from the probability distribution. This allowed the generated melodies to have a certain amount of controlled "chaos". Melodies often contain patterns that are disrupted in surprising and original ways, so by incorporating randomness into the music-generation process some aspect of the creative process was also being modeled.

To address a problem mentioned earlier in Section 2.2, once melodies were generated, repeating notes were treated as the longest possible duration unless there was a measure break, in which case the notes were split apart at the measure break. The question of when to end the generated melodies was addressed simply by arbitrarily choosing a sequence length of 8 measures. This was informed by the fact that most of the fiddle tunes in the training data came in multiples of 4 measures, with 8 measures being quite common.

Every generated melody was sent to a team of two music theorists, Tai Warner and Asher Wolf, for analysis (along with five melodies from the training set). Analysts were unaware of which melodies were generated by which network. Each analyst produced a global coherency score from 1, not globally coherent at all, to 10, perfectly globally coherent, based on their research of fiddle music theory. Roughly, global coherence was judged to mean repeating patterns and musical structures/themes, as well as chord progressions that make sense given fiddle tune conventions.

4 Results

Analysts were able to correctly identify the training data melodies (example in Figure4), giving them an average global coherency score of 9.8. Although there was no significant difference between input-sizes of 64 and 128, 256 time steps led to slightly more globally coherent melodies than the other two networks. Overall, the melodies generated were middlingly globally coherent. However, some interesting features of certain melodies were noted by analysts.

Time Steps	**Average Global Coherence**
64	6.2
128	6.0
256	6.6
Training Melodies	9.8

Table 1: Average Global Coherence Scores for Each Network (based on 5 melodies each)

Figure 5 shows one melody which was labeled the analysts "favorite", despite being given a low global coherency score of 3. Although there seems to be no repetition of structures or themes within the melody, analysts noted that the melody reminded them of the style of Bela Bartok, a 20th century composer who was influenced by Hungarian folk music. Figure 6 shows another example of a melody given a low global coherency score with the analysts noting its weirdly offset beats. Figure 7 shows an example of a melody rated highly in global coherence. It exhibits a similar rhythmic structure on both lines, indicating repetition of a musical idea. Every generated melody retained a single key throughout. This indicates that the networks were able to learn key: the most basic element of global coherency.

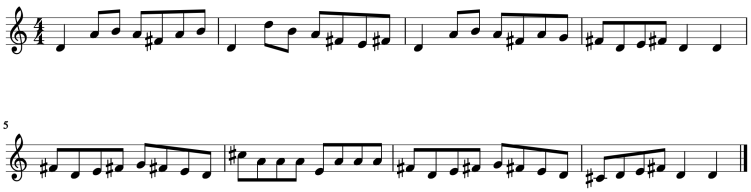

Figure 4: A melody given a global coherency score of 10. Music theorists easily identified melodies from the training set.

5 Discussion

The networks' ability to learn key indicates that LSTMs show promise for learning features of globally coherent melodies. Certainly the available literature indicates that this is true. Although global coherency does seem to be loosely correlated with LSTM input-size, more experiments are

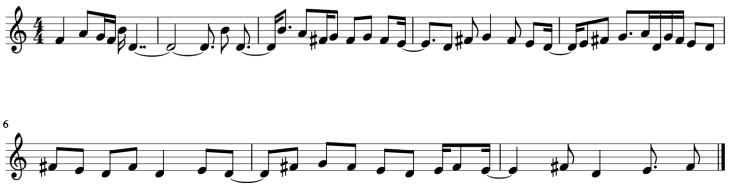

Figure 5: A melody given a global coherency score of 3, but labeled the music theorists' "favorite".

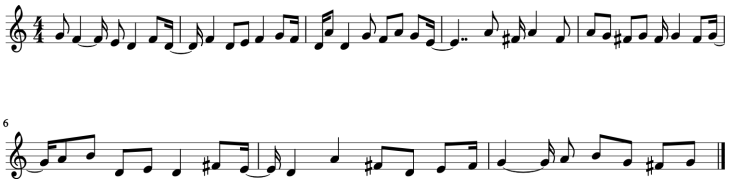

Figure 6: A melody given a global coherency score of 3. Music theorists' noted its weirdly offset beats.

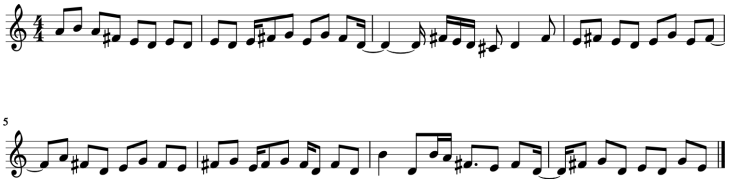

Figure 7: A melody given a global coherency score of 8. Music theorists' only noted that the melody did not resolve at the end.

needed to make any strong conclusions about the data. The computer-generated melodies were easily distinguishable from melodies in the training data, indicating that there is still much more work to be done on the system.

However, despite some disappointing results, responses from the analysts towards the computer-generated melodies were quite interesting on their own, irrespective of a melody's global coherency. In fact, Warner and Wolf found some of the least globally coherent melodies of the bunch most interesting. This suggests that there is some value to a system which does not achieve "global coherency", but nevertheless generates interesting and original melodies. Particularly in the case of "writer's block", a network that can generate unexpected melodies could spark unexpected ideas in composers.

Finding a quantitative way of evaluating the quality of generated melodies was difficult. The privileging of Western music theory and its emphasis on melody is one way to go about doing so, and in fact is how I and most work in the field has approached the task of computer music composition. However, this is a limited understanding of music and what it is about music that excites us so much. I am particularly interested in experimental music from around the world - music which explicitly breaks rules but nevertheless is able to captivate listeners. Could such an experimental, rule-breaking, machine ever exist?

5.1 Further Work

Due to time constraints, there is much further work that could be done on the system. First, the music representation scheme could be improved by adding a 14th entry in each array which would indicate if a note was ending or not. This would solve the problem of the representational equating of repeating 16th notes and notes of longer duration. Experiments could also be done by training the networks on different genres from the music21 corpus. Would networks trained on different genres produce significantly different melodies? Finally, more network architecture engineering could be done by not only tweaking parameters, but also constructing more complex network architectures. This could include more than one LSTM layer, the inclusion of standard feed-forward networks, as well as hierarchical architecture that allow for the learning of higher-level features of melodies and local relationships between adjacent notes.

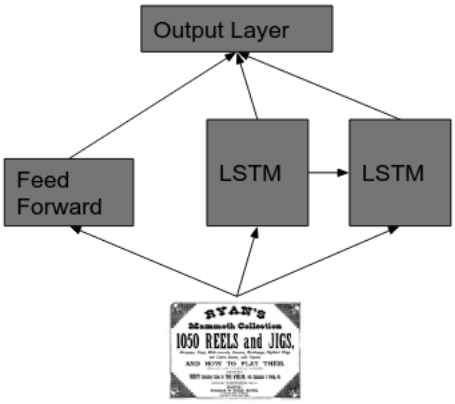

Figure 8: Potential network architecture which includes a feed forward layer as well as two LSTM layers one of which feeds into the other.

References

[1] Liang Zhao Andres E. Coca, Debora C. Correa. Computer-aided music composition with lstm neural network and chaotic inspiration. *Neural Networks (IJCNN)*, 2013.

[2] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. *ISMR*, 2010.

[3] D.Eck and J. Schmidhuber. Finding temporal structure in music: blues improvisation with lstm recurrent networks. *Neural Network for Signal Processing*, 2002.

[4] Michael C. Mozer. Connectionist music composition based on melodic, stylistic, and psychophysical constraints. *Music and Connectionism*, 1992.