COMPOSING MUSIC WITH NEURAL NETWORKS

Jan Sierpina
Institute of Informatics
University of Wroclaw
jansierpina@gmail.com

February 14, 2019

ABSTRACT

Composing of a unique music is a task generally hard for humans. In the history of music there were some geniuses that managed to create a new style or to bring some new schemes to the theory of music. Nevertheless significant part of the work of composers is just mixing already existing parts. In this paper we discuss different ideas of applying a neural network for the task of music composition.

1 Introduction

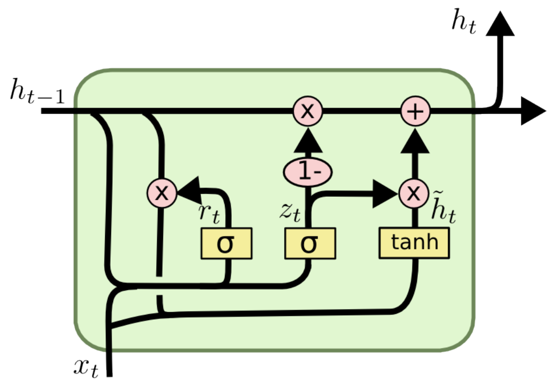
A melody can be considered as a sequence of sounds. Each sound has it's pitch and length. Between two sounds there may occur a rest which is a moment of silence. In this work we focus on the piano music, which often consists of two parts: one for the left hand, using very low pitches (bass voice) and one for the right hand, which is what people call *melody*. A short piece of music is presented in Figure-1, using the music's notation.

Figure 1: Turkish March, W. A. Mozart

2 Recurrent networks and LSTM

As mentioned before, this paper focuses on applying neural networks for music generation. If we consider music as a sequence of sounds a natural approach here would be to use a recurrent neural net (RNN). In simplicity RNN allows us to process sequences. There are dozens of papers regarding RNNs. As an introduction we recommend reading The Unreasonable Effectiveness of Recurrent Neural Networks [1] by Andrej Karpathy.

Long Short Term Memory (LSTM) networks are a special kind of RNNs. Their structure allows them to remember long term dependencies. It sounds reasonable to apply them for music generation. We hope that their structure will catch dependencies typical to songs like melody transitions. For a deeper understanding of LSTM networks we recommend reading Understanding LSTM Networks [2] by Chris Olah.



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Figure 2: An LSTM cell visualisation and equations

3 Data

The music data that is available for free usage consists mostly of MIDI files. From a MIDI file we can easily extract interesting for us components: pitches, lengths, etc. of the sounds. We chose to learn our network on works by W. A. Mozart. The schemes used by Mozart seem to be relatively simple to recognize and thus to learn.

4 First approach

Our first idea was to ignore all the features of the music except the pitches. We extracted the melody from Mozart's songs and generated a dataset of sequences with the length of 50 sounds. We taught the network to predict a successor of each sequence. The number of possible successors is limited by the number of different sounds.

5 Model

Our model consists of two LSTM layers and two fully connected linear layers. Lastly we use log softmax layer to calculate probabilities of each sound. For learning we used SGD algorithm with negative log likelihood loss.

6 Second approach

The first approach didn't take into consideration the length of sounds i. e. the rhythm of music and the harmony (chords). We decided to extract all these features from MIDI files and present data as sequences with three features: pitch, length, and offset in bar. The number of different classes to predict raised up significantly as the number of possible pitches is around 200, lengths - 20, offsets - 20. Teaching the network to predict one of 80000 classes would be hard to achieve with our small dataset. That's why we decided to teach three networks. The first network was predicting offset of succeeding sound. Second network, given sequence of triples and one offset predicts the pitch. The third network, given sequence of triples and the offset and the pitch of the new sound predicts it's length. This architectural decision has it's background in music theory: the offset (the place in time) strongly impacts the other features of the sound.

7 Results

Generated samples can be found in the files: `first.mid` and `second.mid`.

References

[1] Andrej Karpathy, http://karpathy.github.io/2015/05/21/rnn-effectiveness/

[2] Chris Olah, http://colah.github.io/posts/2015-08-Understanding-LSTMs/