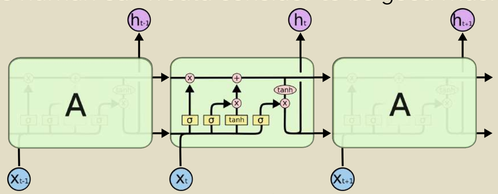
Music Composition with Machine Learning

David Kang (dwkang), Jung Youn J. Kim (jyk423), Simen Ringdahl (ringdahl)
December 11th 2018

Predicting

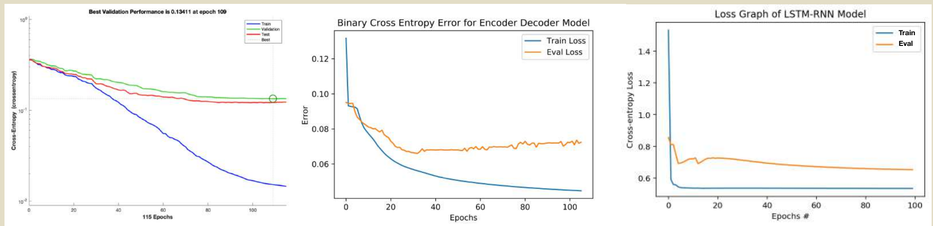
Our goal is to train a machine to **generate music**. We use classification algorithms such as various neural networks and Naive Bayes, and we use the power of Recursive Neural Nets to model sequential data in a similar way to how text is modeled. We feed the algorithm sequences of music, and train it to make new sequences that the human ear would consider to be good music.

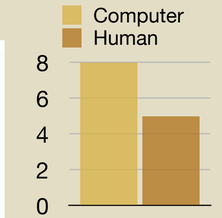


Structure of LSTM-RNN cell

Results

Model	Training Loss	Test Loss	Sample size
Naive Bayes	22.1% accuracy	3.18% accuracy	10 570 notes
LSTM RNN	0.5345	0.6526	22, 290 notes
Encoder-Decoder	0.04473	0.06773	408 700 notes





Results of survey asking whether the computer-generated music was made by humans or a computer.

Features

Although we are able to compute or estimate various features from the data, the only two explicit features are the **pitch quality** and **duration** of the notes. The pitch quality is given as a number representing which key on the piano was pressed:

TL;DR

Have you ever listened to a great song and thought "I could make even better songs"? Neither have we, so we wanted to give that job to a computer.

♫ Our algorithms learn from downloaded songs and attempt to make new songs.
♫ One algorithm looks at chords and notes separately and learns which melodies are common for a given chord.
♫ Other algorithms will predict the next key pressed in a melody based on previous keys pressed.

Do you want to listen to computer-made music? See if you can tell the difference between a musical piece made by a human and a piece made by our algorithm.

Naive Bayes

$$P(\text{note}|\text{chord}) = \frac{P(\text{chord}|\text{note})P(\text{note})}{P(\text{chord})}$$

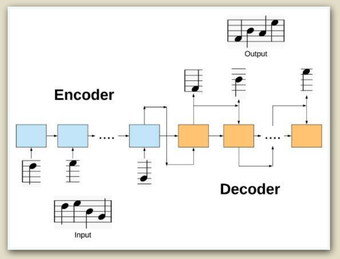
We used a Naive Bayes-like approach for determining which notes should be pressed for a certain chord. By training on many songs, we could find which keys are pressed more commonly for a given chord. Hence we could recreate a song based on these probabilities.

LSTM RNN

$$i_{(t)} = \sigma(W_{xi}^T \cdot x(t) + W_{hi}^T \cdot h_{(t-1)} + b_i)$$
$$f_{(t)} = \sigma(W_{xf}^T \cdot x(t) + W_{hf}^T \cdot h_{(t-1)} + b_f)$$
$$o_{(t)} = \sigma(W_{xo}^T \cdot x(t) + W_{ho}^T \cdot h_{(t-1)} + b_o)$$
$$g_{(t)} = \tanh(W_{xg}^T \cdot x(t) + W_{hg}^T \cdot h_{(t-1)} + b_g)$$
$$c_{(t)} = f_{(t)} \otimes c_{(t-1)} + i_{(t)} \otimes g_{(t)}$$
$$y_{(t)} = h_{(t)} = o_{(t)} \otimes \tanh(c_{(t)})$$

i and f are input and forget gate controllers which handles the parts to be added and erased in the long-term state. The input gate decides which part of g should be added to the long-term state. o is an output gate controller. Lastly, the main layer outputs g and with the results of the other three layers, LSTM cell outputs c and h which corresponds to a long-term state and a short-term state respectively to the next. For the activation functions, we use sigmoid and hyperbolic tangent functions.

Encoder-Decoder

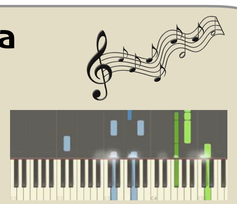


To further expand on the RNN and LSTM, we have used the encoder decoder model with gated recurrent units to try to make music sequence at a time. Instead of having one note as an input as in LSTM, the model takes in multiple notes and outputs multiple notes.

Data

Our main sources of data were:

- Online MIDI-libraries
- Youtube videos



This gave us a time series data with numbers indicating the piano key. One song can have as many as 10,000 keys pressed.

Discussion

There are two main ways in which we can interpret the data. Based on the accuracy of the algorithms, we can certainly see that they are able to predict the next note to a good degree of accuracy, as well as a distinct musical quality. The Naive Bayes approach **demonstrated harmony**, while the neural network had a higher degree of **repetition of notes**. However, music is and always have been **subjective**, so it will naturally be hard to judge.

Future

As we see that our models are promising we wish to extend the project:

- Add more parameters to our model to determine what factors contribute to make a good song.
- Find a better way of determining how successful the generated music is. This is likely to need a completely different model.

References

[1] Xenakis, I. (2001). Formalized music. 1st ed. Hillsdale, NY: Pendragon Press.

[2] Huang, Anna Cheng-Zhi (2018). Music Transformer. ArXiv e-prints. Available at https://arxiv.org/pdf/1809.04281.pdf [Accessed 3 Dec. 2018]

[3] Todd, P. (1991). Music and connectionism. Cambridge, Mass.: MIT Pr.

[4] Barreu, P. (2018). AIVA - The AI composing emotional soundtrack music. [online] Aiva.ai. Available at: https://www.aiva.ai [Accessed 10 Nov. 2018].

[5] Magenta.tensorflow.org. (2018). [online] Available at: https://magenta.tensorflow.org [Accessed 8 Nov. 2018].

[6] Hinton, Geoffrey (2014). RMSprop Gradient Optimization. Slide 29, Lecture 6. Available at: http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf

[7] Johnson, D. (2015). Composing Music With Recurrent Neural Networks. [online] hexahedria. Available at: http://www.hexahedria.com/2015/08/03/composing-music-with-recurrent-neural-networks/ [Accessed 17 Oct. 2018].

[8] Lichtenwalter, R. (2015). Applying Learning Algorithms to Music Generation. [online] Notre Dame: University of Notre Dame. Available at: https://www3.nd.edu/~nchawla/papers/IICAI09.pdf [Accessed 1 Nov. 2018].

[9] Web.mit.edu. (2018). music21: a Toolkit for Computer-Aided Musicology. [online] Available at: http://web.mit.edu/music21/ [Accessed 6 Nov. 2018].

[10] Synthesiagame.com. (2013). Synthesia, Piano for Everyone. [online]. Available at: https://www.synthesiagame.com [Accessed 2 Nov. 2018].

[11] Ba, Kingma. (2018). Adam: a Method for Stochastic Optimization. *International Conference On Learning Representations.*