Automated Music Generation

Jeong-O Jeong (djeongo@stanford.edu)

Overview

The goal of the project is to train neural networks to automatically generate music using three different models, which are 1) baseline, 2) LSTM based and 3) Reinforcement Learning based models. Classical music from Bach and Chopin are used as training data. The generated music is quantitatively compared to the training data to measure how closely the generated music resembles the training data.

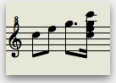
Data

Two sets of classical music data in MIDI (Musical Instrument Digital Interface) format are used: 49 Chopin Opus pieces and 49 Bach Fugue and Prelude pieces.

Data encoding

Open source library called music21 is used to parse the MIDI files. Data is encoded in the following format:

"C6 w0.5 E6 w0.5 G6 w0.75 C6 E6 G6 C7 w2.0"



where tokens starting with "w" represents the amount of wait time before the next note in quarter notes.

Data augmentation

Each of the 98 pieces of music is transposed to all possible keys by transposing to the range between -5 and +6 keys, so that the total training data size becomes 1176 individual pieces of music. This augmentation process should presumably help train the model to successfully generate music in any key instead of being tied to a specific key.

Methods

Baseline

- **Training:** Compute the transition matrix of consecutive notes and store the transition probabilities from one note to the next note.
- **Testing:** Start from a random note and sample from the transition matrix to decide which note to play next.

LSTM based model

- **Training:** Encode the notes as described in the data encoding section.The encoded notes are treated as "words", and the Word2Vec algorithm is used to generate word vectors of size 300.
- The architecture of the network is one layer of LSTM with output size corresponding to the possible number of "words". The Keras library is used to train the model.
- **Testing**: Seed the initial note sequence using existing music and let the model generate the successive notes.

Reinforcement Learning based model

The goal of this model is to augment the LSTM network with hand-crafted rewards based on simple music theory rules.The music generation is formulated as a reinforcement learning with the following setup:

- Initial-state: [random-note]
- End-state: The length of sequence of notes reach 100.
- Actions: Individual notes and wait times, e.g. "C5", "w0.5", "w0.75", etc.
- Reward: A weighted sum of music theory-based rewards and output of the LSTM network
  - Music theory rewards:
    - Reward if the new note is within the key of the music
    - Penalize if the interval between the new note and the previous note is too large, e.g. larger than an octave
    - Penalize if the same note has been repeated more than four times
  - LSTM-network reward:
    - Reward in proportional to the output of LSTM network by treating the output as p(new action | current state).

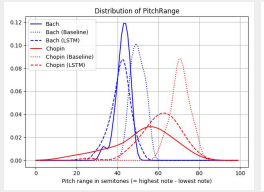
Results

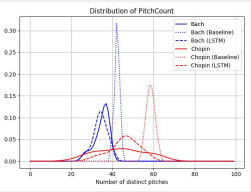
The following features are used to measure how closely the generated music resembles the training data.

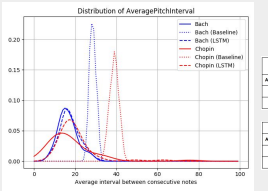
- pitch range
- pitch count
- average pitch interval

The pdf (probability density function) of each of those features of the training data and the generated data are computed to compare how closely the generated data matches the training data.

The following plots show that the LSTM-generated music closely resembles the training data in those features, which shows that it is successfully able to generate music similar to the training data.



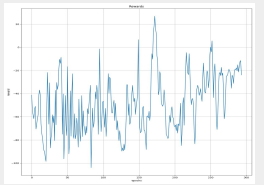




KL Divergence

	$D_{KL}(Chopin_{real} \| Chopin)$	$D_{KL}(Chopin_{baseline} \| Chopin)$	$D_{KL}(Chopin_{LSTM} \| Chopin)$	$D_{KL}(Bach \| Chopin)$
AveragePitchInterval	0.299	3.214	N/A	9.297
PitchCount	0.280	4.267	N/A	1.159
PitchRange	0.240	4.218	N/A	1.585

	$D_{KL}(Bach_{real} \| Bach)$	$D_{KL}(Bach_{baseline} \| Bach)$	$D_{KL}(Bach_{LSTM} \| Bach)$	$D_{KL}(Chopin \| Bach)$
AveragePitchInterval	0.025	2.86	10.691	1.568
PitchCount	0.079	4.287	34.424	12.952
PitchRange	0.080	4.216	2.043	10.210



The average rewards over the epochs are trending upwards, which indicates the Q-network is starting to learn the optimal policy.

Discussions

- The LSTM-based model both subjectively and quantitatively outperforms both the baseline and RL based models. The baseline model does not have memory, so it is not able to accurately predict the best note. The LSTM model performs well, because it is able to produce the most probable note based on the sequence of previous notes.
- The RL based model may not be performing well because the weighting of the music theory rewards vs. the LSTM-network reward may be suboptimal.
- The data encoding scheme proved to be very important for the LSTM network to learn. Initial scheme of quantizing and one-hot encoding the possible note values lead to LSTM predicting "rests" most of the time. The current encoding scheme yielded much better results.

URL to video: Stream: https://youtu.be/e0F97076DsE Download: https://drive.google.com/file/d/1d-iBuYA8y3MSQhVPICKrdTvtGNJpbGGR/view