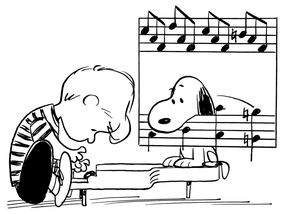
Music Generation

10 - 701 Carnegie Mellon University

Ramesh Balaji | Tanmaya Dabral | Stefani Karp



Problem Statement

Can a bot generate music? We investigate the problem of modeling and generating music.

This problem falls under the broader research umbrellas of ***generative models for sequence data*** and ***machine creativity***.

Datasets

We do not use raw audio but instead use the symbolic representations of music in 2 forms:

- MIDI: Encodes the piece as a sequence of note-on and note-off signals.
- Piano Roll: Encodes the piece as a sequence of sets of keys “depressed” at each time-step.*

We experimented on various datasets in these forms and focused our final analyses on 2 publicly available datasets:

Nottingham**, Piano MIDI

*Piano roll cannot distinguish between 2 consecutive notes of the same pitch and 1 longer note of that pitch.
**Numbers refer to the Nottingham dataset when not otherwise stated.

Representation

Capturing all information about all musical pieces in a fully general representation is extremely challenging. Different representations capture different degrees of information. We explore various representations:

- **Single pitch** at each time step
- **Multiple pitches** at each time step (map each combination of pitches to a unique encoding)
- **Events of (pitch, offset, duration)**: handles rhythm, encodes chords as sequences of events with 0 offset and equal duration
- **Binary piano roll:** handles rhythm, represents each timestep as a binary vector (1s are pressed keys)
- **Pitch-pattern piano roll:** handles rhythm, represents each timestep of a piano roll as a pair = (the lowest note currently depressed, the binary pattern relative to this note) → takes advantage of chord patterns, invariant to starting position

Models and Results

Markov Models

1st-Order Markov Chains (pitch, incl. chords)

Dataset	# of Pitches ("States")	Avg. NLL for Uniform Random Model	Avg. NLL for 1st-Order Markov Chain (TRAIN)	Avg. NLL for 1st-Order Markov Chain (TEST)
Nottingham	392	5.97126184	2.51536437	2.800220907
JSB Chorales	1695	7.43543802	2.965651723	7.220007994
Piano MIDI	6661	8.804024902	3.342980938	7.900961281
MuseData	37010	10.51894343	3.738933387	8.375618867

Higher-Order MCs (only melody)

Focused on melody/ excluded chords to reduce # of states to 33

Overfits to training data

Order	Avg. NLL (TRAIN)	Avg. NLL (TEST)
1	2.039261237	2.065925169
2	1.782213608	2.081827967
3	1.535104991	2.647063309
4	1.165478791	3.535596381
5	0.77047815	4.322571276

Hidden Markov Models

- Observations = pitch, incl. chords
- Hidden states = unknown/learned
- Training becomes slow as # hidden states grows
- Fails to capture meaningful sequential relations between notes (observations)
- For n=1...30 hidden states, **Avg. NLL hovers ~3.2 (Nottingham, Train & Test)**

General Markov Model Challenges

- Large # of different notes/states/observations, even for single-pitch-only (without chords)
- Unseen values in test set (Laplace smoothing)

Language Model (Pitch, MIDI)

LSTMs are a variant of RNN best capable of remembering long-term dependencies and are thus suited for this task. We train a language model on an LSTM (2 layers, 1250 hidden units). The vocabulary of the language consists of symbols, each representing either a pitch or a chord. During the generation phase we sample a single note/chord from the output probability distribution at each timestep.

Train NLL: 1.86 **Test NLL:** 2.02

LM (Pitch-Offset-Duration, MIDI)

We train an LSTM (3 layers, 512 hidden units) on a language model with a more diverse vocabulary. A tuple of (pitch, offset, duration) is fed as input to the LSTM at each timestep and the network is trained on the three different sequences simultaneously. In the generation phase we sample each property independently from the output distribution.

Train NLL: 3.24 **Test NLL:** 10.08

LM (Binary Piano Roll)

We train an LSTM-based language model on the binary representation of the piano roll. Note that since the binary vector may have more than one non-zero dimension, we use an MLP to generate the embeddings. The LSTM has 3 hidden layers with 256 hidden units each. We get the following results:

Dataset	Avg. NLL for Uniform Random Model	Avg. NLL (TRAIN)	Avg. NLL (TEST)
Nottingham	61.29638513	2.60526285	3.833348248
Piano MIDI	61.30573502	6.764258295	8.745947773

LM (Pitch-Pattern Piano Roll)

This language model produces two outputs: an integer representing the lowest note currently being played, and a binary vector representing the rest of the pattern, relative to the lowest note. The rest of the parameters are the same as the other piano roll model. We obtain the following results:

Dataset	Avg. NLL for Uniform Random Model	Avg. NLL (TRAIN)	Avg. NLL (TEST)
Nottingham	41.99063513	2.002010891	3.123676396
Piano MIDI	41.99290309	6.799921293	8.994680461

Data Preprocessing

We used the **Music21** Python library to help process MIDI files. Challenges:

- Developing Music21 **fluency** (e.g., ultimately catching issues/subtleties such as unexpected quantization of offset/duration, etc.)
- **Parsing** MIDI files into various representations
- **Writing** various representations as well-formed MIDI output files

In-Progress Extension

Modeling the latent vectors produced by an autoencoder: Variational autoencoders have been successfully used to generate music. However, given the small amount of data, we intend to train a vanilla autoencoder, and then try to model the distribution of the latent vectors using mixture models.

Conclusions

Quantitative: To quantitatively compare the different representations/methods, we compare the **average negative log likelihood (NLL)** per time step across all songs in the test set. However, the absolute values of the NLLs are not fully comparable across different representations. Thus, we compare with training set NLL and uniform random models.

Markov chains vs. RNNs: On Nottingham, 1st-order Markov chains performed reasonably well, but still worse (NLL 2.80) than the comparable RNN (NLL 2.02).
Higher-order MCs (on melody only) appeared to overfit to the training data (struggled to “learn” longer-term structure).

Piano roll: We compare a simple binary representation and our pitch-pattern representation. Our hypothesis is partially validated as the pitch-pattern format yields significantly better scores on the Nottingham dataset, but yields a marginally worse score on the Piano MIDI dataset. We believe this discrepancy is due to the simplistic chord accompaniment present in the Nottingham dataset.

Qualitative:

YOU BE THE JUDGE!