Music Generation with LSTMs

Joyce Xu
Stanford University
jexu@stanford.edu

Sam Xu
Stanford University
samx@stanford.edu

Eric Tang
Stanford University
etang21@stanford.edu

November 17, 2018

Abstract

Music composition is an extremely creative task, and consequently very difficult for AI models to successfully perform. We hope to generate music using audio embeddings from deep neural networks, and in doing so, further our understanding of computational creativity. We also hope to visualize and explore relationships between music embeddings using tools such as Principal Component Analysis and t-sne visualization. In this milestone, we generate baseline compositions using Logistic Regression, a traditional classification algorithm, adapted for application to the composition.

I. Introduction

Music exhibits both short-term structure, such as the relation between successive chords and notes, and long-term structure, such as a song's overall key, tempo, and melody. Previous results in music generation have seen limited success, largely due to difficulties in capturing music's complex long-term structure. Recently, however, the use of novel network architectures, such as Long Short-Term Memory networks (LSTMs) and variational autoencoders (VAEs) have opened new possibilities for music generation.[**?**].

Notably, these networks have also been used to generate vector embeddings of musical notes and chords. These embeddings can be used to train compositional models, feeding in richly structured data to generate successive notes. In our work, we begin with simple baseline Logistic Regression models, and plan to gradually incorporate these embeddings, network architectures, and an expanded corpus to improve our model's performance.

Relevant audio processing tools and datasets also improve our ability to tackle this problem. The music21 library, developed at MIT, offers a suite of tools for audio processing. The NSynth dataset contains over 300,000 annotated musical notes for over one thousand unique instruments, as well as high-quality music files.[**?**]

II. Methods

i. Dataset

We begin our exploration with a dataset of 26 Beethoven compositions, stored in the popular MIDI format. MIDI files contain information about each note's pitch, duration, and other data. This gives us a rich starting place for our exploration.

III. Preliminary Experiments

i. Baseline: Logistic Regression

i.1 Data Processing

We begin processing our data into a set of sequences, each with a one-hot encoding. In our preliminary experiments, we reduce all chords to a single note, and convert all notes to a one-hot encoded vector representation. We then extracted sequences of 10 notes, and used these sequences to predict the following note. Each training input is a vector of concatenated one-hot encodings, representing a sequence of notes. Each target output is a label representing the next note. With our 26 songs, this yields

a total of 74,593 training examples, which split into an 85/15 train/dev split.

i.2 Model

Using these concatenated vector inputs and label outputs, we then trained a logistic regression classifier to predict subsequent notes given a sequence of notes. Our baseline treats the problem as a multiclass classification problem: there are 78 possible successor notes in our database, and our model emits the most probable successor to a given sequence.

i.3 Evaluation

Evaluating on our development model, the logistic regression model obtains 28.1% accuracy in predicting the a sequence's successive note. For comparison, the most common note (C4) comprises only 3.2% of notes in our corpus.

i.4 Error Analysis

We examined the logistic regression model for one common heuristic it might use to predict the next note: simply predicting the previous note in the sequence. It appears to only do this 1.8% of the time, which reassures us that our model is (hopefully) learning something vaguely valuable?

IV. Next Steps

First, we must adapt the existing logistic regression model into a composer. By feeding the model a seed string of notes, then repeatedly asking our model to predict a subsequent note, we obtain a string of notes which serve as our model's composition.

Second, we will incorporate chord and rhythmic information into our existing model. Our current model only deals with one-hot encoding of single notes; we can handle chords by reframing our problem as a multilabel problem with inputs vectors marked as 1 for every note present. We may also explore the areas of predicting rhythm and syncopation, with a similar strategy.

Third, using the sequence extraction methods described above, we plan to next feed these one-hot encoded sequences into recurrent neural network architectures. We will likely begin with a vanilla Recurrent Neural Nework (RNN), then move to tuning a Long Short-Term Memory network (LSTM). Improving the performance of these networks will require a good deal of hyperparameter tuning as well.

Fourth, we aim to incorporate the music vector embeddings mentioned earlier, developed in previous work on audio processing and composition. We can then feed these embeddings into our models in place of the existing one-hot encodings, which should enhance our models with richer and more accessible information.

Fifth, we hope to expand our musical range beyond our existing work on the 26-work Beethoven corpus. We may augment our dataset with more classical works, or consider finding a larger corpus in the genres of Jazz or electronica.

Finally, we hope to evaluate our ultimate composition using human evaluators. Previous papers in music generation have used a small number of human evaluators to compare the performance of new models with previous models. This can be measured either by evaluators' personal preference, or how "human" they rate the music to be. We plan to recruit several human testers to compare the acoustic quality of our baseline, neural network model, and human standard.

V. Contributions

References

[Magenta] https://ai.google.research/pubs/pub4119