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

December 12, 2018

Abstract

Music composition is an extremely creative task, and consequently very difficult for AI models to successfully perform. In this paper, we explore several techniques for generating music using deep neural networks, specifically Long Short-Term Memory networks (LSTMs). We also hope to visualize and explore relationships between music embeddings using tools such as Principal Component Analysis and t-sne visualization. In a human evaluation of musical quality, our LSTM-generated compositions outperformed a composition generated using logistic regression techniques, but failed to match the quality of human-generated compositions. We briefly potential techniques to improve the quality of our LSTM compositions.

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 [1].

In this paper, we train a logistic regression model, LSTM, and VAE to take in a sequence of single notes (the melody) and predict the subsequent note in the composition. For each model, our input is the sequence of previous notes in the melody, with each note's pitch encoded as a one-hot vector. Our model then outputs a one-hot vector corresponding to the most plausible subsequent note of the melody. We use each model to compose music by feeding the model a seed sequence of notes, then repeatedly predicting the subsequent note and appending it to the composition.

II. Related Work

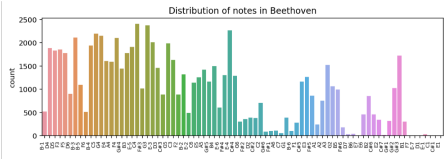
Early attempts to compose music using neural networks utilized Recurrent Neural Networks (RNNs) to predict successive notes in a composition, concatenating these predicted notes to form a melody. Unfortunately, RNN models failed to sufficiently model long-term structures in music, typically suffering from "a lack of global coherence" [2]. Early work by Eck and Schmidhuber overcame these challenges by using LSTMs to model sequences of notes, finding better long-term structure in the resulting compositions of blues melodies [3]. More recent works have used Variational Autoencoders (VAEs) to model, reconstruct, and compose sequences of musical notes. In particular, Roberts et al. find high performance using Hierarchical VAEs, which significantly outperform flat VAEs in listener evaluations [1].

Recent papers have also utilized neural network models, such as LSTMs, to extract embedded vector representations for notes in each sequence. Huang and Wu train LSTM models on several datasets of classical music and extract the learned vector embeddings from their models [4]. They then use PCA and t-sne visualization techniques to visualize the learned

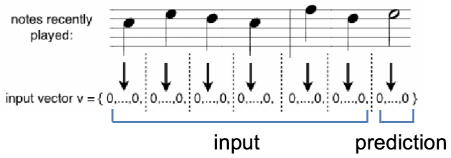
embeddings; their visualizations display clear structure in the learned embeddings, further affirming the power of LSTM models. Madjiheurem et al. take a similar approach by using a sequence-to-sequence LSTM model to predict successive chords, and taking the LSTM's vector encoding of an input chord as the vector embedding for a given chord [5]. They find that this sequence-to-sequence model outperforms bilinear and autoregressive models, while also producing more sensible embedded representations.

III. Dataset

We begin our exploration with two datasets: one of 26 Beethoven compositions, and the other of 18 Mozart compositions, both stored in the popular MIDI format. MIDI files contain information about each note's pitch and duration in a score. This gives us a rich starting place for our exploration. Both collections were sourced from a repository of classical piano scores in MIDI format [6]. We then used the music21 library, developed at music21, to process our MIDI files into sequences of pitches [7]. We tackled the simplified problem of composing melodies without chords, by taking just one pitch from each chord in the existing melodies. The resulting distribution of extracted notes is displayed below.



We then featurized the notes/chords as sequences of multi-hot vectors, where we first mapped all unique pitches in song to an index, then transformed individual notes to one-hot vector, and chords to multi-hot vectors (based on pitches).



We used a sliding window approach to capture the temporal structure in the data. Our input at each time-step is the vector concatenation of the past 10 notes, and our label is the next note. With our 26 songs, this yields a total of 74,593 training examples, which we split into an 85/15 train/dev split.

IV. Methods

i. Baseline: Logistic Regression

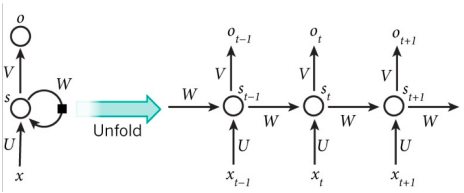
We begin with a simple logistic regression classifier trained to output the succesor note to a sequence of notes, encoded in one-hot vector form. Our baseline treats the problem as a multiclass classification problem: there are 78 possible successor notes in our database, and our model emits the most probable successor to a given sequence.

As a brief exploration of the logistic regression model's predictive power, we evaluate its predictive accuracy on our development set. The logistic regression model obtains 28.5% accuracy in predicting the a sequence's successive note on the Beethoven dataset, and an accuracy of 29.9% for the Mozart dataset.

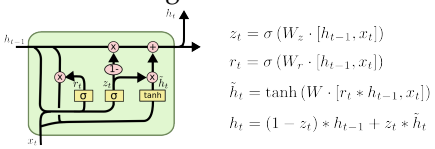
Then, to compose a score with our logistic regression model, we feed the model a seed string of notes and repeatedly ask our model to predict a subsequent note. With the subsequent note, we have a new a string of notes which can serve as our model's next generation seed.

ii. LSTM

Our next model a long short-term memory (LSTM) model, which is a refinement on the more general class of recurrent neural network (RNN) models. Recurrent neural networks are designed to capture and remember structure across time. In particular, the input to the model at each timestep is not only the new note, but the hidden state at the previous step of the model. A simplified depiction of an RNN is included below.



LSTMs are a special class of RNNs augmented to better remember long-term dependencies. Within each hidden node, there is an additional cell state and a series of gates which control how much information is updated or forgotten in the cell state. These gates are depicted in the diagram below.



In addition, our LSTM implementation incorporate attention, which improves the ability of LSTMs to encode long-term structure. While a basic LSTM solely uses the previous output vector as input to the next cell, LSTMs with attention also take the previous n output vectors as input. The model applies an attention mask to these vectors, determining the relative weight given to each previous output. By consulting these previous output vectors, models with attention are better able to capture long-term structures like repeated notes, rhythms, and song key.

V. Next Steps

i. Incorporating Chords and Rhythm

First and foremost, we intend to extend our models to handle and predict more complicated musical structures, such as chords and rhythms. Right now, 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, which we are already in the middle of transitioning to.

Similarly, because we are reducing all our training data into sequences of notes, we are currently throwing away all rhythmic information in the data. At generation time, we end up generating one note (or chord) at a time, each with the same length, which does not mimic any rhythm patterns in real music. We have two proposed solutions.

First, at each step, we could have the neural network simultaneously predict the next note and the duration of the next note, and train on both losses simultaneously (tuning the weight of the latter loss to an optimal performance). Another idea would be to discretize the space of note lengths, and at each smallest unit, predict either the next note, or predict a "hold" beat or "rest" beat. If we predict "hold," we would continue with the previous note, and if we predict "rest", there would be no output for that beat. While this idea would require more data preprocessing and significantly simplifying some input songs and rhythmic patterns, it may be much easier to learn, as we only add 2 more classes to the label space instead of a whole separate target to learn and optimize for.

ii. Improving the Model

Firstly, we intend to develop more powerful models to handle the complext structure in musical data. 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), and possibly try generative models such as Variational Autoencoders as well. Improving the performance of these networks will likely require a good deal of hyperparameter tuning and a more diverse training corpus.

Next, 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.

Third, 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.

VI. Contributions

Eric handled data preprocessing to extract note sequences from MIDI files and set up the logistic regression baseline. Joyce and Sam have run an LSTM baseline to compose using our Mozart dataset, and are currently testing other instrumental and synthesized datasets.

References

[1] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical latent vector model for learning long-term structure in music. 2018.

[2] Michael C. Mozer. Neural network composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. 1994.

[3] Eck. A first look at music composition using lstm recurrent neural networks. 2002.

[4] Allen Huang and Raymond Wu. Deep learning for music. https://arxiv.org/pdf/1606.04930.pdf, 2016.

[5] Sephora Madjiheurem, Lizhen Qu, and Christian Walder. Chord2vec: Learning musical chord embeddings. http://www.cs.nott.ac.uk/~psztg/cml/2016/papers/CML2016_paper_5.pdf, 2016.

[6] Classical piano midi page. http://www.piano-midi.de/midi_files.htm.

[7] music21: A toolkit for computer-aided musicology. http://web.mit.edu/music21/.