Machine Learning and Music Composition

Daniel Woeste
Division of Math and Sicence
University of Minnesota, Morris
Morris, Minnesota, USA 56267
woest015@morris.umn.edu

ABSTRACT

1. INTRODUCTION

2. BACKGROUND

We use this section to introduce important terminology for the rest of the paper.

2.1 Music

In this section we explain the terminology that is important to the music side of the paper.

2.1.1 *Melody*

good music definitions, need to figure out how to work them into a coherent paragraph or 2, this is super clunky to me currently

A melodic progression is the distance traveled when changing from one note to another. These distances are broken down into whole steps and half steps. Combining several of these progressions together gives us a melody. A melody is a combination of pitches and rhythms, note duration, that is considered pleasing to the ear of the listener. At any point during a piece of music, the most important part being played is considered the melody.

This may not seem very important now, but it will be important later when explaining how one of the programs tried to evaluate music automatically.

Chords during music are when several notes are being played at the same time. This can either have a pleasing affect known as harmony (consonance) or a displeasing effect known as dissonance. While most songs are tend towards the harmony side of chordal structures, dissonance is what gives the music drive and forward motion.

The texture of the music also plays an important role in the style of music that can be generated by machine learning programs. Two different styles that are used by the later discussed programs are homophonic and polyphonic textures. A homophonic texture is a piece of music that only contains one melody played at a single time. It may be played by more than one instrument simultaneously, but there is not an equally important part of the music being played at that moment. On the other hand polyphonic means that there are more than one melodic line being played at the same time. Polyphonic music is a more common style to jazz and more contemporary pieces, while some of the simpler classical music and pop music tend to favor homophonic due to its easier to grasp and understand style.

UMM CSci Senior Seminar Conference, December 2017 Morris, MN.

2.2 General Framework

All of the methods described in this paper share a similar framework that would be useful to understand. The commonality between all of the methods is that every note is generated in relation to a cache of previous notes. Meaning that every note is dependant upon a set of notes before it, whether it is only the previous note or a larger set of notes. This allows the program to determine a melodic progression for the music.

2.3 Machine Learning

Machine learning, a subcategory of artificial intelligence, is an area of computer science that focuses on programs that are able to systematically change themselves based upon a learning period. These programs are designed to alter themselves without the need for human input, these style of programs are beneficial due to the fact that they are not as strictly bound to a set of instruction. Instead, during the learning period, the programs are able to change themselves to best fit however they data demands that work. This style of data driven learning allows machine learning programs to be well suited for areas such as self driving cars, text prediction, online recommendations, and even the complex field of music.

2.3.1 *Decision Trees*

Decision trees are a structure that represent the tests and outcome of an algorithm. Every node on the decision tree represents a possible test or choice that an algorithm can make. The resulting branches from that come from that node represent all of the possible outcomes of that test. This can easily be represented by flipping a coin multiple times in a row. Every time the coin is flipped, a node is created in the decision tree. In this instance, the two nodes for the coin flip are heads and tails. For every subsequent coin flip, a new set of nodes and leaves are generated. This allows the decision tree to contain all possible outcomes for the Nth coin flip.

2.3.2 *Overfitting*

Overfitting is possible side effect that can occur with decision trees. When a decision tree is made and for an application, it can become too complex and dependant on the

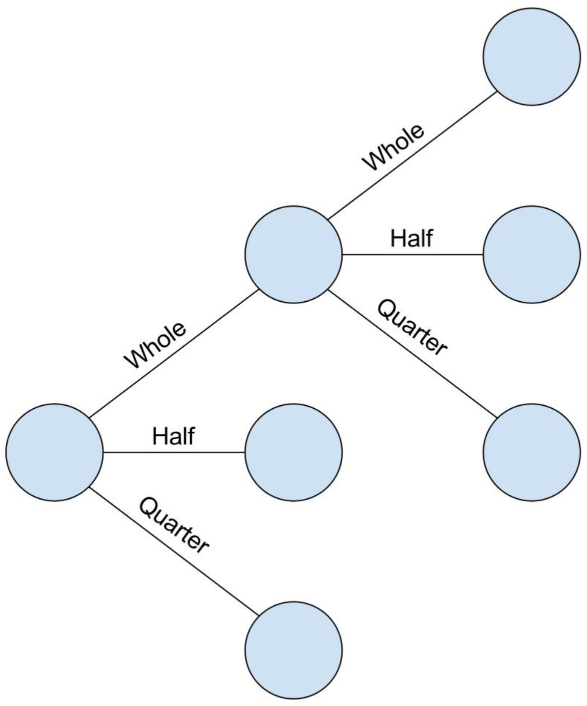

Figure 1: Descision tree example

particular set of data that it was trained upon. This occurs when a tree gains too many parameters, making the tree too specific to the data that it was trained on, or the nodes the tree branches from are inconsequential to the actual results. A consequence of overfitting happens when the tree is then tried on data different from what is was trained on. The tree loses its accuracy at predicting what the possible results may be.

2.4 Finite Automata

Finite automata, otherwise known as finite state machines, are representations of a an abstract computational model that can exist in only one of a finite number of states at a time. The transition from one state to another is caused by external inputs to finite automata. Every finite automata has a start state, often represented by the head of an arrow pointing the that state indicated as the start state, and an end state, often represented by a second circle around the last state. The finite automata listen below deviates from these notations, the start state is the state with the value of negative four, and the end state is the state with the value of 4. In this automata they can be determined by the fact that the start state does not have any recurring transitions back to it, and the final state has not possible transition to a new state once it is reached.

3. METHODS

In this section, we will be describing three separate subcategories of machine learning. The three subcategories of machine learning that we will be describing are *random forests*, *markov chains*, and *neural networks*.

First, we discuss the use of random forests. Random forests operate by generating and altering a large collection of decision trees. The generated output of the random forest is the result of the average answer of all of the decision trees in the forest. As with all machine learning a set of amount of time is required to teach the algorithm to properly generate answers. For the case of random forest, each decision tree is made on only a small part of the set of data, with overlap among all of the trees. This overlap in the data that the trees are built on helps to reduce over fitting of the trees to the data. When it comes to music we discuss a program known as ALYSIA, that uses two random forests to generate music from a given set of lyrics.

Next, markov chains function fundamentally differently than random forests. Markov chains use a state machine and a statistical models to predict the next value. Figure 1 shows a simple finite automata that generates the note length, but not pitch values for a song. The values 1, 2, and 4 represent quarter, half, and whole notes respectively. The value of -4 is a starting state that holds no value musically, other than being a place to start the song. The automata visualizes the way the markov chain uses probabilistic determination to predict what the next note or value will be. In the terms of music this means that for each state in the automata it has a percentage chance of moving to each of the other nodes in the automata. In this case, the first note will be a quarter note with an 80% chance of having a repeated quarter note, and only a 10% chance of having a half note following.

introduce the word trained instead of generated to make sense with the neural networks paragraph

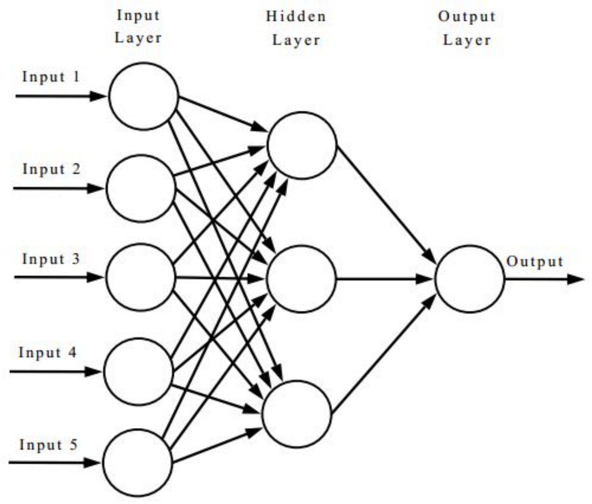

Figure 2: example of a neural network

Neural networks offer yet another approach to using computers to generate music, this time being a deterministic style of algorithm. Neurals networks are a style of machine learning that is based upon the functions of the human brain. This is done by generating many highly interconnected nodes (neurons) that are working together to solve a single problem. Similar to random forests, neural networks are trained upon a given set of data to produce a network that will produce a favorable. Unlike the previous two methods, neural networks are almost always deterministic, meaning for the same input a neural network will always

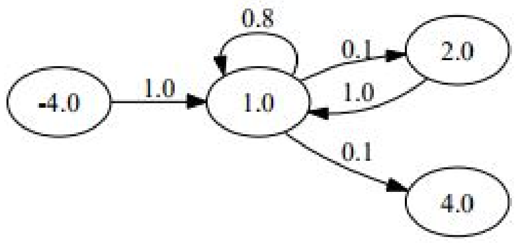

Figure 3: Finite automata detailing the Markov Generated in Automatic Composition of Music with Methods of Computational Intelligence

have the same output. This differs fundamentally from the other two methods that rely heavily on elements of chance to determine an outcome.

4. METHODS AND MUSIC

In this section, we will discuss how the previously stated methods can be applied to the composition of music with specific examples.

4.1 ALYSIA

ALYSIA, Automated LYrical Song-writing Application, is a program that uses random forests to generate songs and melodies for a given set of pop lyrics. ALYSIA contains two different random forests that work in tandem: one to produce rhythm, the other to produce pitch values. [ALYSIA] Random forests can provide some inherent benefits over Markov chains. Even though both models work from a cache of previous notes and both predict what the next note should be. Markov chains offer only a statistical probability of what the next note should be, while random forests include a reasoning to what the next note should be.

ALYSIA is able to encode a reason for the next note because, instead of relying on entirely random chance for the next note, it uses a system known as function known as feature extraction to determine the shape of the background music that the ALYSIA is trying to compose music to. The feature extractions includes parameters such as key signature, time signature, note duration, accidental, and many others. It even includes feature extraction for the lyrics that ALYSIA is trying to match as well; such as, word frequency, and even word vowel strength (value extracted from CMU-Dict v0.07 database.) This function allows ALYSIA to examine the previous few notes of the melody, if not the entire melody, in order to predict what the best next note will be.

talk about the co creative nature of ALYSIA mentioned in the paper

4.2 LSTM and RNN

Next we will examine neural networks that were used to generate an entirely different style of music, polyphonic music. Polyphonic music is a more complicated style of music that contains multiple melody lines being played all at once. This adds an added level of complication to what the neural network has to keep track of due to complexities of music in relation to music patterns and harmonic relations between the multiple lines of music. The paper by Daniel D. Johnson covers several types of neural networks including *recurrent neural networks* (RNN) based upon the *long short-term memory* system (LSTM).

A recurrent neural network is very similar to the neural networks that were described above. The difference between to two is that instead of having several layers of nodes that always progress from one layer to next, in a forward motion, recurrent networks allow for backwards travel along the network. Meaning that as a node completes an operation, the transition (arrow pointing to next node) may not point at a node in the next layer. The transition can be back to the node itself, or even to a node in the previous layers.

Long short-term memory is a specific alteration to RNN style that allows the program to store dependencies, what can be considered as previous context, throughout iterations of the network. This works type of model is particularly well suited for the generation of much becuase every note is dependant upon the notes before it to make a current melody. The long term memory of the network to determine a context of the music, better allowing the program to come up with a coherent next note. For example, an RNN with LSTM would be able to remember what key the music is in, for this instance lets say C major, and it would also be able to remember key musical features such an ascending melodic line. With this knowledge the network could determine that the appropriate following action might be a descending musical line.

4.3 Markov Chains

The program based upon markov chains in Klinger et al produces melodic lines by running the produced music through multiple iterations of the program. Each iteration of the music is focused on generating a new layer to the music. The first iteration of the music is used to generate a simple rhythmic pattern for the rest of the program to alter. Figure 3 shows a simplistic version of a markov chain generated to produce simplistic patterns of only quarter and half notes, with a melody that always ends in a whole note.

From there the resulting output from the rhythmic markov chain is then piped into a new markov chain that is used to set the pitches from the rhythm. For this the, the program could use two different styles of markov chains. These are stated as an absolute or a relative markov chain in the Klinger paper. The difference between the two styles of markov chains is that; for the absolute chain, it functions purely by chosing a note value to work with. Meaning that it chooses one of the available notes in the octave (CDEFGABC) to use as the next note value. Whereas the relative markov chain uses interval length, or the distance from one pitch to the next to determine the next note. So instead of generating an A to a B transition, the relative markov chain would output a one step transition. The benefit of the relative style is that it is less work to transpose the music from one to to another because everything is based upon step distance rather than absolute note values.

The last step to the markov model is labeled as post processing. The post processing of the music is a step that takes the already generated simplistic melody and then runs it through a last markov chain in an attempt the complicate the melody. The complication of the melody is an attempt to make the music more interesting to the listener. The post processing can be run as many times as the user feels necessary. The post processing can achieve these alteration through several methods: note splitting, one point mutation, and recombination.

insert images from paper that deal with the musical mutations

One-point mutation is process of taking the melody and traversing over it with a low chance for each note to change pitch value.

Note splitting is the similar to the one-point notation in the fact that it is still a traversal over the melody, but instead of the changing the pitch value of the note, it splits the note into an equal value of shorter duration notes. For example note splitting could change a quarter note into a run sixteenth notes.

Recombination works in an entirely different way than the previous two methods. The concept of recombination is taking two original parent melodies and combining them into a single child melody.

4.4 Citations

Acknowledgments

Fix the section heads because currently they are not organized in a way that makes sense.

the picture on descision trees needs to be smaller than it is currently

I know the markov chain graphic is in a very weird spot, but there was a weird spacing issue with it earlier so currently i have it placed there to get rid of the spacing problem