Abstract

The Music Pattern Analysis System (MPAS) is a Java system that was developed to apply the core concepts of text analysis to music. Given a song in midi format, MPAS is capable of comparing the pitch and rhythmic patterns in it to those found in a selection of libraries from a group of composers and determine to which composer the piece is most similar. MPAS was tested against four composers from separate periods of western classical music, and the songs it was given to analyze were written by one of the four. This showed how well the system was performing, as hits and misses were identifiable. For our test case, results showed that MPAS could successfully identify the composer 75% of the time.

Tyler Jenkins

Dr. Carol Browning

Senior Research

2 May 2014

Textual Analysis and Recognition Applied to Music

1. Introduction

Over the course of two semesters at Drury University, I’ve worked with Dr. Browning to develop a program in Java which applies the basic ideas of textual analysis and recognition to music. We’ve named this project the Music Pattern Analysis System (MPAS for short). After deciding upon the general direction to take with the project, we had to determine what particular aspects of a piece to analyze, what pieces we would consider, and what the software would attempt to determine about a piece based on its analysis. MPAS utilizes the jmusic library in order to read and work with midi files. It can analyze a collection of midi files in order to build up a collection of probabilities for pitch and rhythmic patterns as unigrams, bigrams, or trigrams. MPAS can then analyze a single piece of music and compare it with multiple libraries to attempt to distinguish to which the piece is most similar. It also has a method for running analysis on a collection of songs and compiling the results into one file for quicker testing and usability.

2. Text Analysis

For background on the main ideas of text analysis or natural language processing, we referenced *Artificial Intelligence: A Modern Approach*. The core concept that was used was the idea of n-gram character analysis. An n-gram as Russel and Norvig defined it is, “a sequence of written symbols of length n” (861). The prefixes uni-, bi-, and tri-, are simply referring to 1-grams, 2-grams, and 3-grams respectively. When analyzing text, the symbol sequences used for n-grams could be considered as n character, or n words. Applying it to music, we used notes for our symbols. The two aspects MPAS was designed to analyze were the two primary distinguishing features of a note: its pitch and duration.

The direction chosen for MPAS was attempting to recognize the composer of a song, or to recognize a style of music. Doing this through n-gram analysis would determine if simply collecting data on pitch and duration of notes would provide enough information to successfully do so. In essence this would allow us to see how much information about musical styling was present in the most basic structures of music.

In an attempt to help set up several distinct stylistic groups for MPAS to attempt to distinguish between, four composers from different periods of western classical music were chosen: William Byrd to represent the renaissance period, Mozart to represent the classical period, Bach to represent the baroque period, and Chopin to represent the romantic period. The end goal was for MPAS to be able to distinguish which of the composers wrote a song when presented with one written by one of these four.

3. Design

In order to achieve this goal, there are several things MPAS would need to be able to do. First, it would need to build a set of probabilities for each composer built from a library of their works and broken down in some way by pitch and rhythm patterns. After that, the gathered data could be used to find which composer an individual song is most similar to as far as n-grams of pitch and rhythm are concerned. The actual formula used to find the probability of it being a given composer is given by (initial probability)\*∏(patterni’s probability)^(number of occurrences of patterni in a given song) for every patterni.

After gathering a selection of midi files to use as the library for Bach, the first step was designing a method to run through the files and count up the totals of every unique pitch pattern and every unique duration pattern. After this, to determine the probability that Bach would write that kind of pitch or that note length, we simply divided each count by the total number of patterns examined. We then did the same for Chopin and used those two as the test case while developing the groundwork of the system.

The second step was to set up a method to perform analysis on one specific song and determine who wrote it based on the results of analyzing the libraries. To do this MPAS needs to run through one song and again count up the unique pitch and duration values and then using those counts in the above formula. This tells us by which composer MPAS believes any given song is most likely to have been written. The initial probabilities for each composer is simply set to be 1/(the number of composers). For two composers this is .5, or fifty percent is assumed, and .25 or twenty-five percent for four. These are used do to the fact that the songs we are having the system analyze are selected randomly from one of the available composers for whom MPAS has a library available.

MPAS was originally set up to use unigrams. It looked at pitch intervals, which are the number of positive or negative semitone changes reach to the following note. Over the course of its development we tried several different ways of analyzing with this data to give the best results. First we tried the idea of rounding durations to condense some of the probabilities, and truncating the pitch intervals to exclude very extreme ones, as in jumps of several octaves. Those ideas were dropped later as the library sizes for the composers were expanded. It seemed better to just leave it raw, since condensing it would actually cause us to lose some data.

The method for examining the pitch also went through several iterations. It began as pitch intervals, was changed to absolute pitch, and eventually was changed to look at the pitch of a note in relation to the key. The methods for analyzing both pitch and rhythm patterns also went from unigram to bigram to, finally, trigram analysis in the current version of MPAS. Throughout the development of MPAS many issues had to be solved, two were particularly large.

4. Issues

4.1 Underflow

A large issue that had to be circumvented when analyzing a full song was that the equation for determining probabilities involves exponential terms and products which become larger with longer songs, since they contain more patterns. This would cause underflow very quickly even using doubles to track the probabilities. Multiplying all the percentages by a constant wouldn’t solve the issue either, as if they were larger than 1, overflow would instead arise.

To try to handle this problem, we developed a few different methods. The idea to break songs up into smaller chunks was used for a while. This allowed bits that were small enough to not over/underflow to be examined and the combined results could be used to make a guess. The idea of using a bigdecimal object was also examined. But eventually, we settled on converting everything to logarithmic space for calculations, which would take much longer to over/underflow. This fix has worked for every size of song that MPAS has been given in testing.

4.2 Patterns not found

After the main components of the system were essentially completed, the remaining two composers were added in. After a fair amount of testing and bug fixes, one last major issue was uncovered in the system. This was the issue of what to do with a pattern in a song that was not found in the library for a composer. The problem was how MPAS should handle, for example, Bach’s probability when a rhythmic pattern that it had never seen Bach do was found in the song it was analyzing. Originally MPAS was set up to simply ignore the pattern and continue, which would work ok for percentages ranging 0 to 100, though still not ideal. However, they were set up to range 0 to 1, meaning every pattern found would be multiplying the probability by a number less than one, so it would be getting smaller while progressing through the song. So if a whole song contained no rhythms recognized for Bach, his probability would simply stay at .25 while the actual composer’s would shrink some as patterns were recognized for them.

Smoothing methods would be the usual solution for this. However, many involve using the vocabulary size to approximate a small chance for unrecognized patterns. The vocabulary size for a trigram of doubles is around 6.267911251143719\*1057, that is to say, it is not small. Clearly, listing all of these for four composers would greatly increase the size of the file in which the data was stored as well as likely having a noticeable effect on run times for analyzing a library. Additionally, simple smoothed approximations were likely going to be close enough to zero to effectively bring the probability for a composer down to zero whenever any single rhythm pattern wasn’t recognized, simply because of the sheer volume of options. So in order to approximate the effect smoothing would have, we simply chose an arbitrary very small (several orders less than the smallest found from examining the composer’s libraries) percent to use whenever an unrecognized pattern was encountered.

5. Results and Analysis

After all was done, the results MPAS gave came out very well. After testing 40 songs, 10 from each composer, the system successfully identified 30. The largest detractor from the hit rate was in trying to identify songs written by Chopin. For all composers apart from him, the software correctly identified 9 out of 10 songs. For Chopin it only was correct for 3 out of 10, often mistaking them for Bach or Mozart.

Interestingly, Bach and Mozart are cited as strong influences for Chopin, so that may well contribute to his low hit rate. Overall, we found the accuracy of the results somewhat surprising, since this was only from examining frequencies for rhythm and pitch patterns. No chord analysis or dynamics or any of the many other facets of music were involved. The 90% success rate on Byrd, Bach, and Mozart would suggest that each had some distinct features when compared with the others. Whereas, Chopin’s 30% success rate paired with being confused for his two main stylistic influences would suggest that MPAS functions best when comparing stylistically distinct composers.

6. Directions for Future Studies

There are several future directions the system could take and several improvements on it that could be made. For improvements, likely the largest would be to re-address the issue of smoothing, and find a more satisfying solution to implement. Additionally, MPAS could be more user friendly. Designing a more fluid user interface as well as reorganizing how the data is stored would make MPAS much easier to understand and use.

For some future directions or possible applications, MPAS can currently be used with any 2 to 4 composers or music artists. Hence, it could possibly be useful when a song is known to be from one of a small group of artists, and needs to be identified as one of them. MPAS can currently distinguish between distinct styles very well, so it could also have some applications in genre identification. It could also be improved to examine other features of music, such as chord progressions, in order to further strengthen its recognition.

There are also possible applications in the area of generating random music, trying to generate a piece in a style similar to a particular composer. Though as it stands, MPAS doesn’t account for the interplay of melody and harmony or know any of the traditional rules for composition. Thus, it would likely require some further modification before being applied to that field.

The midi files used for building and testing MPAS were all found freely online. A tremendous thanks to all the dedicated musicians who took the time to synthesize these and make them available. Here are the particular sites used for each composer: Byrd-<http://www.classicalmidiconnection.com/byrd.html>, Mozart-<http://www.classicalmidiconnection.com/mozart.html>, Bach- <http://www.jsbach.net/midi/>, Chopin- <http://www.piano-midi.de/chopin.htm> and <http://www.classicalmidiconnection.com/chopin.html>.

Works Cited

Russell, Stuart J., and Peter Norvig. Artificial Intelligence: A Modern Approach. Third ed. Harlow: Pearson Education, 2010. Print.