MELODY INFORMED MUSICAL CHORD GENERATION USING HMM

Sheng Xu
University of Rochester
Rochester, NY, 14629
sxu28@ur.rochester.edu

Albert Peyton
University of Rochester
Rochester, NY, 14629
apeyton@ur.rochester.edu

Ryan Bhular
University of Rochester
Rochester, NY, 14629
rbhular@ur.rochester.edu

ABSTRACT

This paper introduces a melody informed chord generation model using hidden markov models (HMM). We claim that by using pitch information from a melody the HMM should be able to output an acceptable chord progression based both on the note played at the particular instance of the melody and the preceding chord. We assume that one of the three notes in a triad must be the current note in the melody . We trained two transition matrices of chordal states, based on a dataset of about 1,300 pop songs and another of 32 Beethoven sonatas (classical). With this information we were able to use the viterbi algorithm in order to calculate the most likely chord progression for our input melody. This chord progression is realized in discrete time according to each observation, and is written to MIDI so to be used in a DAW.

1. INTRODUCTION

Automatic chordal harmonization of a melody is something most musicians and musical composers alike could benefit from immensely. For the less experienced musician or composer, knowledge of chord progressions and harmony can be sparse, making such a tool very useful in the creative process. Though maybe a trivial task for seasoned musicians and composers, this process can allow for such artists to draw inspiration for potential ideas, and also allow for them to focus on other aspects of creating music.

Across all western music trends emerge when it comes to the generation of harmonic structure. Music tends to move in a systematic fashion, while still allowing practitioners freedom within these constraints (generally). We propose a melody-based chord generation model using observations from the pitch information in the input melody and taking the hidden states as the chords to be generated.

We estimated 2 transition matrices for our HMM. The first of which was trained on 1,000 songs from the McGill Billboard dataset [1]. The second was estimated using the dataset provided by [2], that was used in for their research.

With these parameters found, our approach takes the inputted audio, and converts it to a sequence of observations using both the YIN pitch estimation algorithm [3] and Beat tracking by Dynamic Programming [4]. Now with the observation sequence, we use Viterbi Algorithm [5, 6] to determine the most probable chords (states) at each observation (note).

2. MUSIC THEORY

This paper aims to use prior knowledge of music theory in order to create an efficient and accurate model. All of this information can be found in [9].

2.1. Musical Key

For all of the 12 natural notes (C to B) ,and any of their enharmonic equivalents (ie. F# has the same pitch as D♭), there exists a key that begins on each note. Musical key, is a set of seven notes, and different keys of the same nature (ie. major or minor) have the same intervals between notes but in turn a different collection of pitches.

Furthermore, major and minor keys can be described in pairs with a major and minor key containing the same seven pitches but starting at different notes (ie. C major and A minor). These relative minor key occurs with the tonic (or starting pitch) nine half steps (think of this like 9 keys to right on a piano) above (Major 6^{th}) that of the major key.

2.2. Chord and Triads

The most general formation of chords is the triad. The triad consists of three notes: the root (the most identifiable pitch in the trio), the third (four or three half steps above the root for major or minor chords respectively), and the fifth (7 half steps above the root).

In four voice music it is often that the melody (highest voice) is a borrowed note from one of the chordal tones accompanying it. With this approach often times we can infer what the chord or the melody should be given the other.

3. MODEL

The basic outline of our model is as follows: Our model starts with a single inputted audio file. Pitch detection and beat tracking are then used in conjunction to determine the observed sequence of notes. We are then able to estimate the emission matrix for this sequence, assuming that all observed notes are chordal tones at each observation. We find the chord sequence using these found parameters with the found transition matrix and initial probabilities from our training. The general process can be visualized below, in Figure 1.

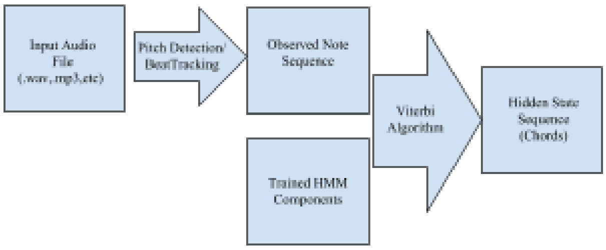

Figure 1. Model Overview

3.1. Pitch Detection

Our first task is to find the notes sequence for each input audio file. To detect the pitches (notes) of an input audio file, we adopt YIN algorithm [3]. The reasons for using YIN algorithm are as follows. First, our input audio files are monophonic, and YIN algorithm is among those with the best performance for single pitch detection. Second, the algorithm has no limit on frequency search, which means it is suitable for music ranging from the low-pitched to the high-pitched. Third, based on the famous autocorrelation algorithm, this algorithm features relatively few parameters needed to be tuned and low latency needed to be implemented.

We normalize the found pitches by converting them to their normalized pitch class. For those unfamiliar with this concept, it every instance of the same note across octaves is assigned the same numerical value. This can be visualized in the figure below.

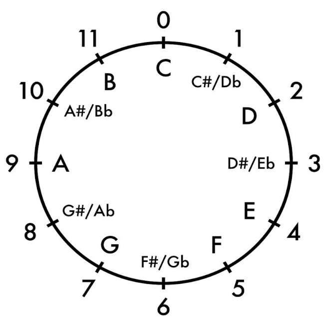

Figure 2. Numerical Representation of Musical Notes

3.2. Key Identification

With the pitch content determined from the audio sample, it is then fairly easy to identify the key of the sample. We do this with a 12 x 12 matrix which maps the probability that each pitch class belongs to each different major key. We then create a binary 12 x Frame matrix, which contains nonzero values at the present pitch class (1-12) at each frame. We multiply these matrices, resulting in a key probability x Frame matrix which we sum across each row to find the most probable key in the sample. This process can be seen in the figure below.

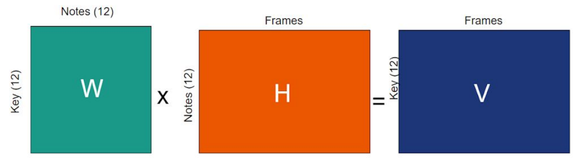

Figure 3. Key Detection Model

We then determine whether the observed pitches belong to the major or it's relative minor key based on the the distribution of pitches. In a minor key we expect to see a higher concentration of scale degree 6, 3 and 7; whereas in a major key we expect to see more of scale degrees 1, 5, and 2.

3.3. Beat Tracking

Currently, Beat tracking by Dynamic Programming [4] is used to detect the beats. This allows logical and evenly spaced chords to be generated. This algorithm uses a recursive approach to estimate the locations of onsets in the musical piece. The score **D** of each onset can be determined by:

$$D(n) = \begin{cases} \Delta(n) \;, & L = 1 \\ \Delta(n) + \lambda P_{\hat{\delta}}(n - b_{L-1}) + D(b_{L-1}) \;, & L > 1 \end{cases} \tag{1}$$

Where $\boldsymbol{\lambda}$ defines the tradeoff between onset strength and beat,b is the beat instance, Δ is the observed onset strength, and $P_{\hat{\delta}}$ is a penalty function defined by:

$$P_{\hat{\delta}}(\delta) = -\log(\delta/\hat{\delta}) \tag{2}$$

With $\hat{\delta}$ being the rough estimate of beat spacing [7].
With the found beat times, we use the pitches present at these beats to be our observed melody sequence.

3.4. Hidden Markov Model (HMM)

In short, a HMM is a way of modeling a system in which the underlying states are unknown and markovian, and the observed system at each point is a result of these hidden states at each respective point.

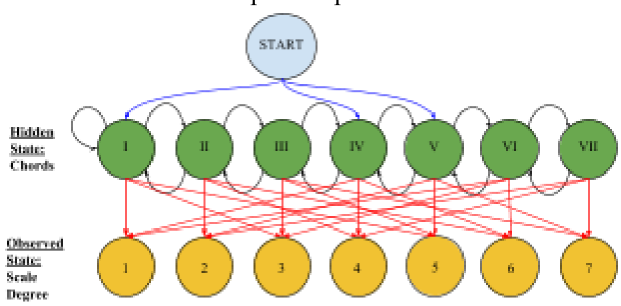

Figure 4: HMM showing connections for the strongest probabilities

3.4.1. *Music Modeling as an HMM*

As has been stated many times prior in our paper, we treat the melody as the observed state and we are seeking the sequence of hidden states (chord progression).

For our model, we normalize all pitches to be scale degrees, and similarly we normalize all chords to be roman numerals. This way, the chord progression is not influenced by key.

We also make the scale degrees relative to either the Major tonic or the tonic of the relative minor, depending on which was estimated.

3.4.2. *Estimated Emission Matrix*

Our model proposes that one of the three chord in a triad (chord) should be the note present in the input melody. This limits the options of the output chords to 3.

The emission matrix has a dimensionality of Observed states x 7. The emission matrix for each observed pitch is set to have a probability of: ½ on the chord where the pitch is the root of the chord, ⅓ on the chord where the pitch is the third of the chord, and ⅙ on the chord where the pitch is the fifth of the chord.

Because V dominant 7th chords are very present in all forms of music, we also make it so the pitch 4th scale degree has a non-zero probability of being emitted from the fifth chordal state.

We create this emission matrix simply with conditionals based on the observed note sequence.

3.4.3. *Estimated Initial Probabilities*

We estimate the initial probabilities in a similar fashion, only looking at the first observation in the melody sequence. Like the emission matrix, we pick an initial probability where this first observation is a chordal tone. However, we weight chords I, IV, and V more than their counterparts as typically a musical phrase will begin with on one of these harmonies.

3.4.4. *Estimated Transition Matrix*

We used the McGill Billboard Dataset and the Beethoven sonata Dataset in training our initial transition matrix [1][2]. Through developing a program, we have converted all the chord labels in the datasets to roman numeral notation (ie. tonic would be 1). This allows us to create a general case for all keys since musical patterns do not change based on key.

By counting the transitions of all the songs and normalizing the matrix we can generate our transition space. This transition matrix shows the probability of one chord number to another. Since in each key there are only 7 triads available (ignoring inversions), the dimensionality of our matrix is 7x7.

3.4.5. *Viterbi Algorithm*

In order to determine the best possible path for the chord progression, we use the Viterbi Algorithm. We can describe a hidden Markov chain by $\theta = (A,B,\boldsymbol{\pi})$ [6]. Where A is the initial transition matrix of chords, B is the initial emission matrix and $\boldsymbol{\pi}$ is the initial probabilities for all the states. In addition, we define Y as our observation sequence. For every specific observation sequence Y, the initial parameters A, B and $\boldsymbol{\pi}$ are estimated by using the general parameters estimated from the dataset.

3.5. VOICING CHORDS TO MIDI

The viterbi algorithm returns the chord number realized at each point of the melody. We use this information in with a number of conditionals to voice the chords. This process takes into account both the key of the sample as well as whether it is minor or major. As it stands now it returns all chords in root position. We account for whether the song is minor or major by shifting the scale degrees up by 2. We have a set of realized root position chords for both major and minor keys.

We also implemented an algorithm that articulates the chords in a simple arpeggiated style, playing the notes of the realized chords in a root-third-fifth-root pattern.

For this process we utilized functions written by Ken Schutte [10].

4. RESULTS

Something to note from our model is that the results are quite subjective. Though parameters are set for chord generation (the note in the melody and the previous chord) the results may differ from listener to listener. So for the purpose of our results we cannot show much to ground truth or raw data. Instead we can simply check these two things:

{1} The note in the melody is present as one of the chord tones.

{2} A majority of chord progressions follow conventions often seen in music

4.1. TRANSCRIBED RESULTS

Below are 2 examples of observation sequences, one major and one minor; each realized with one of the two estimated transition matrices, classical and pop.

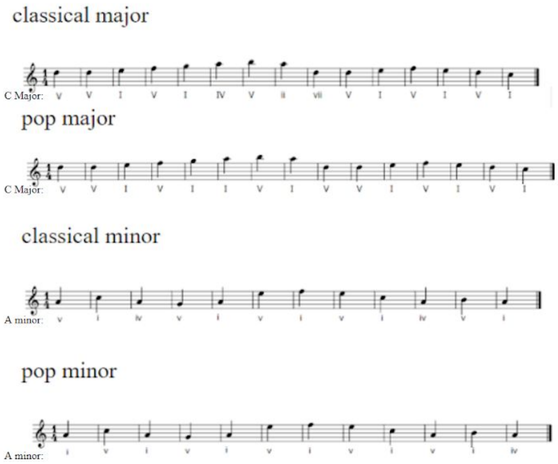

Figure 5. Melody vs the Output Chord Number

4.2. Discussion of results

As can be seen above, our algorithm does succeed in producing a meaningful chord progression built around the detected melody.

In these tests we used rhythmically simple note progressions so to focus on the calculated chord progressions.

We notice that most of the melodic content fits into the harmony of each chord. This is better illustrated in the progressions created with the classical transition matrix rather than those produced by the pop transition matrix. We believe this to be because pop music generally speaking has very few harmonic changes when compared to classical music. This is further exemplified by the fact our pop matrix typically only returns 1 and 5 chords. Though this may not be ideal, it is not inherently wrong.

That being said we see cadences in both the pop and classical results.

Our algorithm has few limitations in regard to the initial two steps of the program: finding the observed note sequence and using HMM to generate chords. When creating the MIDI notes in an arpeggiated style, the algorithm loses accuracy with longer audio inputs.

5. FUTURE WORK

5.1. Tempo Estimation

Our current program does not have automatic tempo estimation.This task is trivial but does not affect the results nor demonstration of our project, assuming the tempo of the melody is known. This is a fair assumption for this stage of the project because we have tested this on controlled examples. However, to take this program to a more applicable level, automatic tempo estimation would be needed (especially if this ever has the possibility of being implemented on audio streams).

5.2. Training on Different Music Genres

Currently we have an initial transition matrix trained on a dataset comprised of pop and classical songs [2], we would like to expand this and see how different music genre specific datasets perform relatively, as well as potentially finding a common transition matrix based on trends found in all the genres we look at.

5.3. Chord Voicing

Finally, we intend to work on how all the chords are voiced. Though in some genres all root position chords may be acceptable and even commonplace, we would like to develop some way for our algorithm to invert chords when it sees fit just to add some more realism to our progression.

6. ACKNOWLEDGMENTS

We would like to thank Professor Zhiyao Duan and all of the teaching assistants of ECE 477 at the University of Rochester. The material they taught us have given us the resources to complete this project and the course work of this class has proved to be an integral part of our model.

7. CONCLUSION

In summary, our model worked adequately in determining a chord progression for a sequence of melody notes. Both the classical and pop transition matrices yielded good results, but the results created with the classical matrix were undoubtedly better. We were successful in achieving our goal, but still recognize that our program could be expanded and improved.

8. REFERENCES

[1] John Ashley Burgoyne, Jonathan Wild, and Ichiro Fujinaga, 'An Expert Ground Truth Set for Audio Chord Recognition and Music Analysis', in *Proceedings of the 12th International Society for Music Information Retrieval Conference*, ed. Anssi Klapuri and Colby Leider (Miami, FL, 2011), pp. 633–38

[2] Chen, T. and Su, L. (2018). Functional Harmony Recognition of Symbolic Music Data With Multi-Task Recurrent Neural Networks. *Proceedings of the 19th International Society for Music Information Retrieval Conference, Paris, France*.

[3] de Cheveigné, A. and Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and

music. *The Journal of the Acoustical Society of America*, 111(4), pp.1917-1930.

[4] Ellis, D. (2007). Beat Tracking by Dynamic Programming. *Journal of New Music Research*, 36(1), pp.51-60.

[5] Muller, M. (2015). *Fundamentals Of Music Processing*. Berlin: Springer International Publishing, pp.273-293.

[6] Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), pp.257-286.

[7] Duan, Z. (2018). *Rhythm Analysis*.

[8] Duan, Z. (2018). *HMM Audio Modeling*.

[9] Roig-Francolí, M. (2011). *Harmony in context*. 2nd ed. New York: McGraw-Hill, pp.38-67, 106-128.

[10] Schutte, K. (2012). MATLAB and MIDI. [online] Kenschutte.com. Available at: http://kenschutte.com/midi [Accessed 13 Dec. 2018].