Musical Metacreation: Papers from the 2013 AIIDE Workshop (WS-13-22)

Automatic Harmonization Using a Hidden Semi-Markov Model

Ryan Groves
McGill University,
ryan.groves@mail.mcgill.ca

Abstract

Hidden Markov Models have been used frequently in the audio domain to identify underlying musical structure. Much less work has been done in the purely symbolic realm. Recently, a substantial amount of expert-labelled symbolic musical data has been injected into the research community. The new availability of data allows for the application of machine learning models to purely symbolic tasks. Similarly, the continued expansion of the field of machine learning provides new perspectives and implementations of machine learning methods, which are powerful tools when approaching complex musical challenges. This research explores the use of an extended probabilistic model such as the Hidden Semi-Markov Model (HSMM) to approach the task of automatic harmonization. One distinct advantage of the HSMM is that it is able to automatically differentiate harmonic boundaries, through its inclusion of an extra parameter: duration. In this way, a melody can be harmonized automatically in the style of a particular corpus. In the case of this research, the corpus was in the style of Rock 'n' Roll.

Introduction

The application of computers to the task of harmonic analysis has a rich history. Computers have proven to be both a powerful and flexible tool for any field, and music research is no exception. Furthermore, methods have been applied, computationally, to music which have been adapted from research in many fields. One prominent research method is through the field of linguistics. Music has shown to have many common features with natural spoken languages. The challenge of identifying triadic harmonies by employing a systematic grammar was approached as early as 1968 (Winograd). Much more recently, context-free grammars have been shown to be useful in identifying similar high-level harmonic structures in jazz music (Rohrmeier 2007), and have also been used in combination with machine learning methods to identify similarity in harmonic structures (De Haas 2009).

Researchers have also considered mathematical and geometric representations of pitch and time in order to discover patterns in harmony. Fred Lerdahl created a hierarchical system of tonality in his book Tonal Pitch Space (1988). In David Lewin's Generalized Musical Intervals and Transformations (1987), Lewin defines sets of pitches, as well as corresponding groups of transformations. These pitches coupled with their transformational groups create geometric "spaces" which define their own functional harmony. Neo-Riemannian theory is a sub-discipline of this field of transformational theory, and defines specific functions for moving from one triad to another (Cohn 1998). Julian Hook formalized the syntax and implications of these transformational spaces with his theory of Uniform Triadic Transformations (Hook 2002). Elaine Chew created her own model of the tonal system by creating a three-dimensional array of pitches in the shape of a spiral, which she uses to quantize various elements of tonality (Chew 2000).

Yet another approach is to automatic harmonic analysis is to create a rule-based system. One such system was created with over 300 rules to identify triadic harmonies from digital scores (Maxwell 1992). David Cope incorporated a machine learning method to automatically compose new first-species counterpoint harmonies given a corpus of existing compositions which exemplify counterpoint (Cope 2004). His system was able to divine a rule set from the existing corpus, and immediately apply it to create new compositions. Pardo and Birmingham sought to formalize the evaluation of chordal analysis systems, and presented their own method of labelling (Pardo 2002). Finally, the prominent systems for harmonic analysis of symbolic scores were introduced and reviewed systematically (Kroger 2008).

To date, the field which utilizes machine learning techniques to approach musical tasks is both young and burgeoning. As the prominent computational musicologist and algorithmic composer David Cope said in only 2004:

> Although significant research in machine learning has taken place in the artificial intelligence community, very little similar research has taken place in the field of music (Cope 2004).

Since then, there has been a huge interest in machine learning applications to harmonic analysis, especially in the audio realm. Specifically, probabilistic models have been used extensively to identify and predict harmony. In 2003, Daniel Ellis and Alexander Sheh borrowed techniques from the field of speech recognition to segment and label chords from audio with a Hidden Markov Model (HMM) (Ellis 2003). In 2008, Microsoft researchers approached the task of automatic accompaniment for sung pitches also using an HMM (Morris 2008). Their system, MySong, analysed and estimated the pitch of a user's vocal input, and automatically discovered the hidden sequence of chords that best fit the set of pitches sung. A year later, a comprehensive approach to the harmonic analysis problem was undertaken for symbolic input using a probabilistic system (Temperley 2009). The fundamental challenge in using machine learning is modelling the task in a way that is congruent with the underlying musical model; that is, injecting the machine learning models with musical intuition.

Audio-based models are limited with certain assumptions, which are often presented outright. Certain of these assumptions were more limiting than others. For example, one assumption listed in Microsoft's research is:

> A sufficient statistic (for the purpose of chordal accompaniment) for the notes in a measure is the fraction of the measure during which each pitch class (C, C#, D, etc.) is heard (Morris 2008).

Because of this assumption, notes are grouped together by similar pitch classes, and their durations are accumulated over each measure. Collecting notes into "buckets" or "vectors" in such a way eliminates 3 things: the duration of individual notes (since notes are grouped in corresponding pitch classes), the onset timing of the notes, and sequential order of the notes. Ellis and Sheh also reduced notes to pitch-based feature information, and avoided explicitly modelling the duration. In both cases, a vector of the different fundamental pitches present in the measure is a useful approximation, but an approximation nonetheless. This assumption was made partially in order to facilitate the process of formatting the sung input into musical input; it allowed the researchers to avoid onset detection and note segmentation in audio. The result is a model that has reduced musical intuition.

Another assumption that the group from Microsoft laid out was this:

> The location of measure boundaries in a vocal melody is known during both training and interactive use of our model. In the training phase, measure boundaries are delineated in training data; during interactive use, measure boundaries are determined by the timing of the drum beat along with which the user sings (Morris 2008).

The ramifications of this assumption is that every chord is normalized, and effectively considered to be the same length. This eliminates the duration parameter almost entirely, and loses a lot of contextual information in the process. A chord that sounds for a duration of a single beat, for example, greatly differs in its harmonic implications than the same chord sounded for a full measure. For the work of Ellis and Sheh 2003, they did not have boundaries labelled for the training phase, so their chord boundaries were also an approximation.

Thus, in developing a new model to more accurately reflect the underlying musical model that we would like our machine to learn, it is imperative to incorporate the duration and timing information. For exactly that reason, this paper describes the method for automatic accompaniment using the Hidden Semi-Markov Model. Using this model, the goal is to generate a convincing Rock 'n' Roll chord progression given an input melody, without sacrificing the fundamental attributes of chord duration and the sequential timing of notes.

The HSMM

As mentioned, the model chosen to learn the given task of automatic harmonization is the Hidden Semi-Markov Model (HSMM). This variant of the Hidden Markov Model inserts an extra parameter into the state sequence information which models the duration, or dwell time of each given state. The HSMM is compatible with modelling music, because the duration of a given chord in a chord progression will affect that chord's role in the musical structure. Since we are considering each chord to be a state in our hidden state sequence, the HSMM is appropriate. In an HSMM, a state can dwell over a certain number of observations, which means that each state can have its own sub-sequence of observations. In terms of our musical model, this means that we can consider each individual note to be an observation (rather than a bucket of notes as in audio-based methods), and that each chord can be associated with a variable number of notes. To accurately represent the duration of a chord with the state dwell time, three potential HSMMs were considered (Yu 2009).

The Explicit Duration HSMM

In the Explicit Duration HSMM, each state is considered for all its possible durations (we will use the term "duration" and "dwell time" synonymously to mean the integer number of observations that occur during that state), and the best one is chosen at transition time. A state is therefore not allowed to transition to itself, since that would be equivalent to extending the current state's dwell time (which has already been determined). This model is of explicit duration because the dwell time is considered and explicitly decided upon entering the given state. Furthermore, with Explicit Duration HSMMs, the state transition probabilities are independent of the previous state's duration. If we consider

$$a_{(i,d)} \rightarrow a_{(i+1,d)} \tag{1}$$

to be our transition probabilities for a general HSMM, then in the explicit case we would have

$$a_{(i)} \rightarrow a_{(i+1,d)} \tag{2}$$

The Variable Time HSMM

The Variable Time HSMM considers the dwell time at the current observation time for the current state, creating variability by allowing states to transition to themselves. A common method for the implementation of a Variable Time HSMM is to consider every state to be a vector of (i, d), where i is the given state, and d is its current dwell time. The state transition probabilities, then, are dependent on the previous state's duration as well. The transitions that allow for the varying dwell time consist of a state transition where

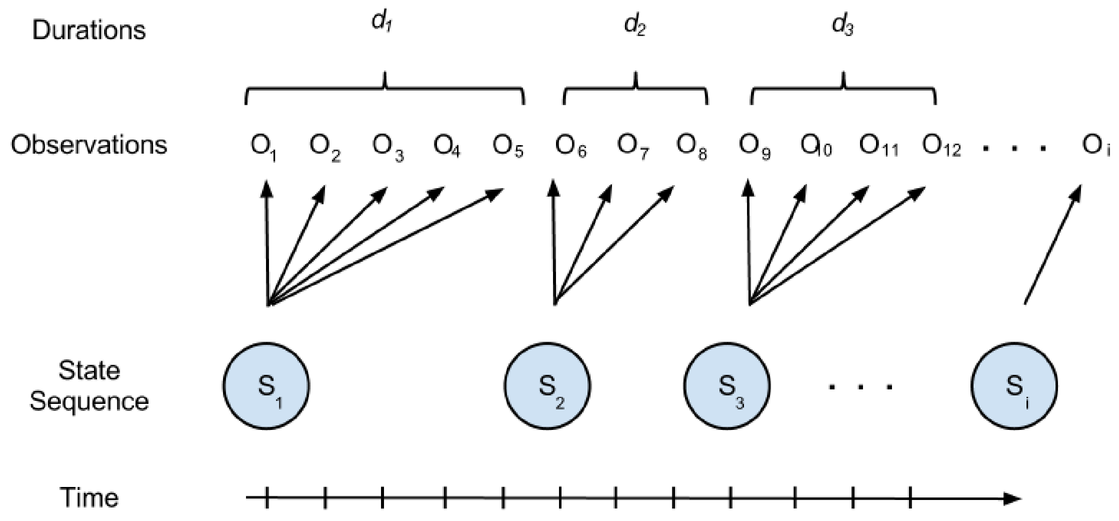

Durations
d_1
d_2
d_3
Observations
O_1 O_2 O_3 O_4 O_5 O_6 O_7 O_8 O_9 O_{10} O_{11} O_{12} . . . O_i
State Sequence
S_1
S_2
S_3
. . .
S_i
Time

Figure 1: The HSMM in graphical form

the state is unchanged, but the dwell time is incremented to $d + 1$. The two possibilities for state transitions are then:

$$a_{(i,d)} \rightarrow a_{(j,1)} \tag{3}$$

$$a_{(i,d)} \rightarrow a_{(i,d+1)} \tag{4}$$

The Residential Time HSMM

The third variant of the HSMM is the Residential Time HSMM. In this model, self-transitions are again allowed. However, similar to the Explicit Duration Model, the residential time, τ , is decided upon entry into the state. Subsequently, for every observation after entering a state at time t_r with residential time τ, the probability that the state will transition to itself is set to one for all $t < t_r + \tau$. Again, state transitions are considered independent of the previous state's residential time.

A Derived Model

To successfully include the duration of chords in the state sequence selection, a derived model is best suited. The variability of the duration is indeed a desirable feature, since it allows us to compute the current state's best duration in the forward algorithm; it allows for the consideration of transitions to a different state versus dwelling in the same state at each time index. However, the dependence of the Variable Time HSMM's state transitions on the previous state's duration expands the state set considerably, and would unnecessarily complicate the training and maximization steps, for the first incarnation of the duration variable model. That is, it is useful to isolate the first step of creating variable time states in order to test the results and consider all the implications as compared to a regular HMM. So, the derived model used is a hybrid of the Variable Time HSMM and the Explicit Duration HSMM, in which the duration of a state is variable, but the state transitions are not dependent on the previous state's duration.

Data

The corpus used to train our derived HSMM was David Temperley and Trevor De Clerq's Rock Corpus (Temperley 2011). This corpus represents a set of hand-transcribed digital score representations created by the work of 2 separate musical experts. 200 songs from Rolling Stone's list of the best 500 Rock songs of all time were selected, and both their melody and harmony were transcribed with corresponding timing information for note and chord onsets. For every song, the melody was transcribed by one expert, and the harmony was transcribed by both experts.

There were some exceptionally advantageous features of this data set and the information found in it. For one, the song data could be considered to come from the same genre of music. Since it's from the same genre of music, we can assume that the underlying musical model will remain relatively consistent, and that our generated chord sequences will represent that specific genre. Secondly, we have a complete data set—that is, there is no missing data. HMMs (and correspondingly HSMMs) are designed to be able to find the most probable hidden sequence, even if there are no examples of those hidden sequences for the data that is given. For the Rock Corpus, we have the hidden sequences for the data on which the HSMM is trained, hence the training process is much simplified. Third, the data provides all of the key signature information necessary. That allows the assumption that the key signature is known to be consistent with our training set. Lastly, Temperley provides some tools for formatting the data with a handful of different representations. One such representation is chromatic scale pitch class, which is the representation that was chosen. With such a representation, we can simplify pitches to their relative pitch class, but still include pitches that are outside of the given key signature.

An example of both an encoded melody and an encoded harmony is shown in Figure 2. For the melody file, each line represents an individual note. The four columns represent (from left to right): start time in corresponding audio (in seconds), start time in beats, absolute MIDI pitch, and key-relative MIDI pitch. For the harmony file, each line represents an individual chord. The columns represent: start time in corresponding audio (in seconds), start time in beats, roman numeral, chromatic key-relative root, diatonic root (e.g., VI = 6), current key tonic (in pitch class), and absolute root (also in pitch class).

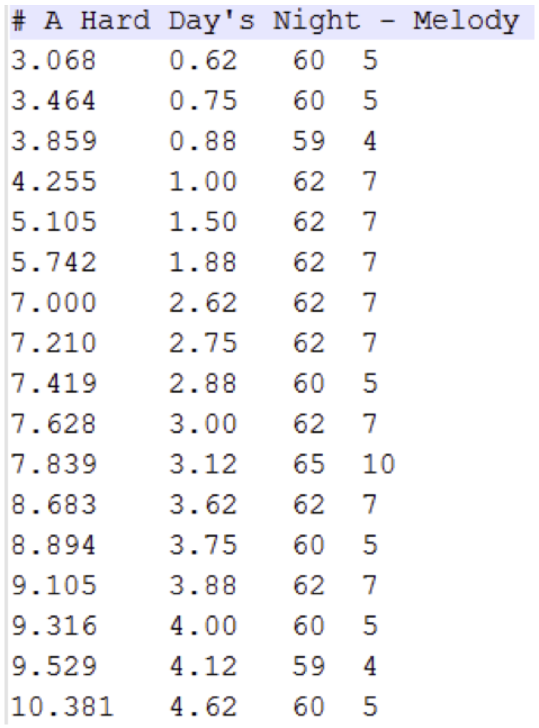
Implementation

Pre-processing

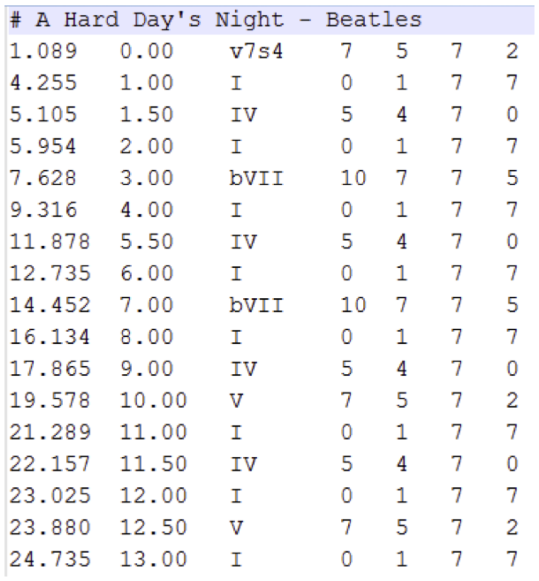
As mentioned above, our observations now represent individual notes. This differs from audio-based models in which notes are simplified to frame-based pitch information. This representation allows us to model the different possible durations of our chord states, relative to the notes. In order to digitize a melody into distinct note observations, each note was reduced to its key-relative pitch class representation.

This creates an assumption that notes an octave apart represent the same functional pitch (i.e. its class) for any given note. This assumption is made frequently when trying to model musical concepts, and although it is a simplification, the advantages of grouping similar notes together outweigh the loss of musical precision. One other possible observation was added to represent the lack of any notes played over a given chord state.

The states are represented by key-relative roman numerals. In general, these roman numerals stand for triads that are relative to the given key signature. The transcribers considered a higher level of detail than was necessary for the effective encoding of degree triads—alterations, inversions, and additions to the fundamental triads were included. When every alteration was considered to be its own state (as well as an "End" chord which represented the end of the song), a total of 143 distinct chords were found in the set of 200 songs. Because repetitions of chord phrases were counted as new and unique observations, it is possible that a single song with a large number of a particular chord (especially if that chord were uncommon) could sway the overall distributions. It was necessary, then, to group these chords by their triadic equivalence, in order to find the common functional style from the data set as a whole. One such grouping was, for all



```
# A Hard Day's Night - Melody
3.068    0.62   60  5
3.464    0.75   60  5
3.859    0.88   59  4
4.255    1.00   62  7
5.105    1.50   62  7
5.742    1.88   62  7
7.000    2.62   62  7
7.210    2.75   62  7
7.419    2.88   60  5
7.628    3.00   62  7
7.839    3.12   65  10
8.683    3.62   62  7
8.894    3.75   60  5
9.105    3.88   62  7
9.316    4.00   60  5
9.529    4.12   59  4
10.381   4.62   60  5
```



```
# A Hard Day's Night - Beatles
1.089   0.00   v7s4   7   5   7   2
4.255   1.00   I      0   1   7   7
5.105   1.50   IV     5   4   7   0
5.954   2.00   I      0   1   7   7
7.628   3.00   bVII   10  7   7   5
9.316   4.00   I      0   1   7   7
11.878  5.50   IV     5   4   7   0
12.735  6.00   I      0   1   7   7
14.452  7.00   bVII   10  7   7   5
16.134  8.00   I      0   1   7   7
17.865  9.00   IV     5   4   7   0
19.578  10.00  V      7   5   7   2
21.289  11.00  I      0   1   7   7
22.157  11.50  IV     5   4   7   0
23.025  12.00  I      0   1   7   7
23.880  12.50  V      7   5   7   2
24.735  13.00  I      0   1   7   7
```

Figure 2: Example of Melody (left) and Harmony (right) file for an individual transcribed song, specifically "A Hard Day's Night" by the Beatles (Temperley 2011).

roman numerals, all additions were removed, except for the dominant. The reasoning behind this was that there existed many tonic triads with dominant 7ths or 9ths. These dominants have implications of the key tonality, and thus could be considered to be fundamentally different from a simple tonic triad. Dominant additions include the dominant 7, 'd7', as well as the dominant 9, 'd9'(which implies the dominant 7, as a musical rule). These additions were simply reduced to a 'd' following the roman numeral. All inversions were also removed, which is somewhat common in popular harmonic representations. After the state space was grouped in such a manner, the state set was reduced to 57 unique chords, or states.

Learning

The implementation of the derived HSMM is only slightly more complex than that of a regular HMM. For a detailed overview of the workings of an HMM, see Rabiner (1989). Since the state set is complete (there is no missing data), training the data is just a matter of creating the state transition matrix, the observation-per-state matrix, and the duration-per-state matrix. The state transition matrix is created by accumulating the transitions from one state to every other state, for every state, and then normalizing that by the starting state's total occurrences. For the state of C Major, for example, the transition matrix would include a row of transition probabilities which represent the percentage of times C Major transitioned to each other state, normalized between 0 and 1. Similarly, each place in the observation-per-state matrix represents the probability of a particular observation occurring over a particular state. The duration-per-state matrix stores the probabilities for each state to dwell for each of the possible durations.

Because of the format of the transcriptions in Temperley and De Clerq's Rock Corpus, the duration possibilities were actually much more varied than expected. This was due to the fact that chords which transitioned to themselves were transcribed with only one chord symbol. For example, if a song consisted of 3, 4-beat bars that sustained on the **I** chord, it would look identical to a single bar of 12 beats sustained on that **I** chord. Because of this, sometimes there were situations where the most of the song was a single chord (as was the case in Nirvana's "All Apologies", for example), and there was a string of more than 50 observations over that single chord. Therefore, it was necessary to implement a maximum number of observations over which a single state may dwell. A rational approach would be to consider each downbeat to be the start of a new state. However, the downbeat information is not automatically accessible; although the majority of time signatures were 3 or 4 beats with either eighth or quarter note tatums, certain songs contained anacruses or time signature changes which obscured or shifted the downbeats. In order to estimate the maximum number of notes per chord, the total number of notes in the data set were accumulated, as well as the total number of chord changes. Using these numbers (60,799 and 17,853, respectively), the average number of notes per chord transition was found to be 3.41 notes. This is relatively close to one note per beat, if 3 or 4 beats are considered to be the majority of measure lengths, and chords are considered to sustain for an average of one measure. Given this, it is possible to make an educated guess as to the maximum chord duration, in terms of note onsets. A maximum of 16 note observations per chord would allow for vocal phrases with a rapid succession of notes, as well as limit the length of a chord to 16 beats for regions where the rate of note occurrences is closer to average—that is, roughly one note per beat. In summation, the number of total possible observations over a single state was restricted to 16.

Once the model is trained on the given data set, it is possible to compute the most probable chord progression given any melody, using the Viterbi algorithm. In an HSMM, the Viterbi algorithm must be modified to look back and iterate through the possible durations for each states, and to choose the most probable duration for the considered state, given the observations and the previous state. The problem with this distinction is that the HSMM is no longer a realtime-capable algorithm. One of the principles of Hidden Markov Models is that the current state is only dependent on the current observation and the previous state. In order to keep this realtime capability, an approximation was made. Thus, when computing the probability of transitioning from one chord to another, if the chord stays in the same state (call it state i transitioning from time d to $d+1$), then instead of multiplying by a transition probability, the following product was used:

$$\frac{P(i \mid d+1)}{P(i \mid d)} \tag{5}$$

Because of this, at any given point in the most probable sequence, a chord will consider changing states based on the transition probability, and will consider staying in the same state based on the current state's probability of dwelling for time $d+1$.

Conclusion

Future work

There are still many limitations in the model. For example, timing/duration is indeed represented, but the timing of the chord sequences is an abstraction of real musical timing (since states are relative to the *number* of notes). Since the notes currently do not represent duration, a link to the real musical timing of the song is lost. However, this could be remedied by separating the note observations into categories of duration ranges, and encoding that in the note representation. It would also be plausible to model the transitions between notes, in order to better predict note observations, although the model would need to be modified more drastically to accommodate such a change.

A simple, but meaningful improvement would be to add the dependence of a state's transition probability on the duration of the previous state. This would again expand the state space, but there is good musical motivation to categorize chords based on the duration over which they sound. For example, if a chord is only played for 1 beat, versus say 8 beats, the notes that we could expect to be played over that chord would surely change, as well would the chord we could expect to follow said chord. However, more data would most

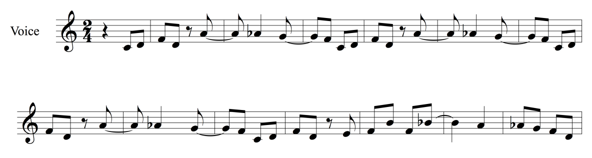


Figure 3: A composed input melody.

likely be needed in order to train a model with different durations of chords as different states, since for every possible state duration we consider, we would be increasing the state space by a multiple of 57.

Another musically intuitive improvement would be to remove the repetition of chord progressions from the training data. Currently, if a verse is repeated, and the chord progression is unchanged, we will treat that chord progression as if it is a new part of the song, when really we have already considered that situation.

We could also remove the assumption that we know the key signature in which the input melody is played. If that were the case, we could find the best possible hidden sequence for all the 12 possible keys, and would simply need to select the key signature of the sequence with the highest probability.

Lastly, it would be very useful to run the Expectation Maximization (EM) routine on new data with missing information. For example, if we had an effective melody extraction method for audio-based rock songs, we could use EM to further train our model on these new song melodies, even though we would not have the chord progression information. By using a melody extraction technique we could potentially do it in an automated way.

The fact that the HSMM remains realtime makes it a viable candidate for realtime instruments. If one were to connect this harmonizer to a MIDI instrument, or to a vocal pitch-tracker, the chord choices could be observed immediately in an interactive way. Similarly, parameters could be created for certain properties of the HSMM, for example the chord complexity of the states, or the frequency of chord changes.

Results

To test the HSMM on the data set, 5-fold cross validation was used. Since there were no parameters to optimize (because the data set is complete), each fold consisted of simply a training set (made up of 4/5 of the data, randomly selected), and a testing set (with the one remaining partition). Each fold used a different testing set, and trained the model on the remaining training set.

Using this model, the Viterbi function was computed over the sequence of notes representing an entire song, and the most probable chord sequence was compared with the song's original, solution chord sequence. For every generated chord in the most probable chord sequence, the correct predictions were simply tallied and compared to the total number of observations in the song. Using this as the benchmark, the correct chord was predicted an average of 25.42% for every song over the five folds. The correct boundary between chords was discovered 10.63% percent of the time (that is, when the solution chords had a boundary, the predicted chord also had a boundary). Considering that there are 57 possible chords to choose from in the state set, this percentage is a considerable increase over a simple random selection (which would equate to 1.75% expected accuracy for randomized guessing).

Furthermore, an example melody was composed specifically to demonstrate its ability to generate a stylistic harmonization from any melodic input. The melody was composed in the key signature of C with a Dorian mode centrality, and was designed to include every possible chromatic note, so as to test the chord generator's flexibility. The melody is shown in Figure 3.

Given that melody, the HSMM generated the most probable chord progression. That progression is shown in Figure 4. There are a few interesting things to point out from the generated chord progression. For one, the V/V, or D major is chosen to accompany the D minor blues lick which starts on the pickup of the third bar. It is not uncommon to use the minor blues scale on the tonic of a major chord in Rock. Another interesting aspect of this progression is the repetition. The opening melodic phrase repeats three times exactly, and then a fourth time with a modified ending. The HSMM generates a repeated phrase as well. However, the phrase it generates is of a longer period than the melodic phrase—it begins on the first melodic phrase, and doesn't repeat until the third melodic phrase. It is quite interesting that it would naturally create a longer phrase structure in the harmony. Lastly, some of the chords which sound for only a single note in length have rather dissonant notes occurring in the melody. I believe this is a situation in which the chord sequence probabilities slightly override the probabilities of the note observation being found over that chord.

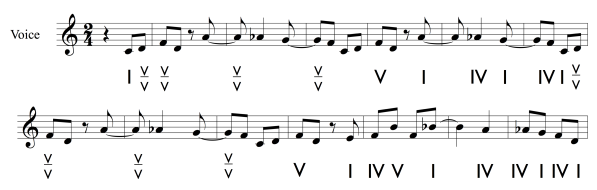

Voice
I V/V V/V V/V V/V V I IV I IV I V/V
V/V V/V V/V V I IV V I IV IV I IV I

Figure 4: A composed input melody, harmonized with the chord sequence generated from the HSMM.

Conclusion

Although simplistic, the HSMM proves to be able to harness the style of a particular corpus. The ability to model duration and maintain the sequence of notes in the underlying representation is advantageous, and allows for more specific duration details to be further incorporated into the model. The Rock 'n' Roll corpus of only 200 songs was shown to be sufficient in training the model on Rock 'n' Roll harmonies. Given the metrics of cross-validation, and the viable concrete example, the HSMM has shown to be a candidate for realtime, automatic harmonization, and in the least is certainly a good candidate for future music research.

References

Elaine Chew. *Towards a Mathematical Model of Tonality*. PhD thesis, University of Southern California, 2000.

Richard Cohn. Introduction to neo-riemannian theory: A survey and historical perspective. *Journal of Music Theory*, 42(2):167–180, 1998.

David Cope. A musical learning algorithm. *Computer Music Journal*, 28(3):12–27, 2004.

David Cope. An expert system for harmonizing chorales in the style of J.S. Bach. *Understanding Music with AI: Perspectives on Music Cognition*, pages 355–63, 1992.

W. Bas De Haas, Martin Rohrmeier, Remco C. Veltkamp, and Frans Wiering. Modeling harmonic similarity using a generative grammar of tonal harmony. In *Proceedings of the 10th International Society for Music Information Retrieval Conference*, 2009.

Julian Hook. Uniform triadic transformations. *Journal of Music Theory*, 46(1/2):57–126, 2002.

P. Kroger, A. Passos, M. Sampaio, and G. de Cidra. Rameau: A system for automatic harmonic analysis. *Proceedings of the 2008 International Computer Music Conference*, pages 273–81, 2008.

Fred Lerdahl. Tonal pitch space. *Music Perception: An Interdisciplinary Journal*, 5(3):315–49, 1988.

David Lewin. *Generalized Musical Intervals and Transformations*. New Haven and London: Yale University Press, 1987.

H. J. Maxwell. An expert system for harmonic analysis of tonal music. *Understanding Music with AI: Perspectives on Music Cognition*, pages 335–53, 1992.

D. Morris, S. Basu, and I. Simon. Exposing parameters of a trained dynamic model for interactive music creation. *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008.

Brian Pardo and W. Birmingham. Algorithms for chordal analysis. *Computer Music Journal*, 26(2):27–49, 2002.

Lawrence R. Rabiner. A tutorial on hidden markov models and selected application in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–86, 1989.

Martin Rohrmeier. A generative grammar approach to diatonic harmonic structure. In *Proceedings of the 4th Sound and Music Computing Conference*, pages 97–100, 2009.

Alexander Sheh and Daniel PW Ellis. Chord segmentation and recognition using em-trained hidden markov models. *ISMIR 2003*, pages 185–191, 2003.

David Temperley. A unified probabilistic model for polyphonic music analysis. *Journal of New Music Research*, 38(1):3–18, 2009.

David Temperley and Trevor De Clerq. A corpus analysis of rock harmony. *Popular Music*, 30(1):47–70, 2011.

Terry Winograd. Machine models of music. Chapter Linguistics and the computer analysis of tonal harmony, pages 113–153. MIT Press, Cambridge, MA, USA, 1968.

Shun-Zheng Yu. Hidden semi-markov model. *Artificial Intelligence, an International Journal*, 174:215–43, 2009.