Automatic Chord Recognition from Audio

Alex R. Emmons
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA 56267
emmon046@morris.umn.edu

ABSTRACT

Automatic chord recognition from audio is used in the area of *Music Information Retrieval* (MIR) to document and categorize music. In addition to providing harmony to music, chords also provide a way to describe the harmony of a piece. In almost every chord recognition system the audio signal is represented by a *Pitch Class Profile* (PCP), which measures the intensity of energy in each of the frequency regions where musical notes occur [4]. Some systems perform what is known as preprocessing before generating a PCP, to get rid of unwanted frequencies in the audio file. The next step, known as pattern matching, is to assign chord labels by matching the harmonic features to a set of chord models. In this paper we will discuss these processes in greater detail and compare the results of three research cases, each of which uses a different chord recognition system.

Keywords

Automatic chord recognition, hidden Markov models, pitch class profile, signal processing

1. INTRODUCTION

A chord is a set of tones played simultaneously. A chord progression is a sequence of chords over time and is what describes the harmony of a piece [2]. Automatic chord recognition is the process of extracting a chord progression from an audio file. These chord sequences are used by musicians as lead sheets (summaries containing chords, melody, and lyrics) as well as by researchers for tasks such as key detection, genre classification, and lyric interpretation. Performing chord analysis by hand is time consuming, prone to human error, and requires two or more trained experts. This is what makes automatic chord recognition an important area of research [3].

The two main steps of automatic chord recognition are feature extraction and pattern matching. Feature extraction is the process of extracting useful information from audio files, and pattern matching is how chord labels are applied to that data.

There are many challenges encountered by systems that process audio signals. There are background noises, percussion instruments, and other unwanted tones in audio recordings. It is also difficult to distinguish when chords change and to line these points up exactly with the beat. Preprocessing helps eliminate unwanted information from the audio files before or during the feature extraction step, depending on the system. An overview of one of the chord recognition systems looked at in this paper can be seen in Figure 1.

Chord recognition systems have been improving and becoming more usable in recent years. This paper will compare three different systems that use a variety of techniques in each step of the process. By looking at the components of the highest performing systems, we will determine the most effective methods used in each step.

2. BACKGROUND

In order to explain the process of automatic chord recognition, some general information about feature extraction and pattern matching is needed.

2.1 Feature Extraction

The first step of generating a chord progression from audio data is processing the signal to extract harmonic features. Feature extraction is a fairly simple process, but can be made more complex by introducing optimization steps to increase accuracy [3]. Preprocessing is one of these steps, performed before generating *Pitch Class Profile* (PCP), which is a representation what notes are present over time.

2.1.1 Preprocessing

In Figure 2 the light areas show where frequencies have been detected, and the dark areas show empty space. It is clear that frequencies other than just the chord tones have been detected because the light areas are not solid white and the dark areas not solid black. The goal of preprocessing is to reduce as much of this background noise as possible from the audio file, in an effort to provide a smooth and clear PCP.

Another issue is that musical instruments produce a series of harmonics at higher and lower frequencies than the tone that is played. These tones, called overtones, can confuse feature extraction techniques and need to be removed.

UMM CSci Senior Seminar Conference, December 2014 Morris, MN.

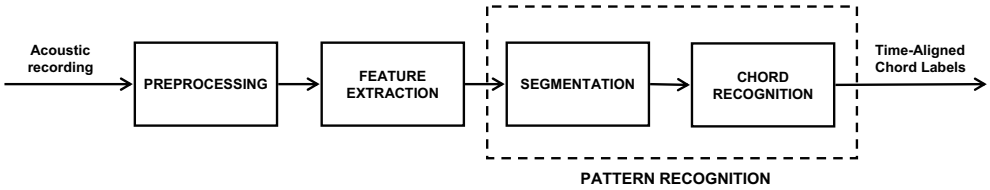

Figure 1: Overview of the chord recognition system used in [4].

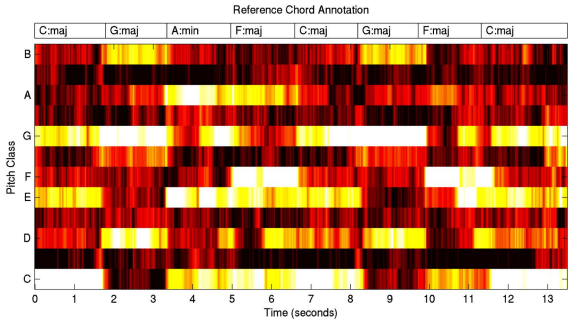

Figure 2: A typical chromagram, or PCP, generated from the opening to *Let It Be* (Lennon/McCartney). Pitch class (chroma) at time t is shown by the intensity, or lightness at that point. The true chord progression (simplified) is shown above for comparison.

Explain two types of preprocessing, background spectrum and overtone removal. Explain types and how they work and give more info. Explain more about overtones and how they are detected, give more examples of 'other tones'. the sum of sinusoidal tones of integer multiples of the fundamental frequency. Add a figure.

2.1.2 *Pitch Class Profile*

The pitch of a note is measured in two dimensions - height and chroma (pitch class). Height tells which octave a note belongs to, and chroma tells where a note stands within the octave (the name of the note). Height is not a factor in determining chord type because two notes that are an octave apart have the same chroma value. A chromagram, or PCP, is a 12-dimensional vector representation of chroma, representing the intensity of each of the twelve semitones in the chromatic scale, over a period of time. An example of a common PCP, along with the actual progression, can be seen in Figure 2 [3]. For over a decade PCP has been the most popular way to represent harmonic features for chord recognition. Most new approaches are variations or refinements of this approach [1]. Modern systems can have many more steps in converting audio to PCP including tuning correction, which compensates for music that is not tuned to standard pitch $A4 = 440\ Hz$, and beat-synchronization, which calculates the average pitch between beats to get rid of changes caused by noise and other transients. [3].

Describe them or some of them. Decide between using pitch class and chroma. More detail in paper 1 under feature extraction

2.2 Pattern Matching

Decide between pattern recognition and pattern matching.

Almost all chord recognition systems use PCP or some other chroma-based feature extraction technique. What differentiates these systems is the mechanism used to label chords.

Define chord model.

Generating the chord model against which the PCP will be matched can be done in one of two ways: manually using musical knowledge, or stochastically, by deriving it from real-world music. In a manual system the chroma values being sounded are compared against pre-defined chord templates.

Define these better.

These two methods are compared by Cho and Bello in [1]. Stochastic chord models are more sophisticated and complex. HMMs used to be the method of choice, but many recent systems prefer Gaussian mixture models [1].

Add sentance describing stocastic chord model system after manual system. Get rid of more sophisticated and complex.

2.2.1 *Hidden Markov Models*

A *hidden Markov model* (HMM) is a statistical model which describes a finite set of states, in this case chords. Transitions between these states are governed by a set of transition probabilities that describe the likelihood of transitioning from one to another [1]. The Viterbi algorithm then finds the most likely sequence by evaluating the joint probability of the output and all possible chord sequences. HMMs are used in a wide variety of pattern recognition environments such as speech, handwriting, and gesture recognition, as well as in bioinformatics.

Need to re-write this section from draft. Include figures for models. find first instance of HMM.

2.2.2 *Gaussian Mixture Models*

Gaussian mixture models (GMMs) are used in many modern chord recognition systems. A GMM consists of a distribution of weighted Gaussian components that represent descriptions of each chord in the training data.

Need a better definition

To train these models the PCP from the training data is transposed to all C-based chords (C-major and C-minor). These normalized chords are used to train the C-major and C-minor models, and then re-transposed to the remaining 11 keys [1].

	Type	Sample Rate	FFT Length	Lifter	HPS Ratio
FV1	FB	44100	32768	yes	5
FV2	PCP	11025	4096	no	1
FV3	PCP	44100	32768	no	1
FV4	PCP	44100	32768	yes	5

Table 1: Feature Vectors used in isolated chord recognition.

Need more here. REWRITE

2.2.3 *Support Vector Models*

3. RESEARCH CASES

This paper looks at three research cases that involve automatic chord recognition. All of the cases use three common steps in the process: feature extraction, preprocessing, and pattern recognition. Here we will give an overview of each system and the datasets that were used.

Decide between study and case.

3.1 Case 1: Effects of Proper Signal Processing

The first study [4] uses a chord recognition system that begins with a preprocessing block, followed by feature extraction, where PCP is calculated. After this, the signal is segmented along predicted chord boundaries, and then chord labels are assigned to each segment. An overview of this system can been seen in Figure 1.

There are two stages in the preprocessing block, to address background noise and overtones. The first step, Homomorphic Liftering, is a method of separating out the frequencies of musical tones from background and system noise. This is done by finding strong frequency peaks in areas corresponding to the pitch range of the notes. Frequencies above and below a specified range can also be removed to reduce noise. The second step, known as the *Harmonic Product Spectrum* (HPS), is a method which emphasizes frequencies when their overtones are present. HPS is calculated by compressing the spectrum by factors of 1 to R and multiplying the resulting compressed spectra. The resulting output energy is then summed according to pitch class and a PCP is created.

The first step of pattern matching for this system is chord segmentation, where the audio signal is segmented at the boundaries where chords change. This can be difficult because some chords are not played all at once, the notes are played in sequence and held. To find the points where change has occurred the PCP is analyzed frame-by-frame to find significant change in pitch-class content.

The next and final step in this system is assigning chord labels to the segments. Given an instance of the PCP for a region of the audio, the most probable chord label is picked from a set of training PCPs. In this study the use of GMMs and SVMs is compared [4].

3.1.1 *Datasets*

In this experiment *MIDI* (Musical Instrument Digital Interface) recordings were used to create time aligned labeled audio. Two datasets were used: one of isolated chords synthesized on piano and strings, and one of continuous single-instrument audio synthesized on piano.

The isolated chord dataset consisted of 7790 chords and

	Label given in:		
Chord Label	**DS1A**	**DS1B**	**DS1C**
Major	Major	Major	Major
Minor	Minor	Minor	Minor
Major 7	-	Major	Major 7
Minor 7	-	Minor	Minor 7
Dom. 7	-	Major	Dom. 7
Dim.	Dim.	Dim.	Dim.
Full Dim.	-	Dim.	Full Dim.
Half Dim.	-	Dim.	Half Dim.
Augmented	Aug.	Aug.	Augmented
Sus. 4	-	-	Sus. 4
7 Sus. 4	-	-	7 Sus. 4

Table 2: Chord complexity levels used in [4].

inversions, where the notes are stacked in a different order. Three chord complexity levels were tested, labeled DS1, DS2, and DS3 (seen in Table 2). DS1 involved the four common triad types (major, minor, augmented, and diminished), across all roots. All other chords were not used. DS2 included variations of the 7th chord (major 7th, minor 7th, dominant 7th, fully diminished 7th, and half diminished 7th), and in DS3 all 11 different chord types that the system could recognize were used.

Four *Feature Vectors* (FV), or combinations of methods used for feature extraction, were compared on each of these complexity levels, seen in Table 1. FV1 started with preprocessing using homomorphic liftering with a lower cutoff of 30Hz and upper frequency cutoff of 4kHz, then computing HPS with a compression ratio R = 5. Instead of PCP, in this case an 84 dimensional vector was used (representing 7 octaves with 12 notes per octave). The hypothesis was that this would provide more information because the octaves weren't flattened down to one. FV2 had a lower sampling rate and *Fast Fourier Transform* (FFT) length, and no preprocessing was performed on the audio signal. This was to match a previous system that they were comparing against. FV3 increases the sample rate and FFT, but still leaves out preprocessing. FV4 introduces preprocessing using the same homomorphic liftering and HPS as FV1.

The continuous single-instrument audio dataset consisted of 50 hymn verses selected from the Trinity Hymnal, with 40 used for training and 10 used for testing. FV4 was used from the isolated chord experiment [4].

3.1.2 *Results*

For the isolated chords dataset, this system was trained using chords synthesized on a piano, and was tested on chords synthesized on both piano and strings. The dataset was randomly divided with 80% of the chords used for training and 20% for testing. Five of these training and testing sets were created and the recognition rates from these

Feature Vector	DS1A	DS1B	DS1C
FV1	83.68	61.85	57.24
FV2	90.33	82.44	82.26
FV3	**91.76**	84.20	84.09
FV4	85.64	79.40	78.93

Table 3: Isolated chord recognition accuracy using GMM, training set: piano, testing set: piano [4].

Feature Vector	DS1A	DS1B	DS1C
FV1	68.06	42.00	33.62
FV2	42.72	18.60	16.30
FV3	43.49	22.00	18.31
FV4	**86.94**	80.23	80.18

Table 4: Isolated chord recognition accuracy using GMM, training set: piano, testing set: strings [4].

were averaged. The overall chord recognition accuracies for GMMs can be seen in table 3 and for piano, and in table 4 for strings. FV4 performed the best with the chords played on strings, and showed the least difference between recognizing piano and string chords [4]. The results of classification using SVMs can be seen in Table 5. Using SVMs clearly outperformed GMMs, with accuracy rates up to 95% on DS1A. This was not the case however, when SVMs were tested on strings when trained on piano. In this case SVMs actually did not work because an SVM requires knowledge of the type of data in the testing set, so these results were not included.

For the continuous single-instrument dataset, two parameters of chord segmentation were compared; the number of scatter points (7 scatter points means the partition frame plus 3 frames on each side), and the window length in seconds. As shown in Table 6, segmentation accuracy of around 90% was achieved. Melody tones that were not part of the chord made this more difficult than labeling the isolated chords [4].

3.2 Case 2: HMM Trained with Audio from Symbolic Data

In the next study [2], symbolic data in Humdrum data format is used to generate training data for an HMM. Humdrum is a software toolkit used for music research. The data is used to generate a chord label file (containing chord names and times), and an audio file, from the same data. Chroma analysis is then performed on the audio file to get a PCP, which is used as input to the trained HMM along with the chord label file. This system is visualized in Figure 3.

A 36-state HMM is used, with each state representing a chord (major, minor, and diminished for each 12 pitches). Once the model parameters are learned, the Viterbi algorithm is applied to find the optimal sequence in a maximum likelihood sense.

Feature Vector	DS1A	DS1B	DS1C
FV1	93.43	88.21	86.52
FV2	94.78	93.26	93.13
FV3	**95.23**	94.31	94.24
FV4	90.56	88.08	87.74

Table 5: Isolated chord recognition accuracy using SVM, training set: piano, testing set: piano [4].

	Number of Scatter Points			
Window Length (s)	**3**	**5**	**7**	**9**
0.025	55.74	69.40	69.67	73.77
0.050	66.12	84.70	88.25	85.25
0.100	68.31	87.98	**90.44**	90.16
0.200	68.31	87.98	**90.44**	**90.44**

Table 6: Continuous single-instrument segmentation accuracy with varying number of scatter points and window lengths [4].

Training Data	Test Data	Recognition Rate
Piano	Piano	68.69
String Quartet	Piano	73.40
Piano & Strings	Piano	74.41
Piano	String Quartet	79.35
String Quartet	String Quartet	79.76
Piano & Strings	String Quartet	80.16

Table 7: Recognition results for all six possible training - test pairs in [2].

3.2.1 Datasets

Two training datasets were used: the first consisted of 81 solo piano pieces by J.S. Bach, Beethoven, and Mozart, and the second consisted of 196 string quartet pieces by Beethoven, Haydn, and Mozart. These models were then tested on excerpts from the Kostka and Payne's book, which includes analysis and audio recordings done by the composers. Two testing sets - five piano solos and five string quartets were selected, with no overlap of the training data. Both of these sets were tested using both of the training datasets, as well as third that consisted of the first two combined, resulting in six possible training - test pairs. The output was compared to the hand-marked data for frame rate accuracy [2].

3.2.2 Results

The results of these experiments can be seen in table 7. The recognition rate was highest for the combined training dataset, at 80%, although the difference is not significant. Further analysis on the results showed that the highest error came from non-chord tones in the melody, especially in faster pieces. This is because the analysis window would span over multiple chord changes, which confused the system. Other issues were caused by the system treating 7th chords the same as triads because 7th chords contain two triads.

3.3 Case 3: Importance of Individual Components

The final study [1] consists of four experiments, testing different methods used for each step of the process. The overall recognition system in this case is defined as a feature extraction step and pattern matching step, with optional pre-filtering and post-filtering steps. An overview of the system can been seen in figure 4. Pre-filtering is applied to PCPs directly before pattern matching, and consists of averaging frames between beats, yielding a beat-synchronized PCP. Post-filtering is performed after pattern matching, in

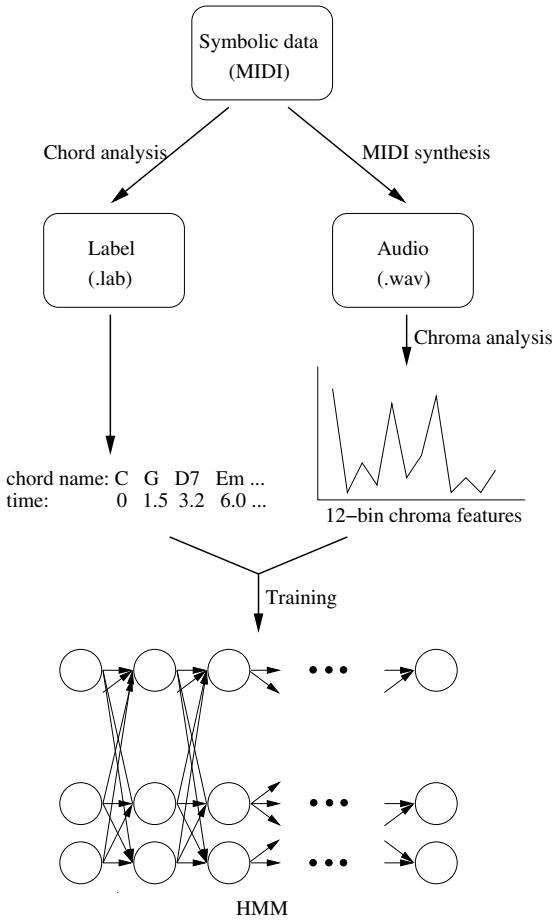

Figure 3: Overview of the HMM trained with audio from symbolic data in [2].

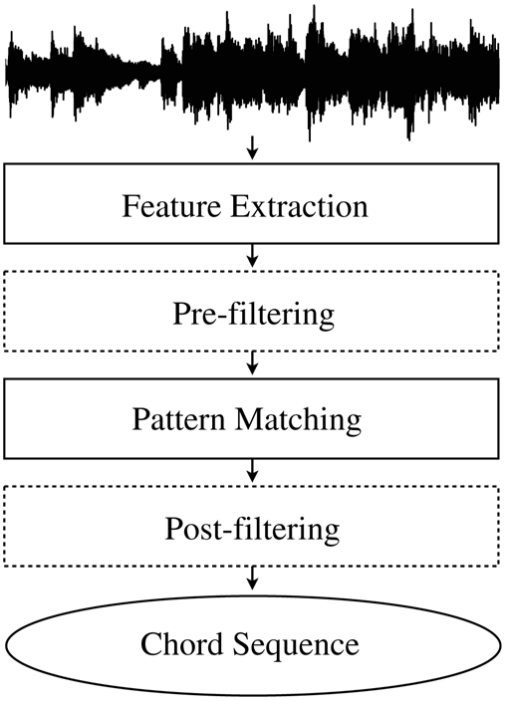

Figure 4: General layout of the chord recognition system used in [1].

the context of a HMM, typically using a Viterbi decoder.

decide between chord-annotated and hand marked or define difference.

3.3.1 Datasets

The dataset consisted of 180 Beatles songs, 20 Queen songs, 20 songs from the RWC (Real World Computing) pop dataset, and 195 songs from the US-Pop dataset. For training, 5-fold cross validation was used with each group having 99 songs selected randomly. For each fold one group is selected and the other four are used for training. Accuracy is represented by the total duration of correct chords out of the total duration of the dataset [1].

Needs more work

3.3.2 Results

In the first experiment of this case no filtering is applied during the process. This experiment is testing the difference between preprocessing techniques in the feature extraction stage. Results can be seen in table 8. The binary template is the hand-made chord model, and the rest are GMMs with a number indicating the number of Gaussian components used. We would expect to see higher accuracy with more processing and more Gaussian components used, but this is not the case. The highest error was in distinguishing major and minor chords with the same root, because two of the notes are shared and the third is only a half-step different.

Need to add the results for pre and post filtering experiments.

	Binary Template	GMM-1	GMM-5	GMM-10	GMM-15	GMM-20	GMM-25
Base	46.95	46.46	46.26	48.10	48.39	48.74	48.77
Overtone removal 1	52.12	49.40	47.38	50.24	51.04	51.42	51.71
Overtone removal 2	54.38	54.51	50.49	50.90	51.42	52.14	51.97
Timbre Homogenization 1	45.18	48.24	50.44	49.34	49.36	49.06	49.35
Timbre Homogenization 2	44.37	40.12	39.80	39.61	40.49	40.87	40.86
OR1 & TH1	55.51	**58.30**	57.58	57.73	57.72	57.70	57.69
OR1 & TH2	53.24	53.83	54.66	54.67	54.00	54.03	53.55
OR2 & TH1	55.00	56.29	53.09	53.24	53.28	53.33	53.43

Table 8: Average accuracy without filtering (research case 3, experiment 1).

4. CONCLUSIONS

Look at the components of the highest performing systems and determine the best techniques used for feature extraction and pattern matching. Discuss the effects of preprocessing during feature extraction and using HMMs to eliminate unlikely sequences. All of the research cases use PCP and preprocessing of some kind. Case 1 and 3 use GMMs. Case 2 and 3 use HMMs.

Talk about some of the issues and pitfalls with these and all chord recognition systems. Including dense recordings, extremely fast chord changes, and other types of chords that are not recognized. There are also chords that can only be determined by their context and chords that can be interpreted in more than one way.

4.1 Future Work

Mention study using online chord database. Genre-specific models.

5. ACKNOWLEDGEMENTS

Many thanks to Elena Machkasova, Kristen Lamberty, and Scott Steffes for their invaluable feedback.

6. REFERENCES

[1] T. Cho and J. Bello. On the relative importance of individual components of chord recognition systems. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(2):477–492, Feb 2014.

[2] K. Lee and M. Slaney. Automatic chord recognition from audio using a supervised hmm trained with audio-from-symbolic data. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, AMCMM '06, pages 11–20, New York, NY, USA, 2006. ACM.

[3] M. McVicar, R. Santos-Rodriguez, Y. Ni, and T. D. Bie. Automatic chord estimation from audio: A review of the state of the art. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(2):556–575, Feb 2014.

[4] J. Morman and L. Rabiner. A system for the automatic segmentation and classification of chord sequences. In *Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia*, AMCMM '06, pages 1–10, New York, NY, USA, 2006. ACM.