A simple chord detecton algorithm

Bowen Dong, Prof. Alexander Lerch

Abstract—In this article, we make research to implement and evaluate a simple chord detection system. Based on the method maintained by former contributions, this system can be divided into many different parts: spectrogram, pitch chroma, chord detection, post-processing (Viterbi), dataset and result evaluation. Each part will be comprehensively discussed below.

Index Terms—Music information retrieval, Chord detection system, Hidden Markov Model, Viterbi algorithm

1 Research Statement

THIS Project is to implement a system to automatically detect the chord progression in pop music, then to evaluate its accuracy. That is, to get a chord name sequence changing with time sequence. The chord name is formed like A, B, C, D flat…, and the time has the unit of second.

A chord, in music, is any harmonic set of pitches/frequencies consisting of multiple notes (also called "pitches") that are heard as if sounding simultaneously. For many practical and theoretical purposes, arpeggios and broken chords (in which the notes of the chord are sounded one after the other, rather than simultaneously), or sequences of chord tones, may also be considered as chords in the right musical context.

And chords sequence is such an important feature in music informatics retrieval field that chord sequences have been used by the research community in high–level tasks such as cover song identification (identifying different versions of the same song e.g.[1], [2]), key detection [3]–[6], genre classification (identifying style [7]), lyric interpretation [8] and audio–to–lyrics alignment [9], [10]. A typical chord sequence in chord detection is shown in figure 1.

```
0.000000 0.414179 N
0.414179 4.365655 F
4.365655 6.690592 G
6.690592 8.780606 C
8.780606 9.717289 A:min
9.717289 10.634761 G
10.634761 12.376258 C
12.376258 14.199024 F
```

Fig. 1. Chord sequence example

In this peoject, we get chord sequence in the same fomat from wave file and compute the correct percentage with annotation.

2 Motivation

As machine lerning become popular these years, recent researches on chord detection are always based on machine learning. But machine learning chord detection meets a problem that since the chords such as dimished, augment are used in a quite smaller number than major and minor chords. This caused problem when detecting such rarely used chords for the lack of training data. Thus, it's very hard to say rule-based method chord detection is outdated. In this project, we try to implement a chord detection system in a more traditional way, which is mainly a rule-based system.

3 Related works

In 2014, Christoph Hausner did a comprehensive work from the fourier transfer to chord detection [11]. In this project, Hausner maintain an evaluation method – computing overlapping ratio (OR).

4 Method

The implemention of this algorithm can be mainly devided into four parts: spectrogram computation, tunning frequency, pitch chroma computation, chord detection.

2.1 Spectrogram

The input of this module is wave file. By using the Fast Fourier Transform [12], we get spectrogram from wave file. Which is a 2-D array expressing the relationship between the frequency and frame.

A fast Fourier transform (FFT) is an algorithm that computes the discrete Fourier transform (DFT) of a sequence, or its inverse (IDFT) [13]. The DFT is calculated by the formula below:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N}$$

After the FFT we get a spectrogram. Figure 2 is a example of a spectrogram of a song:

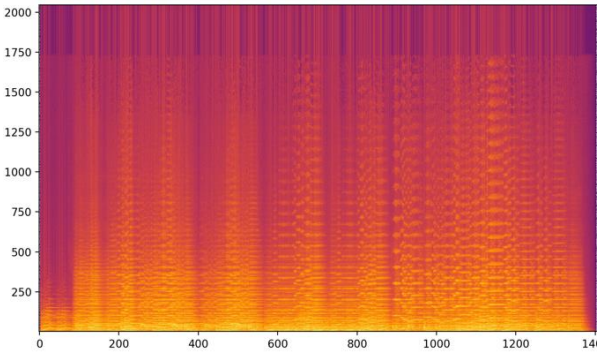

Fig. 2. Spectrogram example

2.2Tunning frequency

After we get a spectrogram, we know that the vertical axis is the unit of the sampling frequency. That is, for example, if we use 44100 as the sampling frequency fs, then the highest frequency can be shown in this spectrogram is 44100. So the frequency area of each section is:

$$fs/window\ length$$

Then the real frequency fr of a bin is:

$$fr = y \times fs/window\ lengt$$

Y is the coordinate on vertical axis.

Then we can compute the pitch, since the real frequency has a one-to-one relation with pitch. We have such a table in figure 3 shows this relation [14]:

Octave→ Note↓	0	1	2	3	4	5	6	7	8	9
C	16.352 (−48)	32.703 (−36)	65.406 (−24)	130.81 (−12)	261.63 (0)	523.25 (+12)	1046.5 (+24)	2093.0 (+36)	4186.0 (+48)	8372.0 (+60)
C♯/D♭	17.324 (−47)	34.648 (−35)	69.296 (−23)	138.59 (−11)	277.18 (+1)	554.37 (+13)	1108.7 (+25)	2217.5 (+37)	4434.9 (+49)	8869.8 (+61)
D	18.354 (−46)	36.708 (−34)	73.416 (−22)	146.83 (−10)	293.66 (+2)	587.33 (+14)	1174.7 (+26)	2349.3 (+38)	4698.6 (+50)	9397.3 (+62)
D♯/E♭	19.445 (−45)	38.891 (−33)	77.782 (−21)	155.56 (−9)	311.13 (+3)	622.25 (+15)	1244.5 (+27)	2489.0 (+39)	4978.0 (+51)	9956.1 (+63)
E	20.602 (−44)	41.203 (−32)	82.407 (−20)	164.81 (−8)	329.63 (+4)	659.26 (+16)	1318.5 (+28)	2637.0 (+40)	5274.0 (+52)	10548 (+64)
F	21.827 (−43)	43.654 (−31)	87.307 (−19)	174.61 (−7)	349.23 (+5)	698.46 (+17)	1396.9 (+29)	2793.8 (+41)	5587.7 (+53)	11175 (+65)
F♯/G♭	23.125 (−42)	46.249 (−30)	92.499 (−18)	185.00 (−6)	369.99 (+6)	739.99 (+18)	1480.0 (+30)	2960.0 (+42)	5919.9 (+54)	11840 (+66)
G	24.500 (−41)	48.999 (−29)	97.999 (−17)	196.00 (−5)	392.00 (+7)	783.99 (+19)	1568.0 (+31)	3136.0 (+43)	6271.9 (+55)	12544 (+67)
G♯/A♭	25.957 (−40)	51.913 (−28)	103.83 (−16)	207.65 (−4)	415.30 (+8)	830.61 (+20)	1661.2 (+32)	3322.4 (+44)	6644.9 (+56)	13290 (+68)
A	27.500 (−39)	55.000 (−27)	110.00 (−15)	220.00 (−3)	440.00 (+9)	880.00 (+21)	1760.0 (+33)	3520.0 (+45)	7040.0 (+57)	14080 (+69)
A♯/B♭	29.135 (−38)	58.270 (−26)	116.54 (−14)	233.08 (−2)	466.16 (+10)	932.33 (+22)	1864.7 (+34)	3729.3 (+46)	7458.6 (+58)	14917 (+70)
B	30.868 (−37)	61.735 (−25)	123.47 (−13)	246.94 (−1)	493.88 (+11)	987.77 (+23)	1975.5 (+35)	3951.1 (+47)	7902.1 (+59)	15804 (+71)

Fig. 3. Relationship between frequency and pitch

And the tuning frequency is implemented by the formula below:

$$p(f) = 69 + 12 \times \log_2(f/f_{A4})$$

In general, f_{A4} is 440Hz.

By this module, we aggregate the long frequency aixs into a pitch vector with 12 members. In this vector, 0-12 means C to B.

2.3 Pitch chroma

Do the computation as mentioned in 2.2 for each point in spectrogram, we get a $12 \times BlockNum$ 2-D array, which is a pitch chroma. Pitch chroma shows the relationship between pitch and time unit (frame). As example, one pitch chroma in this project is shown below in figure 4:

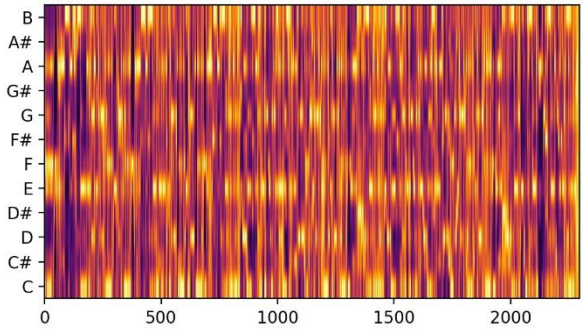

Fig. 4. Pitch chroma example

From pitch chroma, we can roughly see the chord changing with time.

2.4 Chord detection

In order to do the chord detection, firstly we have a chord template such as in figure 5:

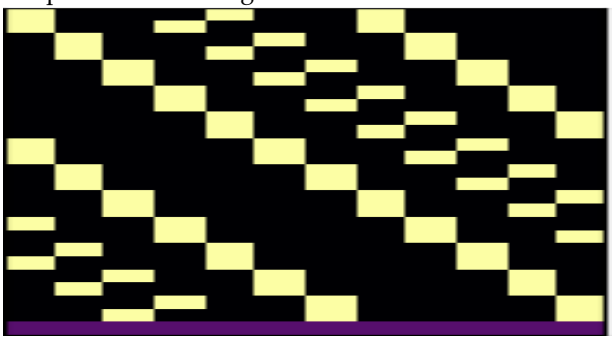

Fig. 5. Chord template

The chord template is 24 vectors which have the same format with pitch vector.

We compute the similarity of each frame in pitch chroma with each vector in chord template, and select the vector has the highest similarity as the result in that frame.

After repeat this operation for all frames, we get a sequence of chord changing with frame. In this project, we also output a figure shows that relationship, for example, in figure 6:

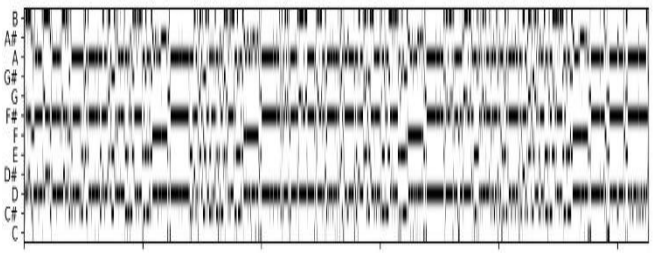

Fig. 6. Chord sequence example

After convert the frame to real time sequence, we get a chord sequence changing with time, that is an elementary result we get.

2.5 Viterbi algorithm

The Viterbi algorithm is a good method for post-processing to improve the accuracy [15].

Hidden Markov Model is a statistic model to describe a processing with invisible state. The difficulty is to get the invisible parameters from visible parameters, then use those invisible parameters for futher analysis.

A Markov chain is shown in figure 7:

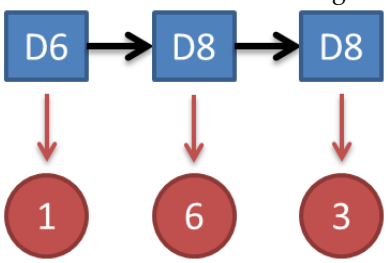

Fig. 7. Markov chain example

In this example, D6, D8 are invisible state, 1,6,3 are visible obervations.
The HMM has 5 parameters:
In a chord detection system:
State: a chord to be estimated
Observation: each pitch chroma is an observation
Transition probability: the probability of a state transiting to another state.
Emission probability: the relationship between obervations and states
Start probability: the probability of a state to be the initial state
These 5 parameters will be prepared and be the input of a Viterbi algorithm. The Viterbi algorithm is a way to compute the chortest path in a Hidden Markov Model.
In chord detection system, that is the most possible chord sequence.

5 Evaluation

We do the evaluation by calculating the overlap ratio OR, that is, the duration of correctly identified chords over duration of all identifiable chords.

$$OR = \frac{\text{duration of correctly identified chords}}{\text{duration of all identifiable chords}}$$

Since the result and the annotation are both have three columns – the first column is the start time of a chord, the second column is the end time of a chord, and the third column is that chord's name. In the traversal of rows of the annotation, we get the duration of a chord by using end time minus start time of a chord, then to determine whether it is correct or not. After accumulation of all chords, we get an overlap ratio of a song
For evaluation, we use the Beatles Dataset made by Queen Mary University of London (http://isophonics.net/content/reference-annotations)
This dataset has the same format mentioned above.
This dataset has complex chord such as seventh chord, inverse chord, which will be regarded as traid and its root position.

6 Result and discussion

Here is the result of the first album in Beatles dataset:

Song	OR
I Saw Her Standing There	0.22040172701099536
Misery	0.27963337166548397
Anna Go to Him	0.1636753020826686
Chains	0.2859835453629951
Boys	0.2137261362767764
Ask me Why	0.23151795130969255
Please Please Me	0.2680379986732092
Love Me	0.16390651053166272
P. S. I Love You	0.46477300647332287
Baby It's You	0.3311624539765844
Do You Want To Know A Secret	0.16092035640541746
A Taste Of Honey	0.31184683405577907
There's A Place	0.3092328238862546
Twist And Shout	0.18497658230667324

Given the length limitation, there is only the list of one album.
When the structure of a song become more complex, the accuracy of a result is lower and vice versa.
This result dosen't have an ideal accuracy. As discussed, one possible explain is the fluctuation in the chord sequence. That is, the chord is detected for each frame, and if there is a wrong chord in one frame, it will be part of the result, which leads to the fluctuations we can find in figure 5. One possible way to solve this problem is to add a smoothing module for the chord sequence. For example, using median filter on possibility chord matrix or a post-processing operation using Viterbi algorithm.

7 Novelty of proposed work

As a learn of music information retrieval, this research is a rare one that includes all the work from spectrogram to chord detection without third party packages except numpy. Besides, other than the pitch chroma, in the chord selection module, this system output another roughly chord sequence figure, which make the intermidate result more clearly ocular.

8 Deliverables

The deliverable of the research is a python application with result, the OR data, output in command line.

9 Conclusion

In this reseach, we can get a conclusion that the result, the accuracy of a chord detection is strongly influenced by the complexity of the structure of the songs. A more complex structure the songs have, a lower accuracy the result will have.
Besides, if we regard the modules mentioned in this article as the main branch in a chord detection algorithm. Compared with the main part, the other modules such as pre-processing, post-processing influence the accuracy even more.
As the first research in music information retrieval field, the aim of this work for me myself is to cross the threshold of this field and to accumulate experience of python programming. Thus this research is expected to include as much stuff as possible in a fundamental part of music information retrieval – the chord detection.
Although dosen't get an accurate result, at least we implemented functions in all modules. As a beginning research this research give me much useful experience, which will be helpful in my future study.

Acknowledgment

The authors wish to thank Music Informatic Group in GT. They gave much helpful advice for this research.

REFERENCES

[1] D. Ellis and G. Poliner, “Identifying ‘cover songs’ wisth chroma features and dynamic programming beat tracking,” in Proc. Int. Conf. Acoust., Speech, Signal Process., 2007, pp. 1429–1433.

[2] E. Gómez and P. Herrera, “The song remains the same: Identifying versions of the same piece using tonal descriptors,” in Proc. 7th Int. Soc. Music Inf. Retrieval, 2006, pp. 180–185.

[3] B. Catteau, J. Martens, and M. Leman, “A probabilistic framework for audio-based tonal key and chord recognition,” in Proc. 30th Annu. Conf. Gesellschaft fur Klassikation, 2007, pp. 637–644, Springer.

[4] A. Shenoy and Y. Wang, “Key, chord, and rhythm tracking of popular music recordings,” J. Comput. Music, vol. 29, no. 3, pp. 75–86, 2005.

[5] K. Lee and M. Slaney, “Acoustic chord transcription and key extraction from audio using key-dependent HMMs trained on synthesized audio,” IEEE Trans. Audio, Speech, Lang. Process., vol. 16, no. 2, pp. 291–301, Feb. 2008

[6] V. Zenz and A. Rauber, “Automatic chord detection incorporating beat and key detection,” in Proc. IEEE Int. Conf. Signal Process. Commun., 2007, pp. 1175–1178.

[7] C. Perez-Sancho, D. Rizo, and J. Inesta, “Genre classification using chords and stochastic language models,” Connect. Sci., vol. 21, no. 2-3, pp. 145–159, 2009

[8] T. O’Hara, “Inferring the meaning of chord sequences via lyrics,” in Proc. 2nd Worskshop Music Recommendation Discovery collocated with ACM-RecSys, 2011, p. 34.

[9] M. Mauch, H. Fujihara, and M. Goto, “Integrating additional chord information into HMM-based lyrics-to-audio alignment,” IEEE Trans. Audio, Speech, Lang. Process., vol. 20, no. 1, pp. 200–210, Jan. 2012.

[10] M. Mauch, H. Fujihara, and M. Goto, “Lyrics-to-audio alignment and phrase-level segmentation using incomplete internet-style chord annotations,” in Proc. 7th Sound Music Comput. Conf., 2010, pp. 9–16

[11] Christoph Hausner, Design and Evaluation of a Simple Chord Detection Algorithm” University of Passau, 2014.

[12] Alexander Lerch, “An Introduction to Audio Content Analysis and Music Information Retrieval,” pp. 46, 2021.

[13] Alan V. Oppenheim, Ronald W. Schafer, “Discrete-Time Signal Processing third edition,” pp.623-625, 1989.

[14] Long737, "Table of note frequencies," CSDN blog, 2017.

[15] Stay_foolish12, "一文搞懂 HMM（隐马尔可夫模型）-Viterbi algorithm," CSDN blog, 2019