Homework 1

Global key and local key detection of audio and symbolic music

Li Su
Institute of Information Science, Academia Sinica, Taiwan
lisu@iis.sinica.edu.tw

(Don't worry if you are not familiar with the music theory of tonality, mode, and key before doing this homework. Instead, taking this assignment will help you learn the theory!)

Tonality, or so-called music *key*, is one of the most important attributes in music. In brief, key refers to two aspects of a musical scale: tonic and mode. The tonic is the first note of a diatonic scale. The mode is usually known as 'major' key, 'minor' key or so. Tonality is identified by the tonic note and the tonic chord. There are some simplified (yet imprecise) ways to identify a musical key. For example, the tonic note is usually recognized as the *first* or the *last* note of a music piece. Moreover, if the chord corresponding to the tonic (i.e., the tonic chord) is a major chord, the music piece is then in major key. On the other hand, if the tonic chord is a minor chord, the music piece is then in minor key. However, this over-simplified way in finding key usually fails in most of the real-world musical data. As a high-level concept, musical key is long-term and context dependent. According to the musical context, a music piece may have a global key as its main key, and local keys which may change several times in the music piece. The detection of global and local keys is still not yet a solved problem in MIR.

In this assignment, we will design some global and local key detection algorithms for global and local music key detection for both audio and symbolic data, with full or limited contextual information. You will learn how to extract audio features, how to deal with MIDI data, and the basic music theory and its computational aspects in this assignment.

The concept of a musical key

Let's start from the notion of the major and minor scales. Denote T as a tone and S a semitone, a major scale is a note sequence represented as T-T-S-T-T-T-S while a minor scale is T-S-T-T-S-T-T. The *functions* of these seven notes are tonic, supertonic, median, subdominant, dominant, submediant, and leading tone, respectively (see Figure 1). The major and minor scales are the two most commonly seen diatonic scale. If the tonic of a major scale is C, we then call it a C major scale. If the tonic of a minor scale is C, we then call it a C minor scale (see Figures 2 and 3). In Western classical music, there are in general 24 keys (two modes time 12 tonic notes).

A major scale and a minor scale that have the same tonic are called *parallel keys*. For example, the parallel minor of a C major key is a C minor key. A major scale and a minor scale that have the same

key signatures are called *relative keys*. have For example, the relative minor key of the `C` major key is the `A` minor key, the relative minor of `E` major is `C#` minor, etc.

(PS: Do not confuse the major/minor key with the major/minor chord. A chord is the co-occurrence of (usually 3) notes, like the major triad and the minor triad, while a key represents the structural information in a diatonic scale.)

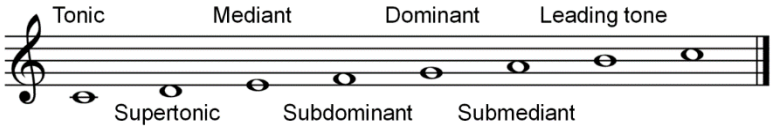

Figure 1: the diatonic scale. (Figure from:)

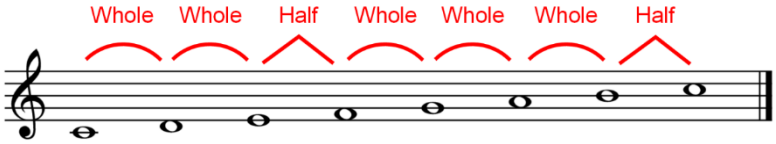

Figure 2: the C major scale.

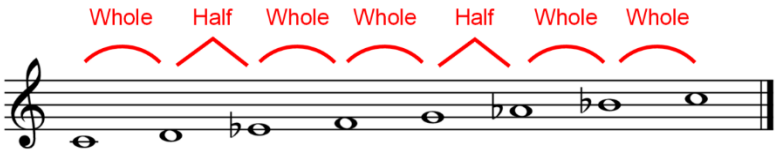

Figure 3: the C minor scale.

Prerequisite:

The following libraries are suggested for this assignment:

- `librosa`, a Python library for music and audio signal processing:
- `pretty-midi`, a Python library for MIDI signal processing:
- `mir_eval`, a Python library for MIR evaluation:

The following datasets will be used:

The **GTZAN** dataset and Alexander Lerch's annotation of key :
[Dataset] https://drive.google.com/open?id=1Xy1AIWa4FifDF6voKVutvmghGOGEsFdZ
[Annotation] https://github.com/alexanderlerch/gtzan_key
Each sample in the GTZAN dataset is a 30-sec clip of music in 10 different genres. We will use the data from the following 9 genres in the dataset for experiment: blues, country, disco, hip-hop, jazz, metal, pop, reggae, and rock.

Schubert Winterreise Dataset (**SWD**):
[Dataset and Annotation] https://zenodo.org/record/4122060#.YituDHpBy5e
This is a multimodal dataset comprising various representations and annotations of Franz Schubert's song cycle Winterreise. Schubert's seminal work constitutes an outstanding example of the Romantic song cycle—a central genre within Western classical music. Some of the versions are unavailable online; our TAs will collect them for you as many as they can.

The **GiantStep** dataset: [Dataset and Annotation]
https://drive.google.com/drive/folders/1D-PKkNWkWIQYcUDQokdzAFU0EL-oa3lc?usp=sharing

In summary, the available data can be downloaded from the following link provided by the TA:
https://drive.google.com/drive/folders/1eS_UUX2MrEbEeTVmiDZwIrSW5VBamNrX

Task 1: Global key detection based on template matching

We assume that the tonic pitch is the one which *appears the most often* in a music recording. Based on this assumption, the tonic pitch of a music recording can be estimated by the following process:

1. Compute the chromagram $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots \mathbf{z}_i, \cdots, \mathbf{z}_N]$. Each $\mathbf{z}_i$ is a 12-dimensional chroma vector at the ith frame, and N is the number of frames in each song.
2. Take average of all the chroma vectors over all the time frames in each song and obtain the song-level chroma vector $\mathbf{x}$ (this process is usually referred to as mean pooling)

$$\mathbf{x} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{z}_i$$

3. The maximal value of the song-level chroma vector indicates the tonal pitch. For example, if the maximal value of $\mathbf{x}$ is at the index of the C note, our estimation of the tonic is C.
4. Based on the estimated tonic, the final step is to find the mode (we consider only major and minor modes in this assignment) with template matching. In this step, the mode is determined by the correlation coefficient $R(\mathbf{x}, \mathbf{y})$ between $\mathbf{x}$ and the **binary-valued templates** $\mathbf{y}$. For example, if the tonic is C, then we consider two mode templates, one for the C major mode and the other for the C minor mode: $\mathbf{y}_{\text{C Major}} = [1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,0\,1]$ and $\mathbf{y}_{\text{c minor}} = [1\,0\,1\,1\,0\,1\,0\,1\,1\,0\,1\,0]$ (The first index of $\mathbf{y}$ indicates C note, the second index is C# note, …, and the 12th index is B note.). If we find that $R(\mathbf{x}, \mathbf{y}_{\text{C Major}}) > R(\mathbf{x}, \mathbf{y}_{\text{c minor}})$, the estimated key is then C Major. The correlation coefficient between the song-level chroma and the template is defined as

$$R(\mathbf{x}, \mathbf{y}) = \frac{\sum_{k=1}^{12}(\mathbf{x}_k - \bar{\mathbf{x}})(\mathbf{y}_k - \bar{\mathbf{y}})}{\sqrt{\sum_{k=1}^{12}(\mathbf{x}_k - \bar{\mathbf{x}})^2 \sum_{k=1}^{12}(\mathbf{y}_k - \bar{\mathbf{y}})^2}}$$

It should be noted that the step 3 and step 4 are interchangeable. We may first compute the correlation coefficients of the song-level chroma and the 12 key templates (for binary templates, there are only 12 templates in total rather than 24, since the template of C Major is the same as the template of a minor) and find the one which achieves the highest correlation coefficient. Then, we compare the values at the two possible tonic positions (e.g., C or A for C major or a minor, respectively). The largest value corresponds to the tonic. You may use some existed package such as `scipy.stats.pearsonr` in the `scipy` library to find the correlation coefficients or you can implement it directly (it is not complicated). There are 24 possible keys, and a music piece only has one global key. In Alexander Lerch's annotation of the GTZAN dataset, the key symbols are indexed as follows (upper case means major key and lower case means minor key):

A	A#	B	C	C#	D	D#	E	F	F#	G	G#
0	1	2	3	4	5	6	7	8	9	10	11
a	a#	b	c	c#	d	d#	e	f	f#	g	g#
12	13	14	15	16	17	18	19	20	21	22	23

5. For evaluation of key finding algorithm, first, the raw accuracy is defined as:

$$\text{ACC} = \frac{\text{number of correct detection}}{\text{number of all music pieces in the dataset}}$$

The raw accuracy is however unable to resolve the ambiguity in key perception. For example, the `C` major key is easily to be detected as `G` major key (a perfect-fifth error), `A` minor key (a relative-major/minor error), or `C` minor key (a parallel-major/minor key), because these erroneous keys are intrinsically "close" to `C` major keys. To solve this issue, we also consider the weighted score, which gives relative weights to the results having relation to the ground key:

Relation to correct key	Points
Same	1.0
Perfect fifth	0.5
Relative major/minor	0.3
Parallel major/minor	0.2
Other	0.0

Therefore, the weighted accuracy is defined as:

$$\text{ACC} = \frac{\#\text{ Same} + 0.5(\#\text{ Fifth}) + 0.3(\#\text{ Relative}) + 0.2\ (\text{Parallel})}{\#\text{ of all music pieces in the dataset}}$$

You can directly use the evaluation function `mir_eval.key.evaluate` in the `mir_eval` library.

Besides the binary-valued templates, let's also consider other designs of the key templates:

The Krumhansl-Schmuckler key-finding algorithm. A more advanced set of templates for key detection is the Krumhansl-Schmuckler (K-S) profile. Instead of using binary-valued templates, we assign values to the template according to human perceptual experiments. The template values are shown in the following Table (see the columns labeled by K-S). The experiment is done by playing a set of context tones or chords, then playing a probe tone, and asking a listener to rate how well the probe tone fit with the context. In this case, we consider using the correlation coefficient between the input chroma features and the K-S profile for key detection. Notice that the major and minor templates are here rendered by different values, so the templates of the C Major and a minor will not be the same. Therefore, in this case we don't need to probe the tonic first, but just need to find the maximal correlation coefficient among the major profile, minor profile, and the 12 circular shifts of them, respectively. A web resource http://rnhart.net/articles/key-finding/ demonstrates this idea.

Major key			Minor key		
Name	Binary	K-S	Name	Binary	K-S
Tonic	1	6.35	Tonic	1	6.33
	0	2.23		0	2.68
Supertonic	1	3.48	Supertonic	1	3.52
	0	2.33	Mediant	1	5.38
Mediant	1	4.38		0	2.60
Subdominant	1	4.09	Subdominant	1	3.53
	0	2.52		0	2.54
Dominant	1	5.19	Dominant	1	4.75
	0	2.39	Submediant	1	3.98
Submediant	1	3.66		0	2.69
	0	2.29	Leading tone	1	3.34
Leading tone	1	2.88		0	3.17

The harmonic templates. We assume that the strength of the fundamental frequency of a note is 1, and the strength of the kth harmonic of a note is α^k, $0 < \alpha < 1$. Consider the harmonic order to seven, then the chroma template of a single C note is

$$\boldsymbol{u_c} = (1 + \alpha + \alpha^3 + \alpha^7, 0,0,0, \alpha^4, 0,0, \alpha^2 + \alpha^5, 0,0, \alpha^6, 0)$$

And the template of the C Major key is then $\boldsymbol{u_C + u_D + u_E + u_F + u_G + u_A + u_B}$. The harmonic templates for the 24 major/minor keys are therefore constructed in this way.

The data-driven templates. All the above templates are determined by our domain knowledge of music. However, construct the template from real-world data (i.e., the machine learning approach) would be expected as the "ultimate" solution because our application scenario is always on the real-world data. To develop data-driven key finding algorithms, practical issues include the size of the data, the consistency between the training and testing data, data imbalance, data augmentation, and over-fitting, etc. In this assignment, as a bonus question, we consider an external dataset (the GiantStep dataset) as the training set, and we wish to train the templates from this training set. This process is also known as dictionary learning in the literature of machine learning. Three dictionary learning methods are suggested in this assignment: 1) means of the chroma vectors for each class, 2) random sampling the chroma vectors from each class, and 3) k-means clustering method. See the description of the bonus question in detail.

Q1 (40%) Perform global key finding on the 9 genres in the GTZAN dataset using the feature settings of 1) STFT-based chromagram, 2) CQT chromagram and 3) CENS chromagram and the matching scheme of 1) binary-valued template matching, 2) K-S template matching, and 3) harmonic template matching (you may try $\alpha = 0.9$). Again, since there is no annotation in the classical genre, you don't need to run that genre. Report the raw accuracy and weighted accuracy per genre and per method. Which genre achieves better performance and why? Which method appear to be more competitive and why? Discuss your results.

Hint: the chroma features can be obtained from the following functions:

- `librosa.feature.chroma_stft`
  `librosa.feature.chroma_cqt`
  `librosa.feature.chroma_cens`

Q2 (30%) Repeat the process in Q1 on the MIDI data and all the available audio versions (i.e., HU33, SC06, FI66, FI80) of the Schubert Winterreise Dataset. Report the average raw accuracy and weighted accuracy for each version. Is there any difference among the versions? Are MIDI data easier for key finding? Discuss your results.

Hint: for symbolic data, you may use `pretty_midi.Instrument.get_chroma` to get the chroma vector.

Q3 (bonus) Construct the templates for the 24 major/minor keys using the GiantStep dataset. There are many possible ways to construct the templates. There can also be multiple templates for each key.

For example, the template of D major can be constructed by taking the average over all chroma vectors annotated as D major in the dataset. We can also take the k-means algorithm over these chroma vectors to obtain k templates for D major. For the keys not in the dataset, you may consider constructing them by circular shifting from the existing keys. Perform global key finding on the GTZAN dataset using the data-driven template. Does this method benefit some genres? Discuss your results.

Task 2: Local key detection

In the previous task, we assume that one music piece has only one key. This is however not the case for Western classical music, where *key modulation* is heavily used. That means, the key of a music piece may change over time. In the following task, we will design a key detector that can find the local key, and we will evaluate the algorithm on both audio and symbolic datasets.

Similarly, the raw accuracy and weighted accuracy of local key finding can be defined as

$$\text{ACC} = \frac{\#\text{ of correct detection}}{\#\text{ of time instances (detections) in all music pieces}}$$

$$\text{ACC} = \frac{\#\text{ Same} + 0.5(\#\text{ Fifth}) + 0.3(\#\text{ Relative}) + 0.2\ (\text{Parallel})}{\#\text{ of all time instances (detections) in all music pieces}}$$

Note that these accuracies count the number of time instances rather than the number of pieces.

Q4 (20%): Based on Task 1, design a local key detector that outputs the key of the music every 0.1 second. That means, there is a key detection output for every time step, and in this task, we set the time step be 0.1 second. Perform your method one the MIDI data and all the available audio versions of the Schubert Winterreise Dataset. For simplicity, let's evaluate the results against the annotator 1. Report the raw accuracy and the weighted accuracy.

Hint: to get the local tonality feature, you may consider the mean-pooled chroma of a segment (maybe 30 seconds or so), not of the whole music piece. For example, the feature representing the local key at the 60th second can be obtained by summing up the chorma vectors from the 45th to the 75th second. You may try the optimal segment size empirically.

Q5 (10%): The local key detection problem can be regarded as a segmentation problem. There has been evaluation metrics for the segmentation performance in the chord recognition problem, but such metrics have not been applied in local key detection. Please apply the over-segmentation, under-segmentation and average segmentation measures (please refer to the directional Hamming divergence and see page 33 in Lecture 3 slides) on the local key detection of the Schubert Winterreise Dataset.

Hint: these metrics have been implemented somewhere in `mir_eval.chord`.

Q6 (bonus): if possible, please design an algorithm that (hopefully can) outperforms the template matching algorithms introduced here. You may use more advanced method (e.g., deep learning) and novel data representations that you may want to create.

Please submit your .zip file containing the report (PDF) and your codes, with the file name "HW1_[your ID]" to the course website.

The deadline of Assignment 1 is April 26, and we will discuss it on May 3.