Cover Song Detection

Mathieu Lagrange

This document, the code and the data are available here:
https://github.com/mathieulagrange/musnu/tpCover

1 Introduction

The aim of this practical is to explore several means of enforcing invariance to signal deformation in order to maximize the locality of similar items in the feature space.

The application area will be the detection of cover songs. A cover song is a song initially played by a given band (say Let it Be from the Beatles) played by another band (cover from David Bowie). Even if the cover is easily and universally recognized by a human listeners (even non musically trained), the sound features that triggers the detection of covers are challenging to design.

The most important feature is arguably the melody. Thus, the system we are goin to be build shall build features that are invariant to:

1. translation: phase shift
2. transposition: frequency shift
3. tempo change: time warping

We are going to implement and test them with 2 increasing difficulty test cases: local deformation and cover songs.

2 Local deformations

2.1 Data

Please consider the **hello** dataset available in the repository. It contains several modified versions of a reference "hello" signal, plus a "null" signal containing white noise.

Listen to the sounds. The **hello.wav** sound is the orginal sound. Others (**'Shift', 'FrameHalf', 'VocoderHalf', 'ResampleTwice'**) are slightly deformed versions. Another one, named **helloNoise.wav** contains white noise.

The aim here is to find a representation that groups together the original hello signal and its modified versions.

2.2 Metric

The amount of grouping the representation r allows is evaluated by measuring how far are the modified versions from the original, divided by the distance of those modified versions to the null signal:

$$score_k = \frac{1/4 \sum_{i=1}^{4} d(r^k_{ori}, r^k_i)}{d(r^k_{ori}, r^k_{null})}$$

where i is the index among the modified signals. The null signal thus acts as a reference point in the feature space.

2.3 Representations

1. **Time**: compute the euclidean distance between the signals.
2. **Frequency**: compute the euclidean distance between the magnitude spectrogram vectorized. Use a frame size of 2048.
3. **Chroma**: In order to have a representation that is invariant to octave transposition, compute the euclidean distance between the 12 dimensional chromagram vectorized (use the `fft2chromamx` function in matlab or the `librosa.feature.chroma_stft` in python).

   Please display and discuss the shape of the projection matrix produced by `fft2chromamx`; It is necessary to select the dimensions that you need by considering the folowing selection: `cmx = fft2chromamx(nfft,12,fs);`
   `cmx = cmx(:,1:(nfft/2)+1);`
4. **Transposed chroma domain**: in order to be invariant to transposition below the octave, for each pair, find the chroma transposition that induces the smallest euclidean distance between the chromagrams (use the `circshift` function in matlab. Keep the transposition index in a vector.
5. **Elastic transposed chroma**: Use the dynamic time warping (dtw) to measure the distance between transposed chromagrams. Use the Manhattan distance for the dtw and consider the optimal transposition index found in the previous representation.

Display the value of the metric for each representation. The value shall decrease as the representation is more and more invariant.

3 Covers

3.1 Data

Please download the covers80 dataset :https://labrosa.ee.columbia.edu/projects/coversongs/covers80/.

In order to keep a reasonable amount of computation time, select the 20 first songs (the dataset will thus be of 40 songs in total) and use only the first 60 seconds of each songs.

3.2 Metrics

In order to evaluate the performance of the proposed systems, we are going to consider 2 metrics:

1. the hit ratio: the percentage of time the system returned the cover of a given seed song
2. the mean reciprocal rank (mrr): the multiplicative inverse of the rank of the actual cover.

$$mrr = \frac{1}{20}\sum_{i=1}^{20}\frac{1}{rank_{cover}}$$

Questions

- What are the performance of each invariance enforcement system (Use the euclidean distance for the dtw) ?
- Listen to some false positive. Are there some that are indeed ambiguous ?

A Report

Please write a report using your favorite word processing tool and output a pdf file. The report shall have for each question a brief description about the way things have been done and some discussion about the resulting behavior.

Send an archive containing the report and the code no later than an hour after the end of the session to `mathieu.lagrange@cnrs.fr`, with the `[ECN]` flag within the title of the message.

B Useful Matlab commands

B.1 Audio

- audioread: load audio stream from file
- audiowrite: write vector or matrices to an audio file
- audioplayer: module that allows you to play sounds

B.2 Miscellaneous

- hist : histogram
- repmat : matrix replication
- imagesc : scaled matrix display

B.3 Documentation

- doc ”command” : documentation of ”command”
- lookfor ”keyword” : show commands with ”keyword” in the description

B.4 Debug

- dbstop if error : stop the execution at failure with active callstack