EQ2341 Pattern Recognition and Machine Learning Assignment 2

Ayushi Shah
ayushi@kth.se

Sri Janani Rengarajan
sjre@kth.se

April 29, 2020

1 Music Features

To get the better understanding of the melodies we have plotted the features. Figure 1 shows the pitch profiles of all the three melodies. We have changed the Y-axes to the logarithmic scales and limited to the frequency 100-300 Hz in order to get the clear view. Figure 2 and figure 3 shows correlation and intensity profiles respectively. We have also transformed the Y-axes of intensity profiles into logarithmic scale since both Pitch and intensities varies a lot. X-axes of all the plots shows time in seconds.
From figure 1 we can easily recognise from observation that plots of melody 1 and melody 2 seems to be similar, so we can say that melodies 1 and 2 share a similar quotient between different frequencies but melody 3 does not. Thus by pure observation we can say that melody 1 and melody 2 are from the same melody but melody 3 is from a different melody. MatLab code for generating these plots is added in the zip folder with the file name as FeaturesFromMelody.m

2 Feature Extractor Design

Using "GetMusicFeatures" function, each song is split into three components - pitch frequency, correlation coefficient between adjacent pitch periods, intensity. For the proposed feature extractor, these three components are combined to obtain a single feature called semitone as the combination of semitones forms the basis of Western music.
Feature extractor design steps:

1. Semitones are computed from the pitch frequencies. The formula is presented in equations 1 and 2
    - f_k = pitch frequency for which semitone value is computed
    - f_0 = base pitch frequency, the lowest frequency of the song
    - k = semitone value for f_k
2. Pauses, noises and non notes are treated as outliers. Outliers are detected by thresholding the pitch, correlation coefficient and intensity signals. For outlier detection, Matlab's inbuilt function "isoutlier" is used. A simple threshold based on mean of the signal is not sufficient for all the signals. For example, the pitch

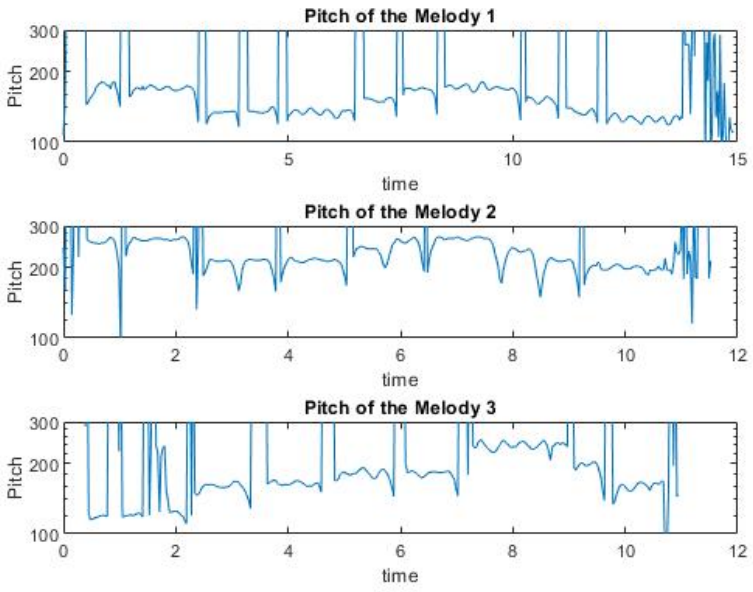

Figure 1: Pitch profiles of the melodies with time in seconds and Pitch in Hertz

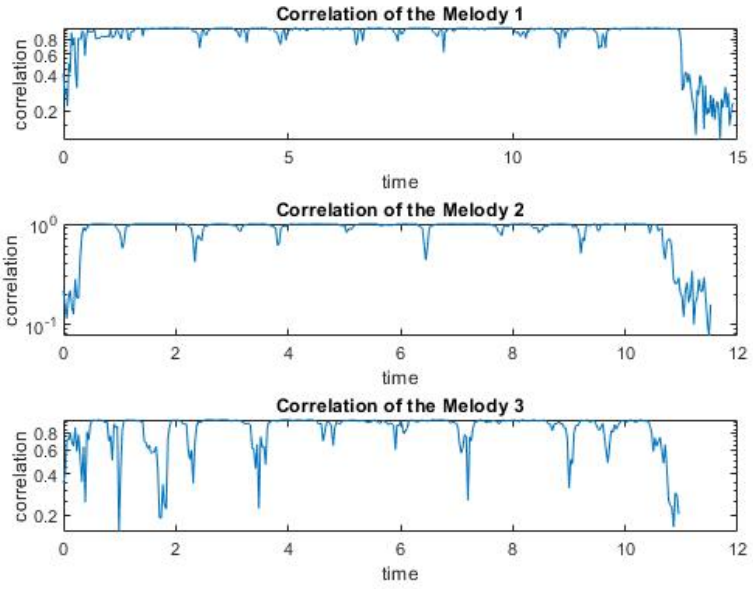

Figure 2: Correlation profiles of the melodies with time in seconds

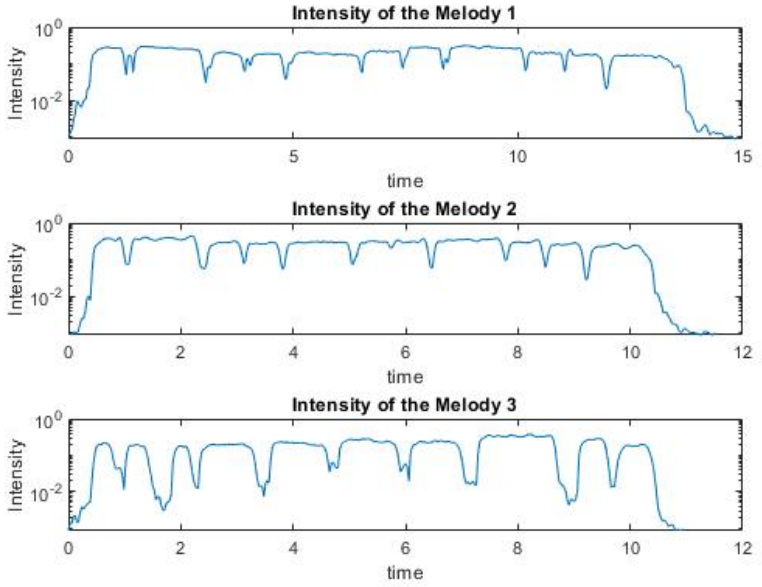

Figure 3: Intensity profiles of the melodies with time in seconds

frequency and correlation coefficient signals have a lot of oscillations, extreme low and high values. These extreme variations may bias the mean value, statistics that can be used for outlier detection is presented in reference [1]. Hence, based on the signal variations different thresholding methods are used for different signals using the Matlab "isoutlier" function.

- Pitch frequency - Thresholding based on "quartiles". An outlier is when the signal value is 1.5 interquartile ranges above the upper quartile or below the lower quartile as shown in figure 4.
- Correlation coefficient - An outlier is when the signal value is more than three scaled deviations away from the median. In figure 5 the center value is the median of the signal, the upper and lower thresholds are 3 deviations away from the median.
- Intensity - An outlier is when the signal value is less than the mean of the signal.

3. If a time instant has an outlier in the {pitch signal} OR {in correlation coefficient AND intensity signals}, the pitch value at that time instant is treated as an outlier. These outlier pitch values are not removed but converted into semitone values that are close to zero. This is done so in order to keep the time duration and continuity of the final semitone feature signal same as the original pitch signal.

4. Extracted semitone features are presented in figure 7

$$\begin{aligned} \log f_k &= \log f_0 + (k-1).\frac{\log f_e - \log f_0}{12} \\ &= \log f_0 + (k-1).\frac{\log 2}{12} \\ \log \frac{f_k}{f_0} &= (k-1).\frac{\log 2}{12} \end{aligned} \tag{1}$$

$$
\begin{aligned}
k &= 12.\frac{\log(f_k/f_0)}{\log 2} + 1 \\
&= 12.\log_2 \frac{f_k}{f_0} + 1
\end{aligned}
\tag{2}
$$

Following is the code for noise detection and semitones calculation.

```
function semitones = semitoneCalc_outlier(frIsequence, time, verbose, melodyNumber)
property = cell(1,3);

%Data processing of 3 properties
for i = 1:3
    x = time;
    if i==1
        prop='pitch';
        A = log(frIsequence(1,:));
        [TF,L,U,C] = isoutlier(A,'quartiles');
        m = mean(A);
    elseif i==2
        prop='Correlation';
        A = frIsequence(2,:);
        [TF,L,U,C] = isoutlier(A);
        m = mean(A);
    else
        prop='Intensity';
        A = log(frIsequence(3,:));
        m = mean(A);
        TF = logical(A<m);
    end
    l = length(x);
    property{i}= x(TF);
%noise calculation
for j = 1:length(time)
    t = time(j);
    if ismember(t,property{1}) || (ismember(t,property{2}) && ismember(t,property{3}))
        noise(j) = 1;
    else
        noise(j) = 0;

    end
end

%semitone calculation
pitch = log(frIsequence(1,:));
basePitch = min(pitch(find(noise==0)));
semitones = 12*log2(pitch/basePitch) + 1;
semitones(find(noise==1)) = 0.5*rand(size(find(noise==1)));
```

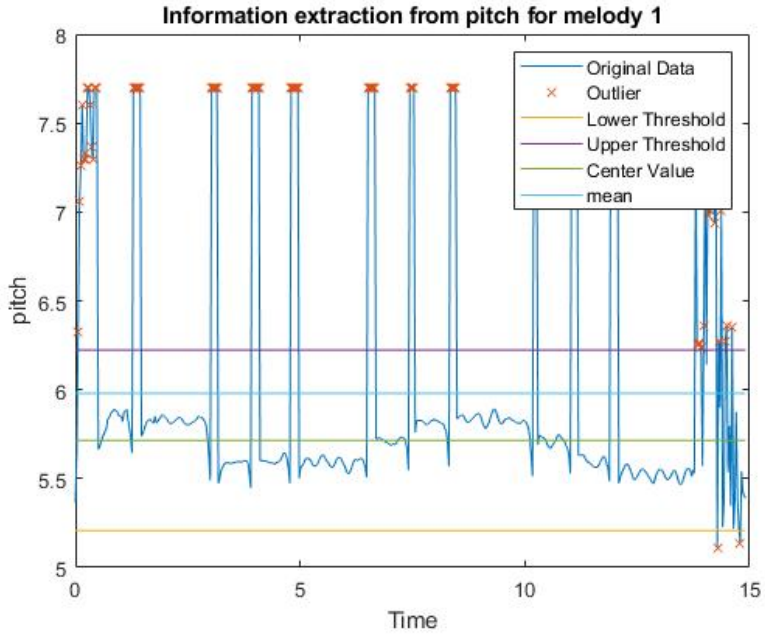

Figure 4: Pitch information extraction with time in seconds and pitch in Hertz

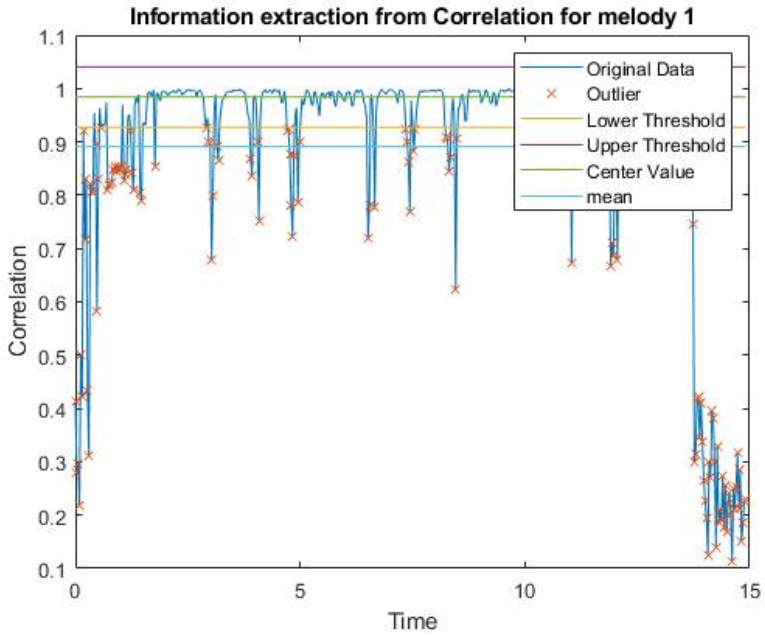

Figure 5: Correlation information extraction with time in seconds

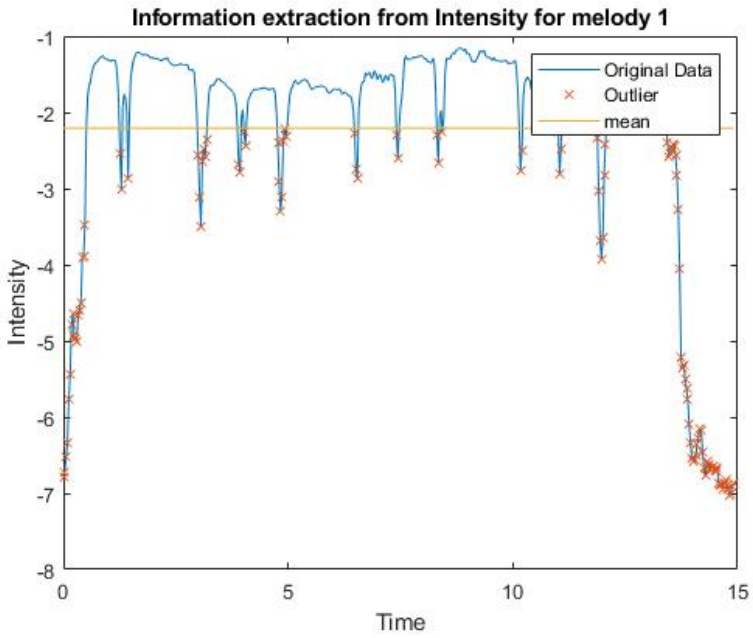

Figure 6: Intensity information extraction with time in seconds

3 Feature Extractor Check

Our feature extractor design can be verified to meet the requirements in the instruction:

1. **They should allow distinguishing between different melodies, i.e., sequences of notes where the ratios between note frequencies may differ.**
   According to equation 2 ratios between note frequencies of the melody decides the combination of semitones. Thus different melodies can be easily distinguished by our feature extraction design.

2. **They should also allow distinguishing between note sequences with the same pitch track, but where note or pause durations differ.**
   Semitones depend on the pitch frequency and not on the time instant at which the pitch occurs. When note or pause durations differ the overall shape of the feature along y-axis remains the same, but stretched or shrinked along the time axis.

3. **They should be insensitive to transposition (all notes in a melody pitched up or down by the same number of semitones).**
   If we have a note sequence $\{f_0, f_1, f_2.....f_n\}$, and we pitch up the sequence by a factor z , output sequence will be $\{z.f_0, z.f_1, z.f_2.....z.f_n\}$.
   According to equation 2
   $k = 12.\log_2 \frac{z.f_k}{z.f_0} + 1 = 12.\log_2 \frac{f_k}{f_0} + 1$
   This proves that our design is insensitive to transposition.

4. **In quiet segments, the pitch track is unreliable and may be influenced by background noises in your recordings. This should not affect the features too much, or how they perceive the relative pitches of two notes separated by a pause.**
   We have set two thresholds for the pitch track to eliminate large or too small values,

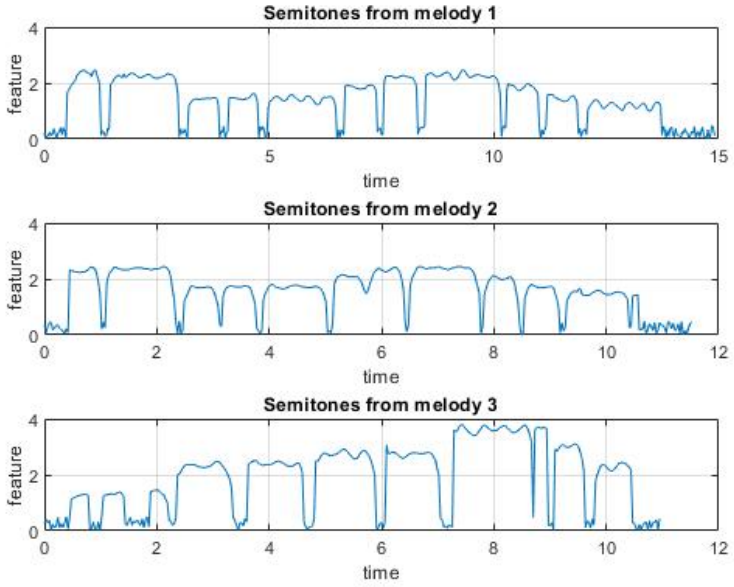

Figure 7: Semitone features with time in seconds

which is useful for detecting the quiet segments. Also we have set threshold value for correlation coefficient which will eliminate out the values with a weak correlation.

5. **They should not be particularly sensitive if the same melody is played at a different volume.**
   Output of our feature extractor design is based on pitch and correlation and not the intensity. Volume is related to intensity and so our design is insensitive to volume changes.

6. **They should not be overly sensitive to brief episodes where the estimated pitch jumps an octave (the frequency suddenly doubles or halves). This is a common error in some pitch estimators, though it does not seem to be particularly prevalent in this melody recognition task.**
   This is a special case of requirement 3 where $z = 0.5$ or $z = 2$. Thus our design is also insensitive to brief episodes where the pitch jumps an octave.

4 Feature Extractor Performance check

Figure 7 shows the extracted features of three melodies together. By observing the plots of the feature sequences we can say that sequence of melody 1 and melody 2 are quite similar to each other, while third one is significantly different. To prove that our feature extractor design is insensitive to transposition we have multiplied pitch track by a scaling factor 1.5. Figure 8 plots the features extracted from the transposed pitch track. It clearly proves that the design is transpose independent.

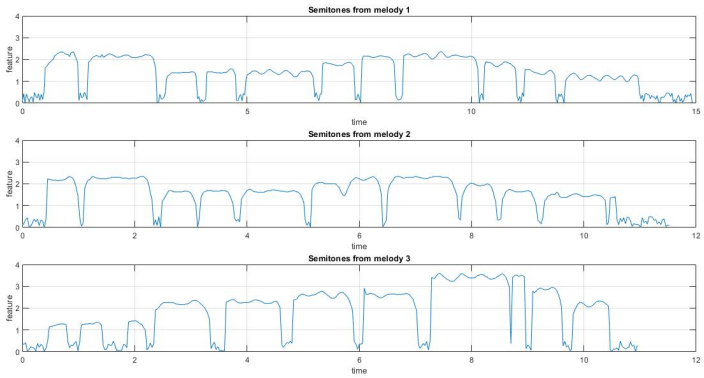

Figure 8: Scaled Semitone features with time in seconds

5 Conclusion

Semitones are the feature extracted from the three components of a song - Pitch frequency, pitch correlation coefficient and intensity. Each semitone can be treated as a value drawn from a Gaussian Distribution whose mean is K (for one octave K varies from 0-12) and standard deviation is 0.5

References

[1] Jiawei Yang, Susanto Rahardja, and Pasi Fränti. "Outlier Detection: How to Threshold Outlier Scores?" In: *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud Computing*. AIIPCC â^{TM}19. Sanya, China: Association for Computing Machinery, 2019. ISBN: 9781450376334. DOI: 10.1145/3371425.3371427. URL: https://doi.org/10.1145/3371425.3371427.