Music emotion classification (design & implementation)

Gordana Vujović

University of Ljubljana
Faculty of computer science and information science
Večna pot 113, Ljubljana
Email: gv673@student.uni-lj.si

Abstract. Music Emotion Classification (MEC) lies on the intersection of these two very interesting and challenging domains of research. As the number of music recordings increases exponentially, the automatic classification of music emotion becomes one of the hot spots on research and engineering. This kind of categorization is particularly important because of its direct link with one of the primary motives to listen to music: to feel emotions. This paper presents music emotion detection based on a statistical model (that give me conclusions about the associations between note value and emotions) and an algorithm for detection tempo that processes audio recordings and determines the beats per minute. The concept of spectral energy flux is defined and leads to an efficient note onset detector, also the digital filter algorithm is defined to facilitate a process of detecting drum and bass sounds as markers for beats. The performance of the proposed method is evaluated using 30 songs from several musical genres. The global recognition rate is 73.7 %. Results are discussed and compared to other tempo estimation system.

Keywords: sound, filters, analysis, emotion classification

1 Introduction

In the past few years, research in Music Emotion Classification (MEC) has been very active to produce automatic classification methods. These algorithms aim to ease the process of annotating data as we are faced with a large and increasing amount of digital music. In fact, a MEC system can even give satisfying results if we consider a few simple categories, if we check for valid agreements between people and if we concentrate on meaningful features.
The goal of this project is to detect the emotions in the song. There are many categories of mood into which songs may be classified, e.g. happy, sad, angry, brooding, calm, uplifting, etc. People listen to different kinds of music depending on their mood. In order to keep the problem simple, we considered two song moods: Happy and Sad. The two musical cues under investigation are tempo and energy unit.

The tempo of a song is usually measured in beats per minute. The tempo of a song, its beats, are usually felt by a listener and drive him, for instance, to dance according to the song's rythm.

The energy (obtained by integrating over the Power Spectral Density) then refers to the spectral energy distribution that would be found per unit time, since the total energy of such a signal over all time would generally be infinite.

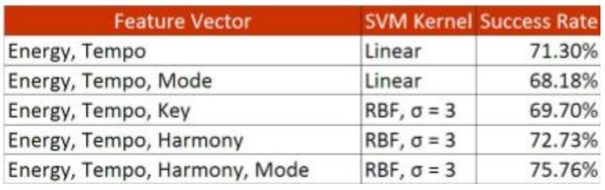
The main problem is how to make the algorithm for detection that get the better results as posssible.
Algorithm for the detection the emotions that is implemented are the combination of the three algorithms:
1. ***Sound Energy Algorithm*** - to detect the sound energy because a sound will be heard as a beat only if his energy is largely superior to the sound's energy history, that is to say if the brain detects a brutal variation in sound energy.
2. ***Low-pass filter and High-pass filter algorithm*** - The aim of this digital filter algorithm is to facilitate a process of detecting drum and base sounds as markers for beats.
3**. *Peak Algorithm*** - to detect peak amplitude changes in multiple parts of the sound spectrum.

This report lists the descripton on the statistical method on which is based our chooses for song moods , describes the design and development process of the algorithms and in the end evaluates its capabilities and discusses further improvements.

2 Releted work

The impact of music on arousal and mood is well established. People often choose to listen music for this very effect (Gabrielsson, 2001), and physiological responses to music differ depending on the type of music heard. Recognizing basic emotions in music such as happiness or sadness is effortless and highly consistent among adults (Peretz et al., 1998).

In the last 10 years, many other studies have assessed the influence of tempo, mostly combined with mode, in affective reactions. (Sotos, Fernández-Caballero and Latorre, 2016) A research work on the impact of individual musical cues in the communication of certain emotions to the listener states the feature vectors beyond energy and tempo didn't add much to the classification process, one must remember that marginal improvement of performance gets successively harder. (Padial and Goel, 2011)



Feature Vector	SVM Kernel	Success Rate
Energy, Tempo	Linear	71.30%
Energy, Tempo, Mode	Linear	68.18%
Energy, Tempo, Key	RBF, σ = 3	69.70%
Energy, Tempo, Harmony	RBF, σ = 3	72.73%
Energy, Tempo, Harmony, Mode	RBF, σ = 3	75.76%

Figure 1 Soft Margin SVM performance for some of the candidate feature sets

In my understanding all these previous studies show a need for going on in experimenting with tempo and energy unit for the sake of regulating emotions through music.

3 METHODS

3.1 *Technology and tools*

I implemented the song emotion detection using Eclipse and the JAVA programming language. The user interface of the song emotion detection is created using JAVA UI. The application used the following additional library:

- package com.jogamp.opengl for fast graphics rendering.
- package com.badlogic.audio for MP3 decoding, audio outputing and FFT spectral analysis.

Using this libraries, the song emotion detection can load audio files of multiple different formats (.wav, .mp3, .aiff, and others) into memory and convert them into the PCM format. This libraries are also used to play the processed (song emotion detection) audio, as well as to showing in the graphic interface the spectrum of the song.

3.2 *Statistical model*

In the research work uses an experiment that are aimed at detecting the individual influential issues related to two basic components of note value, that is, tempo and rhythmic unit. The participants have to indicate how much they feel certain basic emotions while listening to each music excerpt. The rated emotions are "Happiness," "Surprise," and "Sadness." As described before, the tempo is measured according to beats per minute. A very fast tempo, prestissimo, has between 200 and 208 beats per minute, presto has 168 to 200 beats per minute, allegro has between 120 and 168 bpm, moderato has 108 to 120 beats per minute, andante has 76 to 108, adagio has 66 to 76, larghetto has 60 to 66, and largo, the slowest tempo, has 40 to 60. In their experiment, they have decided to use only three different tempos. Tempos to be heard are 90, 120, and 150 bpm, respectively, covering a sufficient range of standard beats. (Sotos, Fernández-Caballero and Latorre, 2016)

In the one-factor repeated measures model, they assume that the variances of the variables are equal. To test this assumption, they used the Mauchly sphericity test (W). The results are the following:

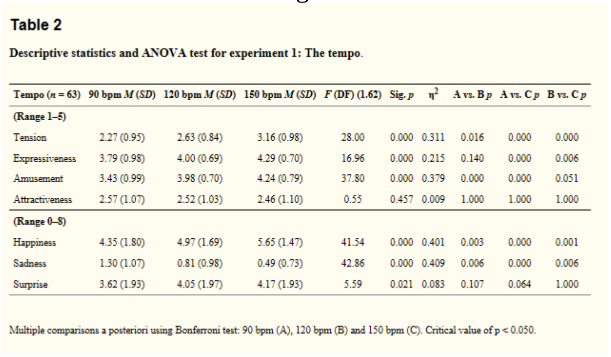


Table 2

Descriptive statistics and ANOVA test for experiment 1: The tempo.

Tempo ($n = 63$)	90 bpm M (SD)	120 bpm M (SD)	150 bpm M (SD)	F (DF) (1.62)	Sig. p	η^2	A vs. B p	A vs. C p	B vs. C p
(Range 1–5)									
Tension	2.27 (0.95)	2.63 (0.84)	3.16 (0.98)	28.00	0.000	0.311	0.016	0.000	0.000
Expressiveness	3.79 (0.98)	4.00 (0.69)	4.29 (0.70)	16.96	0.000	0.215	0.140	0.000	0.006
Amusement	3.43 (0.99)	3.98 (0.70)	4.24 (0.79)	37.80	0.000	0.379	0.000	0.000	0.051
Attractiveness	2.57 (1.07)	2.52 (1.03)	2.46 (1.10)	0.55	0.457	0.009	1.000	1.000	1.000
(Range 0–8)									
Happiness	4.35 (1.80)	4.97 (1.69)	5.65 (1.47)	41.54	0.000	0.401	0.003	0.000	0.001
Sadness	1.30 (1.07)	0.81 (0.98)	0.49 (0.73)	42.86	0.000	0.409	0.006	0.000	0.006
Surprise	3.62 (1.93)	4.05 (1.97)	4.17 (1.93)	5.59	0.021	0.083	0.107	0.064	1.000

Multiple comparisons a posteriori using Bonferroni test: 90 bpm (A), 120 bpm (B) and 150 bpm (C). Critical value of $p < 0.050$.

Figure 2 Results of musical experiment: influence of tempo

With the augmentation of tempo (from 90 to 150 bpm), there is an increase in the mean values of emotions "Happiness" and "Surprise." In addition, there is a decrease in the mean values of "Sadness" emotion.

Moreover, there are significant differences in growth percentages for the parameters when studying the increase/decrease from 90 to 150 bpm.

Indeed, there are emotional factors that suffer large variations, while variation is not so relevant in others. The emotional perception of "Sadness" is the most affected with increasing tempo. The other emotions are less affected.

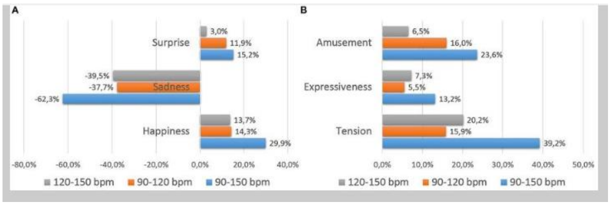


Figure 3 Results of musical experiment: influence of tempo

It seems also useful to investigate the partial variations due to the augmentation from 90 to 120 bpm and from 120 to 150 bpm. Moreover, when the tempo is augmented again (from 120 to 150 bpm), very different growth patterns are observed between the emotions and descriptive scales studied. The perception of emotion "Sadness" decreases by 37.7%, which the highest percentage for all emotions and descriptive words. Other parameters increase their values in a margin from around 12 to 16% when augmenting the pulse from 90 bpm to 120 bpm. Valuations of emotions "Happiness" and "Surprise" rise by 14.3 and 11.9%, respectively

3.3 Algorithms

3.3.1 Sound Energy Algorithm

The algorithm divides the data into blocks of samples, converts into Furier transform and calulate spectrum of block, calculates *flux* as difference between the energy of a current window with the energy of a previous window of blocks. The energy of a block is used to detect a beat.If the energy is above a certain threshold then the block is considered to contain a beat. The threshold is defined on the sort *flux* array as 8 percent of the whole.

```
while (decoder.readSamples(samples) > 0) {
fft.forward(samples);
System.arraycopy(spectrum, 0, lastSpectrum, 0,
spectrum.length);
System.arraycopy(fft.getSpectrum(), 0, spectrum, 0,
spectrum.length);
float flux = 0;
for (int i = 0; i < spectrum.length; i++) {
            float value = (spectrum[i] - lastSpectrum[i]);
            flux += value < 0 ? 0 : value; }
if (maxZaCijeluPjesmu < flux)
            maxZaCijeluPjesmu = flux;
Tuple<Float, Boolean> tuple =new Tuple<Float, Boolean>
(flux, false);
spectralFlux.add(tuple);                    }
mergeSort((ArrayList) spectralFlux);
int top8Posto=(int) (0.08 * (double) spectralFlux .size() );
```

3.3.2 Peak algorithm

The algorithm is based on 4 stages:

1. Searching for the best entering beat
2. Searching for the best ending beat
3. Searching for flux in a surrounding
4. Analyzing the success of the current beat compared to the best so far

In the first stage, I am looking for an entering beat by starting with a sequence that I am obtained from the previous algorithm. The loop will finish when I passed through .
In the second stage, I am looking for the ending beat. I expect that ending beat can be in the three distance from an entering beat and of course entering and ending beat cannot be the same.
In the three-stage, I'm setting the variable *deltaTemp* that is a difference (ending - entering) between index in the sequence where the fluxes are detected. I am passing through the whole sample based on deltaTemp and x_1 (that is the first option for ending beat) and I 'm calculating how many peaks are hit as success and as a failure. This is very important because of the inaccurate resolution of fluxes, he seeks a final peak in a surrounding around the presumed location of the final peak.
In the four-stage, I'm comparing is the Successful Rate (uspjesnostTemp) for 70% better than the previous result and is the number of success detect peak for 1.5 better than the previous result. If the condition is satisfied I'm setting the new value as a candidate for final distance.
On the end, I'm calculating the tempo according to the following equation:

$$tempo = 60/(1024 * finalDistance/sampleRate)$$

```
//stage 1
for (int i0 = 0; i0 <= -2 + 3 * lsIndeksiVrhova.size() / 4;
i0++) {
//stage 2
for (int i1 = i0 + 1; i1 < i0 + 4; i1++) {
    int deltaTemp = lsIndeksiVrhova.get(i1) -
lsIndeksiVrhova.get(i0);
    int opsegGreske = deltaTemp / 4;
    int iKorekcijaProsjekTemp = 0;
    int pogodjenihTemp = 0, promasenihTemp = 0;
    int x1 = lsIndeksiVrhova.get(i1);
    int brojacBeata = 0;
//stage 3
 while (x1 + deltaTemp + opsegGreske < ls0.size()) {
    boolean bPogodjenVrh = false;
 for (i2 = 0; i2 <= opsegGreske; i2++) {
if (ls0.get(x1 + deltaTemp + i2).vrh == true) {
             x1 = x1 + deltaTemp + i2;
             iKorekcijaProsjekTemp += i2;
             bPogodjenVrh = true;
             break; }
 if (ls0.get(x1 + deltaTemp - i2).vrh == true) {
             x1 = x1 + deltaTemp - i2;
             iKorekcijaProsjekTemp -= i2;
             bPogodjenVrh = true;
             break;  }  }
 if (bPogodjenVrh) {
             pogodjenihTemp++;
             ls0.get(x1).NaglasiTemp = true; }
else {
            x1 = x1 + deltaTemp;
             ls0.get(x1).NaglasiTemp = false;}
            brojacBeata++; }
 iKorekcijaProsjekTemp = iKorekcijaProsjekTemp /
            brojacBeata;
int sumaPromAndPogTemp = pogodjenihTemp +
            promasenihTemp;
double uspjesnostTemp = (double) ((double)
            pogodjenihTemp / (double)
            sumaPromAndPogTemp);
//stage 4
if (uspjesnostTemp > 0.7 && pogodjenihTemp >= 1.5 *
            pogodjenihBest) {
deltaBest = deltaTemp;
 pogodjenihBest = pogodjenihTemp;
 promasenihBest = promasenihTemp;
iKorekcijaProsjekBest = iKorekcijaProsjekTemp;
            } }} }
```

3.3.3 Low pass filter algorithm

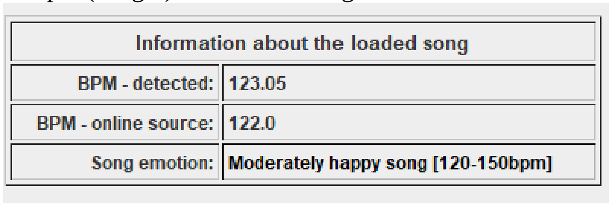
Here, I'll run a low- and high- pass filter over the song and graph the peaks measured from them. Because certain instruments are more faithful representatives of tempo (kick drum, snare drum, claps), I'm hoping that these filters will draw them out more so that "song peaks" will turn into "instrument peaks", or instrument hits.

4 RESULTS

During the evaluation, the algorithm parameters were set as follows. The number of seconds for loading song was set to thirty seconds. The number for refresh a graphic interface during singing the song was set to five times in the second.

To have a better idea of the performance of my algorithm, I decided to compare it with online resources. Mostly I used the results from following website »Song BPM« (https://songbpm.com/vinton-blue-velvet).

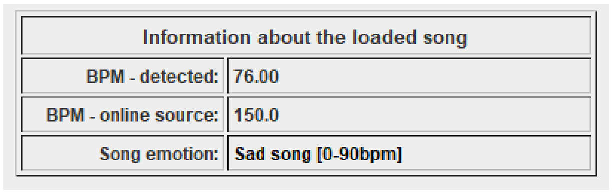
The result, in the song "Röyksopp - This Must Be It.mp3"(song 1) is the following:



Information about the loaded song	
BPM - detected:	123.05
BPM - online source:	122.0
Song emotion:	**Moderately happy song [120-150bpm]**

Figure 4 Song emotion detection – result for song 1

The result, in the song "Royksopp - Something In My Heart.mp3" (song 5) is the following:

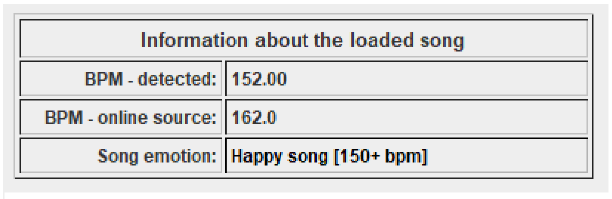


Information about the loaded song	
BPM - detected:	76.00
BPM - online source:	150.0
Song emotion:	**Sad song [0-90bpm]**

Figure 5 Song emotion detection - result for the song 5

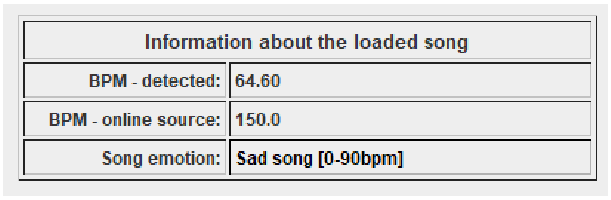
If we only listen to this song, we can hear that an interrupted synthesizer creates an unreal image that the song is faster than it is. The input and output beat are double smaller. So with my algorithm is better results.

The result, in the song "Weird Al Yankovic - Polka face.mp3" (song 6) is the following:



Information about the loaded song	
BPM - detected:	152.00
BPM - online source:	162.0
Song emotion:	**Happy song [150+ bpm]**

Figure 6 Song emotion detection - result for song 6
The result, in the song "Procol harum - A whiter shade of pale.mp3" (song 7) is the following:



Information about the loaded song	
BPM - detected:	64.60
BPM - online source:	150.0
Song emotion:	**Sad song [0-90bpm]**

Figure 7 Song emotion detection -result for song 7
This is an example where the ending beat is too quiet and it didn't detect in the 8% of the highest flux detection.

5 CONCLUSION

This paper has focused on emotion detection through some musical parameters.
In this paper, I have presented an efficient beat detect algorithm that processes audio recordings and detect the emotion. I have also defined the concept of spectral energy flux and used it to derive a new and effective onset detector on which is based beat algorithm. In addition, the proposed tempo detection system is straightforward to implement and has a relatively low computational cost. Considering the subjective nature of mood classification, I believe that 73.7% success is a good result. The success of our algorithm is comparable to the results obtained by different research groups around the world.
Future work should explore other flux estimation techniques and to analysis the results on the larger dataset would allow us greater freedom in playing around with different ways of capturing the tempo and emotion.

REFERENCES

[1] R.W. Hamming, Digital filters, 3rd~ed, Englewood Cliffs, NJ: Prentice Hall, 1989.

[2] http://archive.gamedev.net/archive/reference/programming/features/beatdetection/index.html (Accessed on 31.10.2018)

[3] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4971092/ (Accessed on 03.11.2018)

[4] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3227856/ (Accessed on 03.11.2018)

[5] http://joesul.li/van/beat-detection-using-web-audio/ (Accessed on 31.10.2018)

[6] Miguel Alonso, Bertrand David Richard (2004)Tempo and beat estimation of musical signals https://www.researchgate.net/publication/200806034_Tempo_and_Beat_Estimation_of_Musical_Signals

[7] Gabrielsson A. (2001). Emotions in strong experiences with music, in Music and Emotion: Theory and Research, eds Juslin P., Sloboda J., editors. (New York, NY: Oxford University Press;), 431–449.

[8] Peretz I., Gagnon L., Bouchard B. (1998). Music and emotion: perceptual determinants, immediacy, and isolation after brain damage.https://www.sciencedirect.com/science/article/pii/S0010027798000432?via%3Dihub

[9] http://cs229.stanford.edu/proj2011/GoelPadial-MusicMoodClassification.pdf (Accessed on 31.10.2018)

[10] http://joesul.li/van/beat-detection-using-web-audio/ (Accessed on 01.01.2018)

[11] http://www.cs.man.ac.uk/~nikolaon/~nikolaon_files/NikosNikolaou_MusicEmotionClassification_ECE_TUC.pdf (Accessed on 31.11.2018)