STEM Fellowship Big Data Challenge 2018-2019

A Study on the Method of Identifying Relaxing Music

January 26, 2019

Abstract

Music has been an indispensable element in human culture. This study aimed to summarize the general characteristics of relaxing music, and ultimately, find a method to identify relaxing songs. SVM (Support Vector Machine) and RNN (Recurrent Neural Network) methods were used to build the classification models for relaxing music. Further tests of the two models were also carried out to determine their limitations.

The audio data in this research paper was collected from certain websites as some of them are scientifically proved as relaxing. Tempograms were computed in order to compare the characteristics of relaxing and non-relaxing music. The training data for classification models was obtained through feature extraction from the Mel-spectrogram each song. The two models were then tested with some songs to further test their ability to classifying relaxing and non-relaxing music. Other tests on the trend of the classification score as the tempo of the music increases was also done with linear regression model.

The results show that both classification models work well for songs that are similar to the training data. However, some unexpected results occurred for some music with unfamiliar characteristics, including audio with only uniform beats. A question on the possibility of summarizing all the characteristics of relaxing music is raised up at the end.

Keywords
Relaxing, Tempogram, SVM, RNN.

Introduction

While most people are familiar with music as entertainment, it is not well known that music can also be used to improve human health in several domains such as improving heart rate and reducing anxiety levels through music therapy [1] . The song "Weightlessness" created by Marconi Union was found to be the most relaxing music in the world as it could reduce people's overall anxiety by 65 percent and their usual physiological resting rates by 35 percent [2]. Another study found that the relaxing effects of "Weightlessness" has something to do with its tempo, where the heart rate gradually slows down to match the tempo of the song [3].

The purpose of this research paper is to summarize the general characteristics of relaxing music, and ultimately, find a method to identify relaxing songs. The general characteristics of relaxing music were analyzed by comparing the Tempograms (spectrograms that show the change of tempo over time) of selected relaxing song and non-relaxing song. The identification of relaxing songs was done by using machine learning methods including SVM (Support Vector Machine) and RNN (Recurrent Neural Network).

The methods of machine learning worked very well in the field of music emotions where the model was trained to recognize the associated emotion of the song by using classification algorithms [4]. In this research paper the outcomes are simplified into relaxing and non-relaxing. The definitions of those two terms were taken as the top ranked relaxing and non-relaxing songs found from the internet with only the relaxing songs scientifically proved [2, 5]. Further tests of those two models were also done to seek their limitations.

A successful binary model of identifying relaxing music can be used by sound therapists to identify the desired music for specific clients. In the case of long-term, long-distance space travel, a quick and accurate classifier of relaxing and non-relaxing music is also very important as music could play a significant role in adjusting the traveller's physiological state such as heart rate

and anxiety level. For example, listening to non-relaxing music before sleep could affect the sleep quality.

Materials & Methods

The top ten most relaxing songs as scientifically proved and top ten most non-relaxing songs as found from internet were taken as audio data, see row 1-20 in Table 1 [2, 5]. Some of those songs contain English lyrics, which will be discussed later.

The Tempograms of the song "Weightlessness" as relaxing music and the song "Frankie Teardrop" as non-relaxing music were computed over a time domain of 60 seconds. The Tempograms used auto-correlation method with measuring the tempo at the Measure pulse level. Jupyter notebook was used and the code was written in python, see Figure 10. The reference code comes from the Tempo Estimation section on Python's MIR (Music Information Retrieval) site [6]. The actual code used was modified including some important parameters, see Figure 10.

The features of the music were extracted with using the MFCC (Mel Frequency Cepstral Coefficient) method, where a spectral envelope could be represented with a small set of numbers, see Music Structure Analysis section on Python's MIR site [6]. The method basically extracts important features from the Melspectrogram corresponding to each audio, see Figure 3 and 4 as examples of Melspectrogram. The Melspectrogram is different from Tempogram mostly by its frequency y-axis and a different method of power map [6]. In this research paper, the MFCC was taken as 12, so each small observation (spectral envelope) of the audio can be represented with 12 numbers. The input as extracted features is an array with n rows and 12 columns, where n represents the number of observations. See Figure 11 in the Appendix for the actual code used to compute Melspectrograms.

SVM model was trained with the extracted features as input. Since the model was supervised, all relaxing songs were labelled 0 and all non-relaxing songs were labelled 1. Each observation is a point in a 12-dimensional space. A calculated hyper-plane was used to separate points that were labeled 0 and 1 into two spaces. The reference code is from the Genre Recognition section on Python's MIR site [6]. The actual code was greatly modified with the ability of storing the model, see Figure 12 and 13 in the Appendix. The model was first tested with the same input data, see Table 2. The model was then tested with new songs, see Table 3. The details of the new songs are in Table 2.

The RNN model was trained with the same extracted features of the training songs. See Figure 5 for the architecture of the network. The reference code is from the Neural Network section on Python's MIR site [6]. The actual code was greatly modified, see Figure 14 and 15 in the Appendix for more details. The loss function of the model was printed, see Figure 6. Note that the x-axis of the loss function graph represents the number of epochs. The model was also tested with the same new songs from SVM test, see Table 3. A bar diagram was computed to visualize the difference in the score between SVM and RNN models, see Figure 7. Note that the y-axis represents the difference in score.

A further test on the effects of music's tempo was carried out. Audio files with only uniform beats were tested with yielded scores from both models. A regression model was obtained for each model, see Figure 8 and 9. For more details of the two regression models, see Figure 16 and 17 in the Appendix.

Results

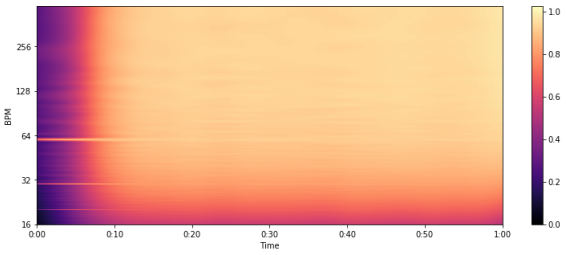

Figure 1: Tempogram of "Weightless" (Relaxing Music R-1)

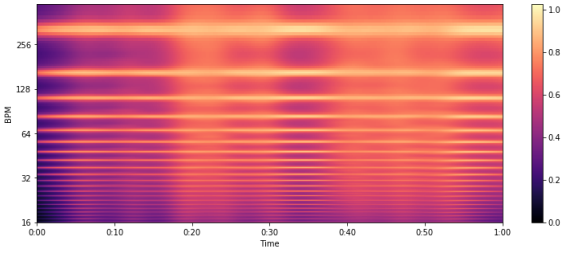

Figure 2: Tempogram of "Frankie Teardrop" (Non-relaxing Music N-6)

Table 1 lists all the training audio and testing audio data for the later classification methods with some relevant details included. For training audio data (row 1-20), the first column of the

No.	Song Title (Artist)	Lyrics	Time
N-1	You Are Not A Riot (The Coup)	English	3:14
N-2	Reverence (The Jesus and Mary Chain)	English	3:39
N-3	My Friend Goo (Sonic Youth)	English	2:27
N-4	Stresses (The Blaggers ITA)	English	3:38
N-5	Accelerator (Primal Scream)	English	3:41
N-6	Frankie Teardrop (Suicide)	English	10:36
N-7	Violet (Hole)	English	3:42
N-8	Debaser (Pixies)	English	2:56
N-9	You Made Me Realise (My Bloody Valentine)	English	3:46
N-10	Surface Envy (Sleater Kinney)	English	3:06
R-1	Weightless (Marconi Union)	N/A	8:08
R-2	Electra (Airstream)	N/A	6:13
R-3	Mellomaniac Chillout Mix (DJ Shah)	N/A	5:39
R-4	Watermark (Enya)	N/A	2:26
R-5	Strawberry Swing (Coldplay)	English	4:14
R-6	Please Don't Go (Barcelona)	English	4:31
R-7	Pure Shores (All Saints)	English	4:09
R-8	Someone Like You (Adele)	English	4:44
R-9	Canzonetta Sull'aria (Mozart)	Italian	3:34
R-10	We Can Fly (Rue du Soleil)	N/A	6:09
r-1	Rainbow (Jay Chou)	Mandarin	4:33
r-2	Chopin (Nocturne)	N/A	4:30
r-3	Bird sounds (N/A)	Bird sounds	5:43
r-4	The Promise of the World (Chieko Baisho)	Japanese	4:08
r-5	The Rain (Joe Hisaishi)	N/A	5:39
n-1	Electryfying (Jim Johnston)	N/A	2:29
n-2	Lose Yourself (Eminem)	English	5:17
n-3	Panda (Desiigner)	English	4:08
n-4	Zenzenzense (RADWIMPS)	Japanese	4:53
n-5	I Want Summer (Jay Chou)	Mandarin	3:50

Table 1: Relaxing and Non-Relaxing Songs

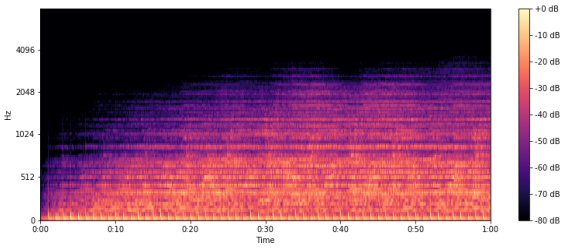

Figure 3: Mel spectrogram of "Weightless" (Relaxing Music R-1)

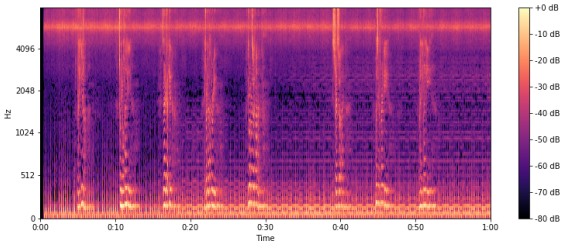

Figure 4: Mel spectrogram of "Frankie Teardrop" (Non-relaxing Music N-6)

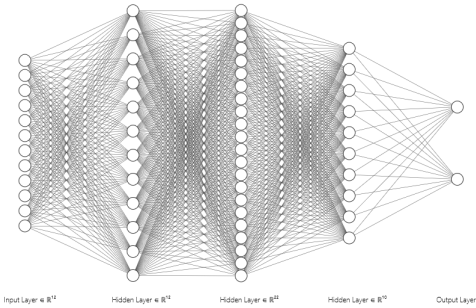

Figure 5: RNN Architecture

table shows either the song is relaxing (R) or non-relaxing (N) with a number indicating its rank;

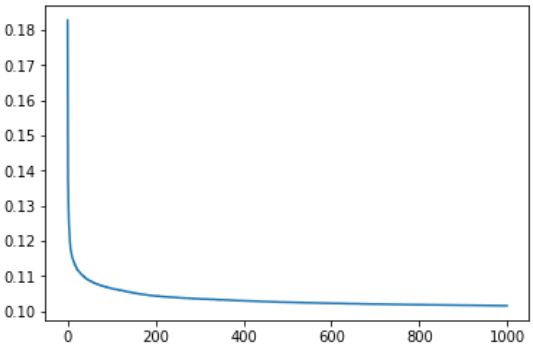

Figure 6: RNN Loss Function

for testing audio data (row 21-30), the first column shows either the song is relaxing (r) or

No.	Song Title	Score
R-1	Weightless	0.003
R-2	Electra	0.008
R-3	Mellomaniac (Chillout Mix)	0.001
R-4	Watermark	0.013
R-5	Strawberry Swing	0.044
R-6	Please Don't Go	0.021
R-7	Pure Shores	0.150
R-8	Someone Like You	0.020
R-9	Canzonetta Sull'aria	0.018
R-10	We Can Fly	0.020
N-1	You Are Not A Riot	0.861
N-2	Reverence	0.993
N-3	My Friend Goo	0.984
N-4	Stresses	0.973
N-5	Accelerator	0.983
N-6	Frankie Teardrop	0.991
N-7	Violet	0.905
N-8	Debaser	0.963
N-9	You Made Me Realise	0.991
N-10	Surface Envy	0.959

Table 2: The Performance of SVM on the Training Data Set

non-relaxing (n) with a number indicating its index (not rank) [2,5]. Note that the identification of relaxing or non-relaxing for the testing songs was based on the reseachers' perception, and it will be discussed later. Figure 1 and 2 are the Tempograms computed for relaxing and non-relaxing songs. Figure 3 and 4 are examples of Melspectrograms which were used to extract features for machine learning; they also provide a good contrast between Tempograms and Melspectrograms. Table 2 lists the pre-testing results of SVM model on each song in the training data set. Figure 5 shows the architecture of the Neural Network model used. Figure 6 shows the loss function of the Neural Network model in regards to the training data, which is similar to the pre-testing results of SVM model. Table 3 shows the testing results of new songs from both machine learning methods. Figure 7 shows the difference in score between the two models for new songs. Figure 8 and 9 show the trend of the score as the tempo of the music increases for both machine learning models. Figures found in the Appendix contain the details of the code and the two regression models.

Discussion

General Characteristics of Relaxing Music

"Weightlessness" was selected as the relaxing song because it had been precisely determined as the most relaxing song in the world [2]. The song "Frankie Teardrop" was selected as the non-relaxing song because only a small component of the song contains lyrics, which the effects of lyrics could be controlled. Because of the complexity of human voice, it is hard to track the tempo where lyrics are present.

As mentioned before, the tempo of a piece of music is the key of determining how relaxing it is. However, tempo changes significantly in some songs and that creates great inaccuracy for representing the overall tempo with a single value in beats per minute. Also, tempo can be measured at different pulse levels (Measure, Tactus, Tatum) and the same song could have correct but different tempos if the pulse levels were chosen differently [7]. Tempogram as a spectrogram is able to show the change in tempo over time and the music's tempo at different pulse levels. Thus, Tempogram is an ideal tool for analyzing the relaxing song's tempo characteristics. The Tempograms in Figure 1 and 2 were computed using auto-correlation method which measures the tempo at Measure level. Another well-known Tempogram is Fourier-Tempogram which measures the tempo at Tatum level [7].

From Figure 1 and 2, it could been seen that relaxing music has a smaller tempo than non-relaxing music. The tempo with the highest amplitude of "Weightlessness" is about 60 bpm throughout the song, where the tempo of the song "Suicide" is much higher, above 256 bpm. Another characteristic of relaxing music which could been seen from the Tempograms is that relaxing music's major tempo doesn't have a very high amplitude. There seems to be a "harmony" in "Weightlessness" where its 60 bpm tempo line is only a little bit more intense than the other tempos. In contrast, the tempo lines in "Frankie Teardrop" are very intense. Similar analysis of the other songs yielded the same results.

Feature Extraction

The features of a song were extracted from its Melspectrogram, see Figure 3 and 4 as examples. As comparing to Auto-Correlation Tempogram, Melspectrogram has frequency as its y-axis. The Melspectrogram also has a different spectrum scale which is more close to human ear perception [8]. Note that there is an uncertainty principle associated with the Mel-

R/N	Song Title	SVM Score	RNN Score
r-1	Rainbow	0.103	0.177
r-2	Chopin	0.003	0.019
r-3	Bird Sounds	0.296	0.533
r-4	The Promise of the World	0.331	0.456
r-5	The Rain	0.085	0.100
n-1	Electryfying	0.955	0.884
n-2	Lose Yourself	0.631	0.562
n-3	Panda	0.672	0.660
n-4	Zenzenzense	0.809	0.792
n-5	I Want Summer	0.586	0.512

Table 3: RNN and SVM Test Results

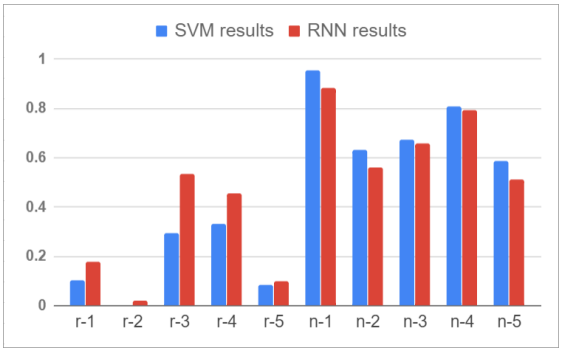

Figure 7: Test Results Comparison

spectrogram; the more certainly the frequency is known, the less certainly the frequency's time domain locality is known. The parameters were set properly for Melspectrograms, see the Appendix for more details.

As briefly mentioned before, the input data as the features of a song is an array with n rows and 12 columns. Each row is an observation and each column is one of the 12 feature extractions for each observation. Such an array is able to represent most of the features of a song.

Scores from the Classification Models

As mentioned before, all the relaxing song observations as training data were scaled to 0 and all the non-relaxing song observations were scaled to 1. The score of a song as number between 0 and 1 could be interpreted as that the more close the score is to 0, the more relaxing the song is, otherwise more non-relaxing.

The music selected for testing include a variety of songs. Some of them are close to the training songs, some of them are very different. For example, the songs "r-1", "r-4", "n-4", and "n-5" contain lyrics that are not in English which is the only lyrics language in the training songs. However, the scores for those four songs generally agree with our personal identification. "r-4" has the highest difference in score among the four songs, this may due to its high vocal pitch.

The songs "r-2", "r-5", "n-1", "n-2" and "n-3" are similar to the training data. "r-2" has piano as the major instrument and it is very close to "R-4"; "r-5" has mainly string instruments and it is very close to "R-4" and "R-9" 's background music; "n-1", "n-2" and "n-3" have hard beats which are similar to most of the non-relaxing songs from training data. Those five songs generally agree with the personal identification. Note that "n-1" has the highest score my due to that it contains a large component of electric sound. Also note that "n-2" and "n-3" have lower scores may due to its rapping content with little electric sound. For example, "n-4" with much higher score has a large component of electric sound. The high difference in score for "n-2" may also due to its rapping content with complex human voice.

The song "r-2" has the highest difference in score. This may due to its unfamiliar content which the sound of the bird is not included in any of the training songs. This can be regarded as one of the limitations of the models.

Another interesting observation from Figure 7 is that SVM model generally yields scores close to the two extremes (0 and 1) while RNN model generally yields scores close to the middle (0.5). The reason for this phenomenon is unknown.

Trend with Higher Tempo

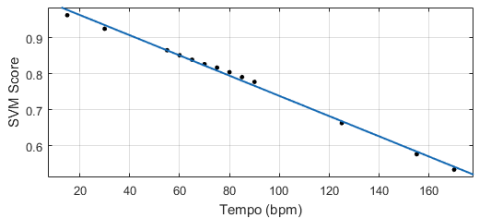

Figure 8: SVM Model Tempo Trend

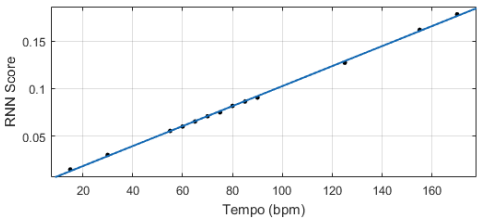

Figure 9: RNN Model Tempo Trend

As proposed before, one of the general characteristics of relaxing music is its relatively smaller tempo value. In order to test the validity of this hypothesis based on Tempograms, audio files with only uniform beats were tested [9].

As it could been seen from Figure 8, as the tempo of the music increases, the score decreases, which means the music becomes more relaxing. The R-squared value is extremely close to 1, means that in the SVM model, the increase in the tempo is very correlated with the decrease in the score.

From Figure 9, it could be seen that as the tempo of the music increases, the score increases. The R-squared value is extremely close to 1, means that in the RNN model, the increase in the tempo is very strongly correlated with the increase in the score.

This nearly perfect linear trend for both models are certainly unexpected. A possible cause might be that the SVM model recognizes all the pauses between beats as non-relaxing and all the beats as relaxing, while the RNN model recognizes all the pauses as relaxing and all the beats as non-relaxing. Since the majority of the audio is pause and the percentage of beats increases as the labeled tempo increases, SVM model will yield a negative trend with overall larger scores and RNN model will yield a positive trend with overall smaller scores.

This result certainly couldn't be used to prove the tempo characteristics of relaxing music since the perfect linear relationship clearly indicates some abnormal phenomenons.

Sources of Error

Most of the training data for the classification models in this research paper is not scientifically as either relaxing or non-relaxing. Those songs were selected based on the given information from certain websites. The identification of testing data is also very biased which depends on the researchers' perception. Thus, the definitions of relaxing and non-relaxing are vague, which could be properly defined with some physiological experiments.

The lyrics as human voice certainly plays a role in identifying relaxing songs. For example, English-speaking people would mentally react to certain words in the lyrics while a person who doesn't know English wouldn't react to them at all. Some techniques of speech recognition could be combined with music recognition model based on the language background the the listener.

The feature extraction for training the two models in this research paper might cause some problems since it skips some parts of the music.

Conclusions

Research Implications

The machine learning models in this research paper worked well with certain limitations identified. Although it was realized later that an emotion diagram would be a better choice than just "relaxing" and "non-relaxing", machine learning models in this research paper still yielded mostly satisfactory results [4].

The interesting point is that machine learning models are able to imitate human hearing perception to a certain degree with extremely limited audio data. This might challenge most people's ideas that music's effects on human perception are too complex for modelling. However, the general characteristics of relaxing music are still not fully known even machine learning could yield expected results. The results raised a question on the possibility of summarizing all the characteristics of relaxing music, as they are hidden inside the "black box" of machine learning.

Future Improvement

In this research paper, RNN was used as the Neural Network Model. For future studies, CNN (Convolutional Neural Network) could be used as the model the the input of the whole spectrogram of a song. More important features of the music could be extracted with CNN model.

Also, the explanation in this research paper for the unexpected tempo trend was solely based on the regression model. For a more reliable explanation, the original arrays should be carefully analyzed. The CNN model may yield a different result since it extracts more features.

The training data for the models in this research paper is very limited and unreliable. Physiological experiments could be carried on to support the reliability of the training data. An example could be the hybrid Neural Network which worked in a study on music emotions [4].

The causes of the phenomenon on the difference in score between two models are still unexplained. The trend of each model's scores remained unjustified. This phenomenon might heavily depend on the given training data and a further analysis is required.

The term "relaxing" and "non-relaxing" in this research paper could be replaced with more accurate emotion terms such as "arousal" and "valence" [4]. A better and more complex emotion diagram could be used to build the classification models.

Acknowledgements

We would like to thank our school teacher Mrs. Wells for offering suggestions on this research paper. We would also like to thank our mentor Curtis Chong for helping us with formatting.

References

[1] McCaffrey T, Edwards J, Fannon D (2011). Is there a role for music therapy in the recovery approach in mental health. The Arts in Psychotherapy. Retrieved from https://ulir.ul.ie/handle/10344/3362

[2] Curtin, M. (2017, May 30). Neuroscience Says Listening to This Song Reduces Anxiety by Up to 65 Percent. Retrieved from https://www.inc.com/melanie-curtin/neuroscience-says-listening-to-this-one-song-reduces-anxiety-by-up-to-65-percent.html

[3] Agrawal A, Makhijani N, Valentini P. (2013). The effect of music on heart rate. J Emerg Investig. Retrieved from https://www .msjonline.org/index.php/ijrms/article/view/3734

[4] Vempala, N. N., Russo, F. A. (2018). Modeling Music Emotion Judgments Using Machine Learning Methods. Frontiers in Psychology, 8.doi:10.3389/fpsyg.2017.02239. Retrieved from https://ww w.ncbi.nlm.nih.gov/pmc/articles/P 560/

[5] Noisy Songs Top 10 Countdown. (2016, February 25). Retrieved from http://www.drunkenwerewolf.com/features/top-ten-noisy-songs/

[6] Notes on Music Information Retrieval¶. (n.d.). Retrieved from https://musicinformationretrieval.com/index.html

[7] MLLER, M. (2016). FUNDAMENTALS OF MUSIC PROCESSING: Audio, analysis, algorithms, applications. SPRINGER. Retrieved from https://www.audiolabs-erlangen.de/content/resources/MIR/00-2017_CourseMIR_ HfM-Karlsruhe/2017_MuellerWeiss_MIR_BeatTracking_handouts.pdf

[8] Mel-frequency cepstrum. Retrieved from https://en.wikipedia.org/wiki/Mel-frequency_cepstrum

[9] Metronome. (2013, February 18). Retrieved from https://www.youtube.com/watch?v=Ka-XEIYVzfQlist=PLR-s-bpe_DUGVBKBxyz_ otdGQTknKuunW

Appendix

```
%matplotlib inline
import seaborn
import numpy, scipy, matplotlib.pyplot as plt, IPython.display as ipd
import librosa, librosa.display

plt.rcParams['figure.figsize'] = (13, 5)
x, sr = librosa.load('Music/R-Weightlessness.wav',22050,True,
                     duration = 60, offset = 0)

hop_length = 200
onset_env = librosa.onset.onset_strength(y=x,
                sr = sr, hop_length = hop_length, n_fft=2048)
tempogram = librosa.feature.tempogram(onset_envelope = onset_env,
                sr = sr, hop_length = hop_length, win_length = 1500)

librosa.display.specshow(tempogram,
                sr = sr, x_axis = 'time', y_axis = 'tempo')
plt.clim(0, 1.025)
plt.colorbar()
```

Figure 10: Tempogram Code

```
%matplotlib inline
import seaborn
import numpy, scipy, matplotlib.pyplot as plt, IPython.display as ipd
import librosa, librosa.display

plt.rcParams['figure.figsize'] = (13, 5)
y, sr = librosa.load('Music/R-Weightlessness.wav',
                     22050,True, duration = 60, offset = 0)

#D = numpy.absolute(librosa.stft(y))**2
#S = librosa.feature.melspectrogram(S=D)

S = librosa.feature.melspectrogram (y=y,
                    sr=sr, n_mels=128, fmax=8000)

librosa.display.specshow(librosa.power_to_db(S, ref = numpy.max),
                    fmax = 8000, x_axis = 'time', y_axis = 'mel')
#plt.clim(0, 1.025)
plt.colorbar(format = '%+2.0f dB')
```

Figure 11: Mel Spectrogram Code

```
import numpy, scipy, matplotlib.pyplot as plt, sklearn, librosa,
mir_eval, IPython.display as ipd, urllib
import librosa.display

from keras.models import Sequential
from keras.layers.core import Dense
import keras.optimizers

scaler = sklearn.preprocessing.StandardScaler()
load_duration = 10
load_offset = 30

def load_scale(filename, d = load_duration, o = load_offset):
    x, sr = librosa.load(filename)
    mfcc = librosa.feature.mfcc(x, sr=sr, n_mfcc=n_mfcc).T
    return scaler.transform(mfcc)


filename_brahms = 'Music/R-DJ.wav'
x_brahms, sr_brahms = librosa.load(filename_brahms)

n_mfcc = 12
mfcc_brahms = librosa.feature.mfcc(x_brahms, sr=sr_brahms,
                                   n_mfcc=n_mfcc).T
mfcc_1 = scaler.fit_transform(mfcc_brahms)

mfcc_2 = load_scale('Music/R-Weightlessness.wav', 30, 0)
mfcc_3 = load_scale('Music/R-someone.wav', 30, 0)
mfcc_4 = load_scale('Music/R-Electra.wav', 30, 0)
mfcc_5 = load_scale('Music/R-Mozart.wav', 30, 0)
mfcc_6 = load_scale('Music/R-strawberry.wav', 30, 0)
mfcc_7 = load_scale('Music/R-pure.wav', 30, 0)
mfcc_8 = load_scale('Music/R-watermark.wav', 30, 0)
mfcc_9 = load_scale('Music/R-Wecan.wav', 30, 0)
mfcc_10 = load_scale('Music/R-please.wav', 30, 0)

mfcc_21 = load_scale('Music/N-violet.wav', 30, 0)
mfcc_22 = load_scale('Music/N-Acc.wav', 30, 0)
mfcc_23 = load_scale('Music/N-Blagger.wav', 30, 0)
mfcc_24 = load_scale('Music/N-bloody.wav', 30, 0)
mfcc_25 = load_scale('Music/N-envy.wav', 30, 0)
mfcc_26 = load_scale('Music/N-myfriend.wav', 30, 0)
mfcc_27 = load_scale('Music/N-Pix.wav', 30, 0)
mfcc_28 = load_scale('Music/N-Rev.wav', 30, 0)
mfcc_29 = load_scale('Music/N-Riot.wav', 30, 0)
mfcc_30 = load_scale('Music/N-suicide.wav', 30, 0)

features_zero = numpy.vstack((mfcc_1,mfcc_2,mfcc_3, mfcc_4,
            mfcc_5, mfcc_6, mfcc_7, mfcc_8, mfcc_9,mfcc_10))
features_one = numpy.vstack((mfcc_21,mfcc_22,mfcc_23,mfcc_24,
            mfcc_25,mfcc_26,mfcc_27,mfcc_28,mfcc_29,mfcc_30))

features = numpy.vstack((features_zero, features_one))
labels = numpy.concatenate((numpy.zeros(len(features_zero)),
                            numpy.ones(len(features_one))))
model = sklearn.svm.SVC()
model.fit(features, labels)

mfcc_brahms_test_scaled = load_scale('Music/Marnie.wav')

features_test = numpy.vstack(mfcc_brahms_test_scaled)
labels_test = numpy.ones(len(features_test))
predicted_labels = model.predict(features_test)
score = model.score(features_test, labels_test)
print(score)

predicted_labels = model.predict(mfcc_brahms_test_scaled)
print(predicted_labels[1:50])

import pickle
file = open("NetworkData.txt", "wb")
file.write(pickle.dumps(model))
```

Figure 12: Generating SVM Code

```
import numpy, scipy, matplotlib.pyplot as plt, sklearn, librosa,
mir_eval, IPython.display as ipd, urllib
import librosa.display

from keras.models import Sequential
from keras.layers.core import Dense
import keras.optimizers

import pickle
file = open("SVM.txt", "rb")
model = pickle.loads(file.read())

scaler = sklearn.preprocessing.StandardScaler()
load_duration = 30
load_offset = 0

def load_scale(filename, d = load_duration, o = load_offset):
    x, sr = librosa.load(filename)
    mfcc = librosa.feature.mfcc(x, sr=sr, n_mfcc=n_mfcc).T
    return scaler.transform(mfcc)

filename_brahms = 'Music/R-DJ.wav'
x_brahms, sr_brahms = librosa.load(filename_brahms)

n_mfcc = 12
mfcc_brahms = librosa.feature.mfcc(x_brahms, sr=sr_brahms,
                    n_mfcc=n_mfcc).T
mfcc_1 = scaler.fit_transform(mfcc_brahms)

mfcc_brahms_test_scaled =
load_scale('Music/tempo/n-90bpm.wav')

features_test = numpy.vstack(mfcc_brahms_test_scaled)
labels_test = numpy.ones(len(features_test))
predicted_labels = model.predict(features_test)
score = model.score(features_test, labels_test)
print(score)
```

Figure 13: Running SVM Code

```
from keras.models import Sequential
from keras.layers.core import Dense
import keras.optimizers
import numpy
import numpy, scipy, matplotlib.pyplot as plt, sklearn, librosa,
mir_eval, IPython.display as ipd, urllib
import librosa.display

model = Sequential()
model.add(Dense(12, activation='relu', input_dim=12))
model.add(Dense(22, activation='relu'))
model.add(Dense(10, activation='relu'))
model.add(Dense(2, activation='softmax'))

optimizer = keras.optimizers.SGD(decay=0.001, momentum=0.99)

model.compile(loss='binary_crossentropy', optimizer=optimizer)

scaler = sklearn.preprocessing.StandardScaler()
load_duration = 10
load_offset = 30

def load_scale(filename, d = load_duration, o = load_offset):
    x, sr = librosa.load(filename)
    mfcc = librosa.feature.mfcc(x, sr=sr, n_mfcc=n_mfcc).T
    return scaler.transform(mfcc)

filename_brahms = 'Music/R-DJ.wav'
x_brahms, sr_brahms = librosa.load(filename_brahms)

n_mfcc = 12
mfcc_brahms = librosa.feature.mfcc(x_brahms, sr=sr_brahms,
                    n_mfcc=n_mfcc).T
mfcc_1 = scaler.fit_transform(mfcc_brahms)
mfcc_2 = load_scale('Music/R-Weightlessness.wav', 30, 0)
mfcc_3 = load_scale('Music/R-someone.wav', 30, 0)
mfcc_4 = load_scale('Music/R-Electra.wav', 30, 0)
mfcc_5 = load_scale('Music/R-Mozart.wav', 30, 0)
mfcc_6 = load_scale('Music/R-strawberry.wav', 30, 0)
mfcc_7 = load_scale('Music/R-pure.wav', 30, 0)
mfcc_8 = load_scale('Music/R-watermark.wav', 30, 0)
mfcc_9 = load_scale('Music/R-Wecan.wav', 30, 0)
mfcc_10 = load_scale('Music/R-please.wav', 30, 0)

mfcc_21 = load_scale('Music/N-violet.wav', 30, 0)
mfcc_22 = load_scale('Music/N-Acc.wav', 30, 0)
mfcc_23 = load_scale('Music/N-Blagger.wav', 30, 0)
mfcc_24 = load_scale('Music/N-bloody.wav', 30, 0)
mfcc_25 = load_scale('Music/N-envy.wav', 30, 0)
mfcc_26 = load_scale('Music/N-myfriend.wav', 30, 0)
mfcc_27 = load_scale('Music/N-Pix.wav', 30, 0)
mfcc_28 = load_scale('Music/N-Rev.wav', 30, 0)
mfcc_29 = load_scale('Music/N-Riot.wav', 30, 0)
mfcc_30 = load_scale('Music/N-suicide.wav', 30, 0)

features_zero = numpy.vstack((mfcc_1,mfcc_2,mfcc_3, mfcc_4,
            mfcc_5, mfcc_6, mfcc_7, mfcc_8, mfcc_9,mfcc_10))
features_one = numpy.vstack((mfcc_21,mfcc_22,mfcc_23,mfcc_24,
            mfcc_25,mfcc_26,mfcc_27,mfcc_28,mfcc_29,mfcc_30))

features = numpy.vstack((features_zero, features_one))
labels_zero = numpy.array([
    [0, 1]
    for i in range(len(features_zero))
])
labels_one = numpy.array([
    [1, 0]
    for i in range(len(features_one))
])
labels = numpy.vstack((labels_zero, labels_one))
#X_train = numpy.random.randn(10000, 2)

#y_train = numpy.array([
    #[float(x[0]*x[1] > 0), float(x[0]*x[1] <= 0)]
    #for x in X_train
#])

results = model.fit(features, labels,
            epochs=1000, batch_size=100)

import pickle
file = open("NNData.txt", "wb")
file.write(pickle.dumps(model))

plt.plot(results.history['loss'])
```

Figure 14: Generating RNN Code

```
import numpy, scipy, matplotlib.pyplot as plt, sklearn, librosa,
mir_eval, IPython.display as ipd, urllib
import librosa.display

from keras.models import Sequential
from keras.layers.core import Dense
import keras.optimizers

import pickle
file = open("NNData.txt", "rb")
model = pickle.loads(file.read())

scaler = sklearn.preprocessing.StandardScaler()
load_duration = 10
load_offset = 30

def load_scale(filename, d = load_duration, o = load_offset):
    x, sr = librosa.load(filename)
    mfcc = librosa.feature.mfcc(x, sr=sr, n_mfcc=n_mfcc).T
    return scaler.transform(mfcc)


filename_brahms = 'Music/R-DJ.wav'
x_brahms, sr_brahms = librosa.load(filename_brahms)

n_mfcc = 12
mfcc_brahms = librosa.feature.mfcc(x_brahms,
                    sr=sr_brahms, n_mfcc=n_mfcc).T
mfcc_1 = scaler.fit_transform(mfcc_brahms)

mfcc_brahms_test_scaled =
load_scale('Music/R-Weightlessness.wav')

features_test = numpy.vstack(mfcc_brahms_test_scaled)
labels_test = numpy.ones(len(features_test))
predicted_labels = model.predict(features_test)
score = 0
for i in range(len(predicted_labels)):
    score += predicted_labels[i][0]
score /= len(predicted_labels)
print(score)
print(predicted_labels[1:50])
```

Figure 15: Running RNN Code

```
Linear model: (SVM)
    f(x) = a*x + b
Coefficients (with 95% confidence bounds):
       a =      -353.8  (-369.8, -337.8)
       b =       361.2  (348.5, 374)

Goodness of fit:
  SSE: 110.1
  R-square: 0.9954
  Adjusted R-square: 0.995
  RMSE: 3.164
```

Figure 16: SVM Linear Regression Data

```
Linear model: (RNN)
    f(x) = a*x + b
Coefficients (with 95% confidence bounds):
       a =     0.00105  (0.001032, 0.001067)
       b =   -0.002287  (-0.003932, -0.0006415)

Goodness of fit:
  SSE: 1.692e-05
  R-square: 0.9994
  Adjusted R-square: 0.9993
  RMSE: 0.00124
```

Figure 17: RNN Linear Regression Data