MUSIC ANALYZER AND PLAGIARISM

Milind Bhattacharya[1], Sweekar Bandkar[2], Amit Badala[3]

[1]*Computer Engineering, Vidyalankar Institute of Technology (VIT), Mumbai University, Mumbai, India*
[2]*Computer Engineering, Vidyalankar Institute of Technology (VIT), Mumbai University, Mumbai, India*
[3]*Computer Engineering, Vidyalankar Institute of Technology (VIT), Mumbai University, Mumbai, India*

Abstract

Music artists put a lot of effort in creating a perfect musical composition. However, some other artists directly lift the original pieces or change them marginally and use it under their own title to gain recognition and success. To curb these mounting violations in copyrights, we created an algorithm to compare pieces of music with various songs in a database to recognize plagiarism.

The algorithm is based on the technique used by Shazam for music identification. The songs in the database are stored in the image form called acoustic fingerprints, which are derived from the spectrograms. A similar fingerprint is created for the song to be matched for plagiarism. Further, scatterplots are generated by comparing the fingerprint of the music under scrutiny to the fingerprints of all the songs in the database. Finally, all the scatterplots are subjected to a diagonal detection algorithm which returns the name of the song with highest percentage of match.

This paper initially presents few theories on sound like amplitude and frequency, which forms the base to the concepts of spectrogram and acoustic fingerprinting. It also explains various techniques of image processing that are used in the creation of scatterplots and detection of diagonals. The MATLAB software is used for the explanation and implementation of the entire algorithm.

Keywords— *Music plagiarism, Music recognition, Acoustic fingerprinting, Spectrogram, Scatterplot, Diagonal detection*

----------***----------

1. INTRODUCTION

The terminologies and processes explained below are used for the implementation of the algorithm. The diagrams used in this section (Fig. 1 to Fig. 4) have been generated using MATLAB for a 10 seconds clip of the song "Kalimba", composed by Mr. Scruff.

1.1 Amplitude

The amplitude of a periodic variable is a measure of its change over a single period. In other words, it is the measure of the amount of energy in a sound wave. Fig. 1 shows the representation of a sound wave with amplitude along vertical axis and time along horizontal axis.

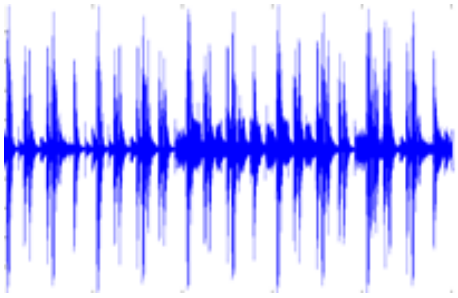

Fig 1 Representation of sound wave

1.2 Frequency

Frequency is the number of occurrences of a repeating event per unit time. It can also be defined as the number of cycles per unit.

1.3 Spectrogram

A graphic representation of a spectrum of frequencies in a sound wave is called a spectrogram. There are many formats of representing a spectrogram. For the implementation of this algorithm, we shall use a three dimensional graph. The horizontal axis represents time, the vertical axis is the frequency and the third dimension indicates the amplitude of a particular frequency at a particular time. This dimension is represented by the intensity or color of each point in the image. Fig. 2 illustrates a typical spectrogram created using this format.

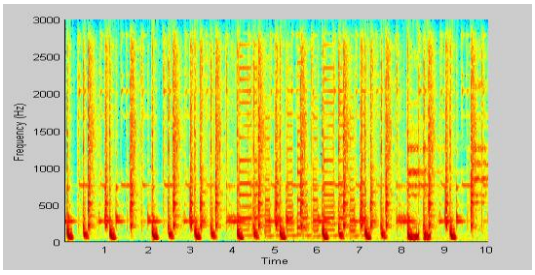

Fig 2 Spectrogram

1.4 Acoustic Fingerprint

An acoustic fingerprint or audio fingerprint is a condensed digital summary of an audio signal. It is generated from the spectrogram and is a unique representation for all audio signals. The acoustic fingerprint in Fig. 3 is generated using the spectrogram in Fig. 2.

Fig. 3 Acoustic fingerprint

1.5 Scatterplot

A scatter plot is a type of mathematical diagram using Cartesian coordinates to display values for two variables for a set of data. The two sets of data are the time values obtained from the acoustic fingerprints of the respective audio signals. The value of one variable determines the position on the horizontal axis and the value of the other variable determines the position on the vertical axis. Same data sets (data extracted from the same acoustic fingerprint) were used to create the scatter plot in Fig. 4 and hence the scatter plot consists of only diagonals.

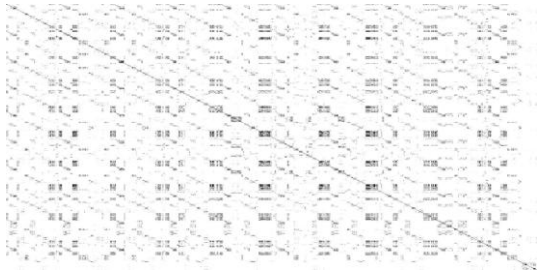

Fig. 4 Scatterplot

2. CREATION OF SONG DATABASE

The song database consists of acoustic fingerprints of various songs which are to be protected against plagiarism. This process includes reading the song files from the directory, creating the spectrogram of each song and finally generating the acoustic fingerprints from the spectrograms and storing it in the database. For the purpose of demonstration, we will consider 10 seconds clips of 10 songs. All the songs are stored in the WAV format. The list of songs used is:-

- Bin Tere – By Vishal-Shekhar
- I Need a Freak – By Lynn Tolliver
- Kalimba – By Mr. Scruff
- Maid with the Flaxen Hair – By Richard Stoltzman
- Mario Takes a Walk – By Jesse Cook
- My Lecon – By JLT
- Sarang Hae Yo – By Kim Hyun Sup
- Sleep Away – By Bob Acri
- Turn Up The Music – By Chris Brown
- Ya Ghali – By Guitara

We shall consider a legal case of music plagiarism to demonstrate the working. Composer of the song "I Need a Freak" (second song in the list), Lynn Tolliver successfully sued The Black Eyed Peas for sampling without his permission in the song "My Humps".

2.1 Reading the Song Files

The first step of this implementation involves reading the song files from a particular directory and obtaining the necessary parameters of the song using MATLAB. The following operations are performed in this step:-

- Store the sampled data in *samplesSong* and the sample rate (Herts) in *fsSong* used to encode the data in the file.

  [*samplesSong*, *fsSong*] =
  wavread ('*songPath**songName.*wav');

- Store the duration of the song in seconds in *timeSong*.

  timeSong = length (*samplesSong*)./*fsSong*;

- Covert the stereo audio samples stored in *samplesSong* to mono samples and store the values in *monoSong*.

  monoSong = *samplesSong* (:,1);

2.2 Creation of Spectrograms

As defined earlier, spectrogram is a graphic or photographic representation of a spectrum of frequencies in a sound wave. MATLAB offers an inbuilt method for creation of spectrogram:

spectrogram (*x*,*window*,*noverlap*,*nfft*,*fs*);

This returns the spectrogram of the input signal x. The other parameters required are explained below.

- *window* is a Hamming window of length *nfft*.
- *noverlap* is the number of samples that each segment overlaps.
- *nfft* is the FFT length and is the maximum of 256 or the next power of 2 greater than the length of each segment of *x*.
- *fs* is the sampling frequency.

We have used the following values for generating the spectrograms:-

- *x* = *monoSong* (Obtained in the earlier steps)
- *window* = 256

- *noverlap* = 250
- *nfft* = 256
- *fs* = *fsSong* (Obtained in the earlier steps)

We store the output of the spectrogram as an image file. Since the duration of each song varies but the number of rows in the spectrogram image remains constant, we need to resize the spectrogram. The number of rows and columns of the spectrogram are stored in r and c respectively.

[*r c*] = size (*spectrogramImage*);

We resize the spectrogram by increasing the number of columns. The new number of columns is calculated by multiplying the current number of columns and the duration of the song in seconds, i.e., timeSong.

scaledSpectrogram = imresize (*spectrogramImage*, [*r c***timeSong*]);

We use this output to overwrite the earlier spectrogram and store it as the final spectrogram image. Fig. 5 is the output obtained after the execution of the initial two steps.

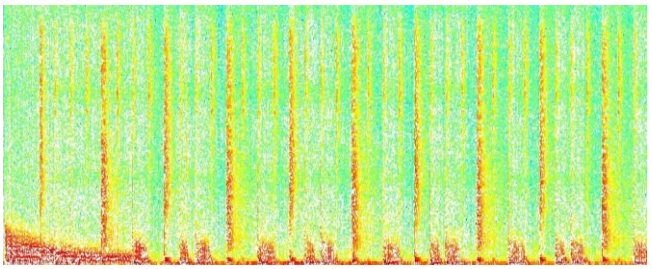

Fig. 5 Spectrogram of the song "I Need a Freak"

2.3 Creation of Acoustic Fingerprints

As defined earlier, an acoustic fingerprint is a condensed digital summary of an audio signal. Acoustic fingerprints are generated from the spectrograms. Initially, the colored spectrogram is converted to a grayscale image. Fig. 6 shows the grayscale form of the spectrogram in Fig. 5.

grayscaleSpectrogram = rgb2gray(*scaledSpectrogram*);

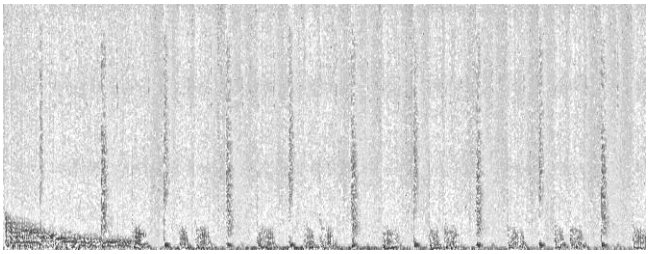

Fig. 6 Grayscale spectrogram

Certain peaks from the grayscale spectrograms are extracted. The algorithm involves determining the lowest pixel value in each column, i.e., peak, converting it into a black pixel (pixel value = 0) and plotting it on a white background (pixel value = 255) at the same position as that of the grayscale spectrogram. Fig. 7 illustrates the fingerprint obtained from the grayscale spectrogram by performing the above operations. This image is stored in the database and acts as a unique representation for each song.

Fig. 7 Fingerprint of the song "I Need a Freak"

3. FINDING A MATCH FOR PLAGIARISM

We will refer the song under scrutiny for plagiarism as a sample. A 5 second clip of the song "My Humps" is used as the sample for this experiment. We will require the fingerprint of the sample and then compare it with all the fingerprints of the songs in the database. The steps involved in creating the spectrogram and the fingerprint for the sample is the same as for the songs. Fig. 8 and Fig. 9 shows the spectrogram and fingerprint respectively, created for the sample.

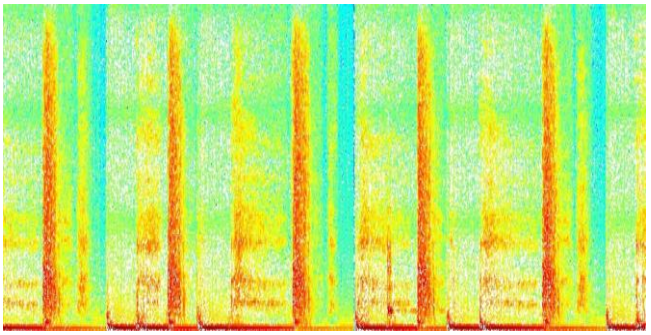

Fig. 8 Spectrogram of the sample "My Humps"

The comparison of the fingerprints is done using a hashing algorithm, which leads to the creation of scatterplots. Finally, diagonals of the scatterplots are detected and stored as an image with a reference in the name to the particular song. The song with the diagonal image having the highest number of black pixels is considered as the best match.

Fig. 9 Fingerprint of the sample "My Humps"

3.1 Creation of Scatterplots

As defined earlier, scatterplot is a type of mathematical diagram using Cartesian coordinates to display values for two variables for a set of data. The time values of the peaks present in the fingerprints of the song and the sample forms the data sets. The column number of a peak in the fingerprint corresponds to its time value in the song. Thus, the first variable of the data set is the column numbers of all the peaks present in the fingerprint of the song in the database. Similarly, the second variable of the data set is the column numbers of all the peaks present in the fingerprint of the sample.

We do not plot all the time values of the peaks to create the scatterplot. We will require another data set consisting of frequency values of the peaks of the song and the sample fingerprints to determine which values should be plotted. Since the vertical axis represents the frequency of the peak, the row number of the peak corresponds to its frequency value. The two variables in this data set are the row numbers of all the peaks present in the fingerprint of the song and the row numbers of all the peaks present in the fingerprint of the sample.

Let the number of peaks in the song fingerprint be x and the number of peaks in sample fingerprint be y.

Variables in first data set:-

1. *timeValueSong,* consisting of *x* data.
2. *timeValueSample,* consisting of *y* data.

Variables in the second data set:-

1. *freqValueSong,* consisting of *x* data.
2. *freqValueSample,* consisting of *y* data.

The time values of the peaks having the same frequency value in the song and sample fingerprints are plotted to form the scatterplot. Frequency value of each peak in sample fingerprint is compared with frequency values of all the peaks in the song fingerprint. If the frequency values are the same, the time value of the peak in sample fingerprint is plotted on the horizontal axis and the time value of the peak from song fingerprint is plotted on the vertical axis as black pixels on the scatterplot.

```
for m=1:y
    for n=1:x
        if (abs (freqValueSong (n) - freqValueSample (m))==0)
scatterPlot( timeValueSample (m), timeValueSong (n))=0;
       end
    end
end
```

Fig. 10 shows the scatterplot created after comparing the fingerprints of the song, "I Need a Freak" and the sample, "My Humps". Similarly, such scatterplots are created for all the songs in the database and stored as image files.

Fig. 10 Scatterplot

3.2 Diagonal Detection

A line detection algorithm is used to detect the diagonals along the angle -45 degrees. We define a mask of size s * s having its diagonal pixels along -45 degrees as 1 and all other pixels as 0. Convolution of this mask and the scatterplot is carried out and the result is stored as the diagonal image.

diagonal = conv2 (*scatterPlot*, *mask*, 'same');

We also define a threshold vale to determine the diagonal pixels in this image. Thus, all the pixels in the diagonal below the threshold value are set as a black pixel and the remaining pixels as white.

Fig. 11 demonstrates the image obtained after running the diagonal detection algorithm on the scatterplot in Fig. 10. In this implementation, we have defined the mask size as s=9 and the threshold value=1500.

Fig. 11 Image of the diagonals derived from the scatterplot

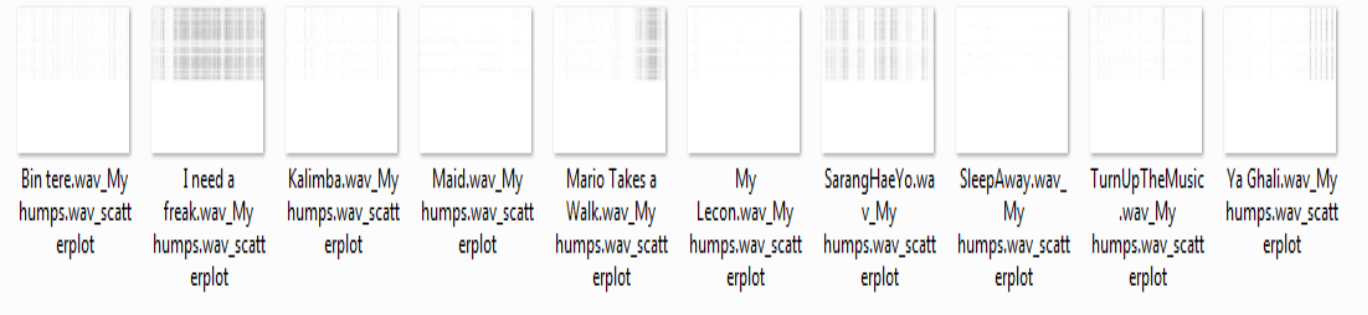

Fig. 12 Screenshot of all the scatterplots created in the directory

3.3 Finding the Best Match

So far we have created the diagonal images for all the scatterplots. We count the number of black pixels present in each diagonal image. The song which corresponds to the diagonal image having maximum number of black pixels is considered to be the best match for plagiarism.

4. CONCLUSIONS

It is practically impossible to identify plagiarism by listening to all the songs ever composed. This algorithm returns a small number of similar sounding compositions and thus makes it feasible to identify plagiarism by listening.

The algorithm creates image representations of the songs to find the similarity in music. One can notice that similar music compositions will have similar spectrograms. However, directly analyzing the spectrograms to recognize similarity would consume high amount of time and memory. The spectrograms were reduced to fingerprints, consequently reducing the processing time. Fig.12 is a screenshot of the directory consisting of the scatterplot images created for the song “My Humps” against all the songs mentioned in list. We can notice that the scatterplot image for the song “I Need a Freak” consists of maximum number of diagonals compared to the other songs. Hence, when we subject these scatterplot images to diagonal detection, it returns “I Need a Freak” as the best match for “My Humps”.

Such positive results were obtained for other pair of songs associated with plagiarism cases as well. Another application of this algorithm is music recognition. If a song exists in the database and the sample is a part of the same song, the algorithm returns the name of the song as the best match.

REFERENCES

[1] Avery Li-Chun Wang, “An Industrial-Strength Audio Search Algorithm”.
[2] ELE 201, Spring 2013, Laboratory No. 2, Part 1 and 2, Shazam.
[3] http://laplacian.wordpress.com/2009/01/10/how-shazam-works/.
[4] http://www.mathworks.in/.
[5] http://en.wikipedia.org/