Music Genre Classification

Hung-Ting Wen* and Yun-shao Sung†

Abstract. Our purpose of this project is to build a learning machine that can distinguish between different music genres. We will implement and experiement on several proposed methods to music genre classification and see if we can achive, or even outperform, results from researches. Our baseline would be 2nd-order coefficients combined with SVM classifier. After then, we will move on to Sparse Representation Classifier.

1. Introduction. Classifying music genre has never been an easy task, even for human beings. Modern music has its own family tree, and certain features of an ancester will be inherited by its decendents and that make classification sometimes impossible. However, for some music applications that constantly recommend musics to its users, a good classification is their basics to recommendation system.

2. Performance Criterion. For each run, we will randomly split 90% of the samples as training set and the remaining 10% as the test set. The performance criterion will based on 1) the length of texture window, 2) analysis window, and the classification accuracy of averaged result over 10 trails.

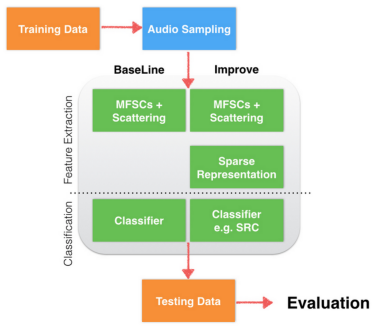

Figure1: Pipeline of our project

3. Background. To have the mel-frequency spectral coefficients (MFSC), we need to relate them to wavelet transform. The Fourier Transform of x(t) is:

$$X(w) = \int x(u)e^{(-iwu)}du \tag{3.1}$$

And if we frame the signal into short frame which xt,T(u) = x(u)wT(u-t), where wT is time frame window of size T:

$$X_{t,T}(w) = \int x_{t,T}(u)e^{(} - iwu)du \tag{3.2}$$

*htw230@nyu.edu
†yss265@nyu.edu

MFSCs are obained by averaging the spectrogram $|X_{t,T}(w)|^2$ over me-frequency interval, which these intervals have constant frequency for bandwidth below 1000Hz and above 1000Hz. Then MFSCs can be written as:

$$M_T x(t,j) = \frac{1}{2\pi} \int |X_{t,T}(w)|^2 |\widehat{\Psi}_j(w)|^2 dw \tag{3.3}$$

where $\widehat{\Psi}_j(w)$ covers a me-frequency interval that indexed j.

With Parsevals and Convolution theorem applied:

$$M_\lambda x(t,j) = \int |x_{t,T} \star \Psi_j(u)|^2 du \tag{3.4}$$

where Ψ_j can be interpreted as dilations of a mother wavelet $\Psi for \lambda > \lambda_0$, and this formula gives us the energy of x in a neighborhood of t of size T and in the mel-frequency interval indexed by j. Due to its properity that not able to capture the non-stationary structures of duration shorter than T, we may need choose the T to be small, and previous paper used 23ms.

$$\begin{bmatrix} \begin{bmatrix} [.....], \\ [.....], \\ [.....], \\ . \\ . \end{bmatrix}, \begin{bmatrix} [.....], \\ [.....], \\ [.....], \\ . \\ . \end{bmatrix}, ... \end{bmatrix} \tag{3.5}$$

Then we can use $\{\Psi_\lambda\}_{\lambda \in \Lambda}$ as our wavelet filter bank, and then the resulting wavelet transform defined by:

$$W_x(t) = \begin{pmatrix} x \star \phi(t) \\ x \star \Psi_\lambda(t) \end{pmatrix}_{\lambda \in \Lambda} \tag{3.6}$$

where ϕ is a low-pass filter covering the low-frequency interval not covered our wavelet filters.

4. Application. For the collection of mel-frequency bank, we will use the tool from Scat-Net to generate a series mel-frequency looks like Figure4.1. The frequency generator our reference paper used is the Galbor filer:

$$\Psi(t) = \theta(t) e^{i2\pi Qt} \Rightarrow \Psi(t) = \theta(t)\cos(2\pi Qt) + \theta(t)\sin(2\pi Qt) \tag{4.1}$$

where θ is Gaussian, with $Q = 16$ and $P = 23$. We think the idea for the series of frequency is that they need to have certain overlap. Therefore, when we are doing scaltering cascade for the filter $\lambda_i \dots \lambda_n$, the signal not being filtered, or emphasized by λ_i will be captured and emphasized by $\lambda_{i+1} \dots \lambda_n$, which are the neighbor filters. We may also insterested to check the effect of the frequency collections to the classfication accuracy, although larger frequency collection implies larger feature vector, which will increase complutation times. For the mel-frequency scattering part, we will use (3.6), which it will output the low-passed signal $x \star \phi_J$, and, at next layer, each $|x \star \Psi_{j_1}|$ is again retransformed by (3.6), which outout $|x \star \Psi_{j_1}| \star \phi_J$ and then computes $||x \star \Psi_{j_1}| \star \Psi_{j_2}| \star \phi_J$ in the next layer(Figure4.2).

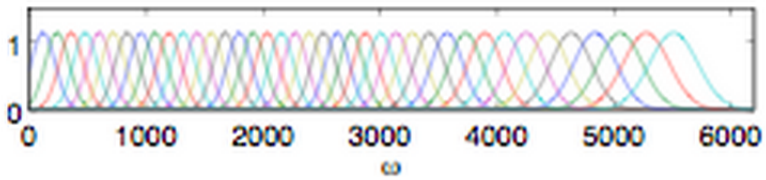

Figure 4.1. *Collection of mel-frequency*

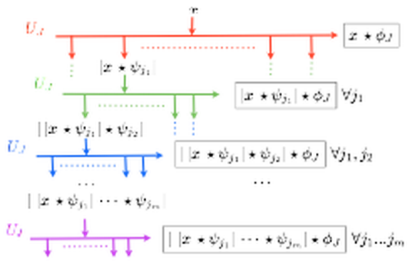

Figure 4.2. *Scattering cascade*

5. Baseline. Our baseline will be using 2nd-order Scattering Coefficients for feature extraction. Combining with SVM classifier, we are expecting to achieve roughly about 85% accuracy as indicated in the paper.

We will use Numpy for implementation. Numpy provides convolve, flatten, and other functions that can help us build our own implementation.

6. Problem formulation of further improvement. This is the part for the further improvement after baseline method. We will combine the scalled feature vector followed by transforming into sparse representation, which governed by the formula below:

$$(l^l) : argmin_{\omega \in \Re^p} \frac{1}{2}|x - D_\omega|_2^2 + \lambda|\omega|_1 \tag{6.1}$$

where D is a N by P dictionary matrix, and ω is a P by M matrix. The size of column which is P in dictionary is the length of atoms or codewords, and the purpose of dictionary is to sparsifies the representation such that can fit $|\omega|_0^0 < L$, where L is the number of non-zero element of each column of *omega*. In (6.1), we are both learning D and ω from data x, and the goal is to have the final ω sparse representation so each of x can have its own sparse representation, and then perform classification base on the representation.

(6.1) will yield the following interesting questions:

1. How we decide the size of atoms? In other word, how should we decide dictionarys number of colums. One approach will be just copy xT, which means number of atoms equals to number of data samples. However, we think this huge dictionary will be very costly during optimization. The alternative might be the number of top best principal component analsis (PCA). If we can get the top best P number of PCA axises, then it may means there are P components existed that can seperate x into clear differernt groups.
2. Whats the strategy of optimizating (6.1), after dictionary initialized in previous step? The

Algorithm 1 2nd-order Scattering Coefficients

for $x \in X$ **do**
  for $t = 1 \dots T$ **do**
    OUTPUT $x \star \phi(t)$

    for $\lambda_1 \in \Lambda$ **do**
      OUTPUT $|x \star \Psi_{\lambda_1}| \star \phi(t)$
      COMPUTE $x \star \Psi_{\lambda_1}$

      for $\lambda_2 \forall \lambda_2 \in \Lambda s.t. \lambda_2 > \lambda_1$ **do**
        OUTPUT $||x \star \Psi_{\lambda_1}| \star \Psi_{\lambda_2}| \star \phi(t)$
      end for
    end for
  end for
end for

first setp will be spase coding, which we can perform by Relaxation (or Baisis Pursuit) or Greedy approaches. For Relaxation approach, the newly defined function will be convex, and there are plenty of algorithm for sovling this like Interior Point Method and Iterative shrinkage. As for Greedy approach, Matching Pursuite can be the method to apply.

Then, the next step will be dictionary update. Based on K-SVD method, we can randomly select the atom in dictionary and find out all the signal that use the atom and we like to very optimize the current selected atom by substracting the contributions of every other atom. Repeat this atom updating for all atom in dictionay then we are done with dictionay and we can repeat sparse coding and so on until the error converge or hit the prefix number of iteration.

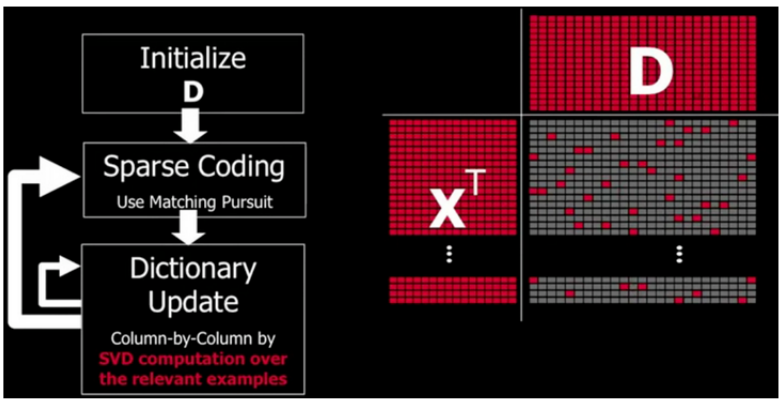

Figure 6.1. *K-SVD Algorithm*

7. Algorithm used in reference paper. The algorithm being used in reference paper are the combinations of scattering of mel-frequency cepstral coefficients (MFCCs), and sparse representation followed by sparse representation classifier (SRC). As for the baseline method, we will only use 2nd-ordered scattering method alone with classification (e.g. SVM) to see the performance and accuracy. The accuracy for this method that listed in the reference is about 82% under 1.5s texture window(Figure7.1), and our gaol for this baseline method will be trying to reproduce the accuracy. Further, we will also be interested to test the accuracy if we expand the scattering to 3rd-order.

For our further improvement, we will like to do the combination of scattering plus transformation to sparse transformation. The idea of the scattering method is that it reduce the in-class variability of signals while still preserving the most of the signal energy. According to the author, 1st-ordered scattering is good enough, and higher orders provide complementary information. Then, the idea of transforming feature vector to spase representation is a data-driven approach, andit will find high dimention and yet sparse encoding of each example with respect to the dictionary. Although some concerns may rise regarding to the enlarging dictionary for better classification performance may decrease the performance, the author claimed dictionary will learn from scattering coefficients to represent the residual in-class variability of the vectors of scattering coefficients. This implies transforming dense high-dimentional scattering representation into a sparse representation by this relative small dictionary could work well together, and is the main issue we set to dicover. Our expected accuracy for this approach will be 89.8%(Figure7.2).

Classifier	Features	Acc. (%)	TW (s)	AW (ms)
CSC	Many features [6]	92.7	3	93
SRC	Auditory cortical features [7]	92.4	30	∅
LDA	MMFCC + MOSC + MNASE [21]	90.6	30	6000, 23
SVM	Learned: DBN [10]	84.3	5	46
SVM	Learned: PSD on octaves [11]	83.4	5	46
AdaBoost	Many features [18]	83	13.9	46
SVM	2nd SC [12]	82	1.5	∅
SVM	Daubechies Wavelets [4]	78.5	30	3
Log. Reg.	Spectral Covariance [19]	77	3	46
SVM	1st SC [12]	72	1.5	∅
LDA	MFCC + other [20]	71	1	20

Figure 7.1. *Genre recognition accuracy of various algorithm on GTZAN dataset*

Layers	TW Features	AW Features	Acc. (%)	TW (s)	AW (ms)
2	2nd SC	MSF	91.2	3	23
1	2nd SC	∅	89.8	3	∅
2	1st SC	1st SC, MSF	89.7	3	23
3	1st SC	1st SC	89.3	3	372, 23
1	2nd SC	∅	88.9	1.5	∅
2	1st SC	1st SC	87.6	3	23
2	1st SC	MSF	86.3	3	23
1	1st SC	∅	82	3	∅
1	1st SC	∅	80.4	1.5	∅

Figure 7.2. *Genre recognition accuracy of proposed method on GTZAN dataset (using SRC)*

8. Datasets. To make comparable results, we will use the same data set as the paper used: GTZAN. This dataset consists of 1000 audio clips, each 30sec long. The dataset contains 10 genres, each of whom has 100 tracks. Each track is 22050Hz mono 16-bit file in .wav format.

REFERENCES

[1] Low pass filter by FFT convolution, *http://www.dsprelated.com/freebooks/sasp/Example_1_Low_Pass_Filtering.html*

[2] Multiscale Scattering for Audio Classifications, *http://www.cmap.polytechnique.fr/scattering/ismir-final.pdf*

[3] Digital image processing: p067- Dictionary Learning, *https://www.youtube.com/watch?v=XLXSVLKZE7U*

[4] Interior-point methods, *https://web.stanford.edu/class/ee364a/lectures/barrier.pdf*

[5] Iterative Shrinkage/Thresholding Algorithms, *http://people.ee.duke.edu/ lcarin/figueiredo.pdf*

[6] Matching pursuits with time-frequency dictionaries, *http://www.cmap.polytechnique.fr/ mallat/papiers/MallatPursuit93.pdf*

[7] Efficient Implementation of the K-SVD Algorithm using Batch Orthogonal Matching Pursuit, *http://www.cs.technion.ac.il/ ronrubin/Publications/KSVD-OMP-v2.pdf*