SCATTERING TRANSFORM FOR INSTRUMENT CLASSIFICATION

Chris Miller, Colin Fahy
Music Information Retrieval, Spring 2015, New York University

ABSTRACT

Mel-Frequency Cepstral Coefficients (MFCCs) are widely used for audio classification tasks, but lose high-frequency information with longer window sizes. The scattering transform has been shown to recover these high-frequency components and allows for feature extraction of longer temporal structures. This paper reviews the benefits of the scattering transform as it applies to instrument classification, and documents a handful of classification implementations as written by the authors and others. The comparative successes and failings of these implementations are discussed.

1. INTRODUCTION

Classification seeks to apply a musically informative label to an audio signal. In the field of Music Information Retrieval, classification of signals by instrument, genre, key, or other quality are common and well-documented problems. The classification process is typically two-fold, beginning with the extraction of a feature vector from a signal, and the classification of that vector using clustering or other machine learning techniques. This feature vector is typically constructed to be relevant to the classification task at hand, containing information about qualities such as chroma for key classification, tempo for genre, or timbre for instruments.

Mel-frequency cepstral coefficients (MFCCs), commonly used for speech recognition [10], have been shown to be useful for a handful of musical classification tasks [7], but lose high frequency information at long window times. The scattering transform recovers this information through a cascade of wavelet transforms and modulus operators. It has been shown in [3], [1] to provide improved classification accuracy over MFCCs.

In this paper, we will examine the scattering transform as it applies to instrument classification. Section 2 will briefly review MFCCs and their shortcomings. Section 3 will review the wavelet transform and then expand to define a scattering transform. Implementation of computation of scattering coefficients both by the authors and by others will be reviewed in Section 4, while classification techniques will be discussed in Section 5. Finally, results for an instrument classification trial using content from MedleyDB [4] will be documented and reviewed.

2. MEL-FREQUENCY CEPSTRAL COEFFICIENTS

In the source-filter model of synthesis, any sound $y(t)$ is taken to be the convolution of an excitation signal $e(t)$ of an LTI system with the impulse response $h(t)$ of that system: $y(t) = e(t) * h(t)$ [9]. Though the definition of timbre is hardly concrete, the particular timbre of an instrument is, in part, contained in its distinctive spectral shaping and resonances described by its impulse response $h(t)$. In the search for a characteristic feature vector representing timbre, one would like to separate this impulse response from pitch information and represent it compactly.

Let $\hat{X}(\omega)$ be the Fourier transform of a time-domain musical signal $x(t)$:

$$\hat{X}(\omega) = \int x(t)e^{-it\omega}dt \tag{1}$$

The Short-Time Fourier Transform of $x(t)$, also called the spectrogram, partitions the signal into overlapping frames of N samples through use of a windowing function $\phi(t)$ of unity sum ($\int \phi(t)dt = 1$) and takes the Fourier transform of each frame:

$$\hat{X}(n, \omega) = \int x(t)\phi(t-n)e^{-it\omega}dt \tag{2}$$

The spectrogram $\hat{X}(n, \omega)$, however, is unstable to time warping. If our signal $x(t)$ undergoes time warping such that $x'(t) = x(t - \epsilon t)$, the Euclidian distance between spectrogram representations $\||\hat{X}(n, \omega)| - |\hat{X}'(n, \omega)|\|$ is large even when ϵ is small, especially in the high frequencies [2]. This is problematic, as a subtly time-warped version of a signal is not perceived as particularly different to the original signal by a listener.

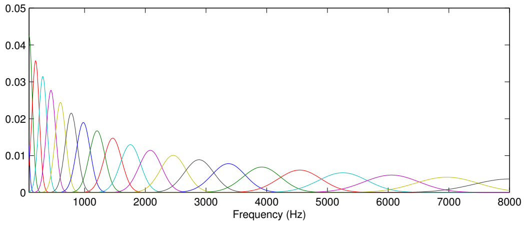


Figure 1. Mel-scale filterbank $\hat{\psi}_\lambda(\omega)$.

Mel-Frequency Spectral Coefficients (MFSCs) average the information in the spectrogram over frequency through use of a log-scale bandpass filter bank. The mel scale is a measurement of frequency that is linear below 1 kHz and logarithmic above, and is defined in relation to Hz as $f_{mel} = 1127.01048 \log(1 + \frac{f_{Hz}}{700})$ [9]. A mel-scale filter bank $\{\hat{\psi}_\lambda(\omega)\}_{\lambda \in \Lambda}$ is built with the center frequency λ of each filter spaced linearly in the mel-scale. Implementations of mel filter banks differ, but a common method requires a user to define a number of filters M and a frequency range $[\lambda_1, \lambda_M]$. The filter bank is therefore constructed to have good frequency support over this range – an example is shown in Figure 1, which contains $M = 20$ filters from $\lambda_1 = 86$ Hz to $\lambda_M = 8000$ Hz.

An MFSC representation is given by

$$Mx(n, \lambda) = \int |\hat{X}(n, \omega)|^2 |\hat{\psi}_\lambda(\omega)|^2 d\omega \tag{3}$$

Mel-Frequency Cepstral Coefficients (MFCCs) are computed from the MFSCs through conversion to decibels and a cosine transform, which decorrelates the mel-dB spectrum [10]. Characteristics of the impulse response $h(t)$ for an instrument are obtained from the low-quefrency cepstral coefficients.

It has been shown in [3], however, that MFSCs lose information in the high frequencies. As a consequence, window sizes N are typically kept small, on the order of 20 milliseconds, to minimize this loss of information. Such short windows, however, put a limit on the ability of Mel-Frequency Spectral/Cepstral Coefficients to capture longer spectral structures, which thus puts significant limits on their use in instrument classification.

3. SCATTERING REPRESENTATION

A scattering representation of order ν retrieves high-frequency information lost due to mel-scale averaging, and thus allows for expansion of analysis window sizes N. Computation of scattering coefficients through a cascade of wavelet transforms and modulus operations is reviewed in this section.

3.1 Wavelet Transform

A wavelet transform is constructed through the dilation and translation of a mother wavelet $\psi(t) \in \mathbf{L}^2(\mathbb{R}^d)$ [8] [5]. This dilation defines a basis of wavelets $\psi_\lambda(t)$ which, in the frequency domain, is equivalent to a filter bank $\hat{\psi}_\lambda(\omega)$. Time-domain wavelets $\psi(t)$ corresponding to filters 2, 5, and 11 in the mel-scale filter bank shown in Figure 1 can be seen in Figure 2. The wavelets used throughout this paper are Morlet wavelets: complex exponentials modulated by a Gaussian envelope.

By Parseval's theorem, the energy carried by the MFSC representation given in Equation 3 can be written in the time domain as a convolution of a windowed signal with a wavelet in our wavelet basis:

$$Mx(n, \lambda) = \int |(x(t)\phi(t-n)) * \psi_\lambda(t)|^2 dt \tag{4}$$

The wavelet transform of the signal $x(t)$ is given as the convolution of the signal with the windowing function $\phi(t)$, resulting in low-pass filtering, as well as convolution with all wavelets defined by the mel-scale filter bank:

$$Wx(t) = \Big(x(t) * \phi(t), \quad x(t) * \psi_\lambda(t)\Big)_{t\in\mathbb{R}, \lambda\in\Lambda} \tag{5}$$

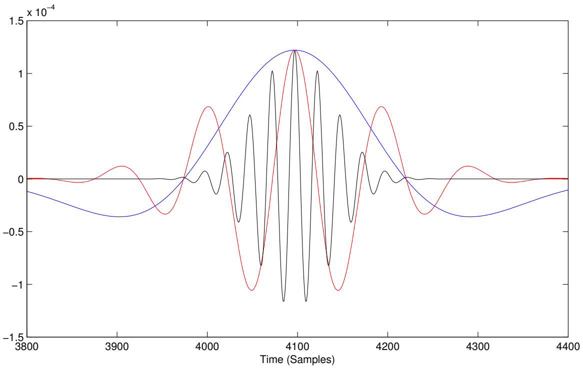


Figure 2. Example wavelets $\psi_{\lambda_2,\lambda_5,\lambda_{11}}(t)$ for the filter bank shown in Figure 1.

A wavelet modulus operator $|W|x(t)$ removes the complex phase of all wavelet coefficients $|x * \psi_\lambda|$, but keeps low-frequency phase information contained in $x * \phi$ [3].

3.2 Scattering Transform

As shown above, MFSCs are approximately equal to the averaged wavelet coefficients $|x(t) * \psi_\lambda(t)| * \phi(t)$ with the square removed to avoid emphasizing larger coefficients. Averaging by convolution with ϕ removes high-frequency content; a successive wavelet modulus operation $|x * \psi_{\lambda^1}| * \psi_{\lambda^2}$ recovers it. This motivates a scattering representation, given by a cascade of wavelet transforms and modulus operations.

A zeroth-order scattering representation $S_0x(t) = x * \phi(t)$ removes all high frequency information by averaging with the temporal window ϕ. This information is recovered by a successive wavelet modulus transform

$$|W|x(t) = \Big(x * \phi(t), \quad |x * \psi_\lambda(t)|\Big)_{t\in\mathbb{R}, \lambda\in\Lambda} \tag{6}$$

The first-order scattering coefficients $S_1x(t)$ mimic MFSCs by averaging with ϕ:

$$S_1x(t, \lambda^1) = |x * \psi_{\lambda^1}(t)| * \phi(t) \tag{7}$$

The scattering coefficients for a path of frequencies $\mathbf{p} = (\lambda^1, \lambda^2, \ldots, \lambda^\nu)$ is therefore given by

$$S_\nu x(t, \mathbf{p}) = \|\ldots\|x * \psi_{\lambda^1}| * \psi_{\lambda^2}| * \ldots | * \psi_{\lambda^\nu}| * \phi(t) \tag{8}$$

where the order ν of the scattering coefficient is given by the length of $\mathbf{p}$. As shown in [3] [8], since the wavelet modulus operator $|W|$ is contractive, it follows that the scattering operator S is also contractive, meaning that S is stable to time-warping and additive noise. Additionally, given appropriate wavelets, the wavelet transform is unitary, which preserves the signal norm. As a result, it can be shown that the energy in $x(t)$ is completely contained in the scattering coefficients $S_\nu x(t)$ as ν goes to infinity:

$$\|S_\nu x\| \to \|x\|, \quad \nu \to \infty \tag{9}$$

The scattering transform is also invertible, so that the original signal x is recoverable from its scattering representation for ν large enough. At window times less than 6 seconds, 98% of the signal's energy is carried by first- and second-order scattering coefficients [1].

4. FEATURE EXTRACTION

Mel-Frequency Cepstral Coefficients are often used to build feature vectors for classification tasks. After converting MFSCs to MFCCs by taking the log and decorrelating with a Discrete Cosine Transform (DCT), MFCCs are normalized across features and further averaged over windows of approximately one second long. A similar concept is used to implement a feature extraction algorithm using scattering representations.

4.1 First-Order Cosine Log Scattering

For our implementation, a mel-scale filter bank is first constructed for the user-defined parameters λ_1, λ_M (both in Hz), and number of filters M. To ensure good frequency support over this range, the center frequencies
$\lambda_1, \lambda_2, \ldots, \lambda_M \in \Lambda$ are deployed linearly in the mel-scale. A filter $\hat{\psi}_{\lambda_j}(\omega)$ is a Gaussian centered at λ_j with a standard deviation σ defined by half of the distance to the next filter:

$$2\sigma_j = (\lambda_{j+1} - \lambda_j) \tag{10}$$

First-order scattering coefficients $S_1x(n, \lambda)$ are computed by fast-convolution of the windowed signal $x(t)\phi(t-n)$ and the wavelets $\psi_\lambda(t)$. As with MFCCs, these coefficients are converted to cepstrum and undergo a cosine transform for a first-order cosine log-scattering (CLS) representation. As the DCT decorrelates the coefficients into an efficient representation, only the first N_{DCT} coefficients are kept (the very first CLS coefficient is thrown away). They are subsequently averaged into feature bins such that each feature vector represents approximately one second of audio.

4.2 Second-Order Cosine Log Scattering

A similar approach is used to construct feature vectors using second-order scattering coefficients. These coefficients are contained in the three-dimensional tensor $S_2x(t, \lambda^1, \lambda^2)$. Higher order scattering representations are significantly more computationally expensive than lower order ones, and [1] shows that not all second-order coefficients need be calculated.

Second-order cosine log scattering coefficients are computed by taking the DCT first along λ^2 and then along λ^1. Implementation of this is, however, is non-trivial, and our current implementation results in rather poor classification for second-order CLS coefficients. As such, they are computed using the ScatNet MATLAB toolbox from

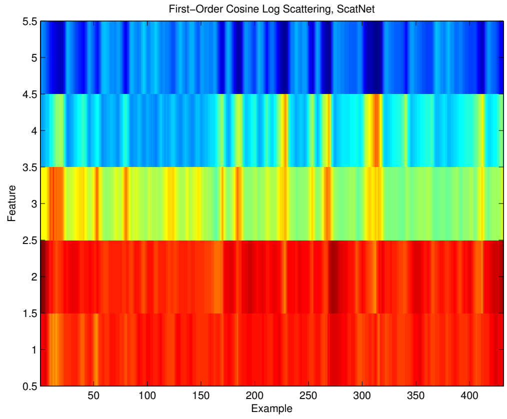


Figure 3. CLS coefficients, ν=1.

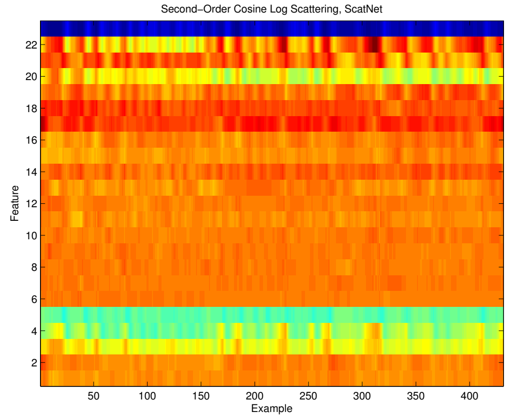


Figure 4. CLS coefficients, ν=2.

the Data Processing and Classification team at École Normale Supérieur, which can be found at the following url: `http://www.di.ens.fr/data/software/`.

Figures 3 and 4 show the first and second order CLS coefficients computed with the ScatNet toolbox. J is set to 8, which determines the maximal scale of the wavelets as well as the window size of ϕ.

5. CLASSIFICATION

Armed with feature vectors that describe timbre, we move on to the classification stage. A training set is compiled from MedleyDB [4] for seven different instruments, with a handful of signals from each class left over for testing purposes. MedleyDB is a dataset of multitrack recordings: as such, each signal contains long passages of silence that will both needlessly increase computation time and confuse the classifiers. A python script extract_instrument_stems.py is included in our wavelet classification toolbox to trim the silence from the files used to build our training (and testing) sets.

5.1 k-Nearest Neighbor

Two classification techniques are used and compared. The first is a simple nearest neighbor classification, which computes the inner product of a given testing feature vector with all feature vectors in the training set. The set of training vectors $f_\tau \in \{f_1, f_2, \ldots, f_T\}$ is constructed by placing each vector in a matrix $\mathbf{F}^\tau$ and normalizing across features. A testing set $\mathbf{F}^\xi$ is constructed in the same way.

A vector of nearest neighbor values $v_\xi(\tau)$ for a given testing feature vector f_ξ is computed by taking its dot product with all members of the testing set:

$$v_\xi(\tau) = \langle f_\xi, f_\tau \rangle \ \ \forall\ \tau \tag{11}$$

The maximum value in the nearest neighbor vector $\sup_\tau\{v(\tau)\}$ is located at the index $\tau = \tau'$ where the training vector $f_{\tau'}$ is most closely aligned with the testing vector f_ξ. The class of $f_{\tau'}$ (such as piano, trombone, cello, etc.) is then assigned to f_ξ. For our testing purposes, the ground truth label of f_ξ is also retrieved so that we may judge our classifier's accuracy.

k-nearest neighbor builds on this by taking the k largest values in $v(\tau)$ and choosing the most common label from those k classifications. Our implementation sets k to be the square root of the total number of samples in the training set $k = \sqrt{T}$.

5.2 Support Vector Machines

SVM (Support Vector Machines) is another supervised learning technique, but for binary classification. SVM constructs a model that separates the data points in feature space belonging to each class with as wide a gap as possible. A handful of techniques exist for multiclass classification using SVM as outlined in [6]. Our classifier uses the "one-for-one" technique.

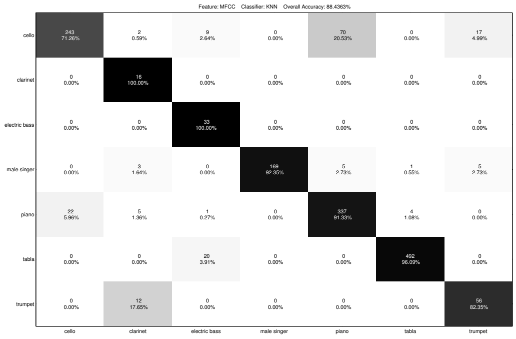
Given a number of classes C (in our case seven), a two class SVM is trained on each pair of possible classes: $C\frac{(C-1)}{2}$. For each sample, each classifier will predict either one or the other class. The winner is given 1 vote. The class with the most votes is the winner for that sample – the so-called "Max Wins" strategy. One-for-One gives comparable accuracy to the other techniques from [6] but with much quicker computation time. We use the LIBSVM package for SVM classification, found here: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`.

Feature	k-NN	SVM
MFCC	88.44%	90.74%
CLS, ν = 1	84.43%	89.95%
CLS, ν = 1 (ScatNet)	59.49%	61.40%
CLS, ν = 2 (ScatNet)	61.13%	62.71%

Table 1. Classification accuracies for different feature extraction methods and classifiers.

6. RESULTS

Our classification results are shown in Table 1. MFCCs perform well (88.44% overall accuracy) for k-Nearest Neighbor classification and slightly better (90.74%) for SVM. Our own implementation of first-order cosine log scattering coefficients are not far behind – reaching 89.95% classification accuracy for SVM. This is not surprising since, as discussed throughout this paper, MFCCs and first-order CLS coefficients are approximately the same thing. The differences here come from the usage of different bandpass filter shapes – MFCC computation uses skewed triangular filters, while our implementation of first-order scattering uses Gaussian bandpass filters (i.e. Morlet wavelets).



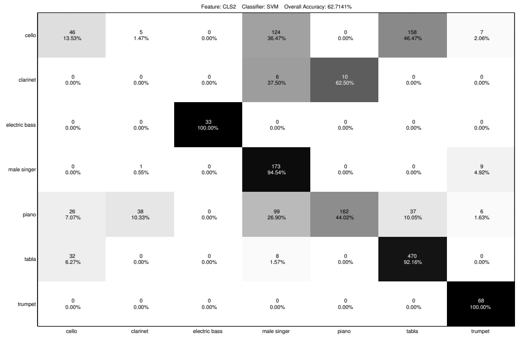
Feature: MFCC Classifier: KNN Overall Accuracy: 88.4363%

	cello	clarinet	electric bass	male singer	piano	tabla	trumpet
cello	243 71.26%	2 0.59%	9 2.64%	0 0.00%	70 20.53%	0 0.00%	17 4.99%
clarinet	0 0.00%	16 100.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%
electric bass	0 0.00%	0 0.00%	33 100.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%
male singer	0 0.00%	3 1.64%	0 0.00%	169 92.35%	5 2.73%	1 0.55%	5 2.73%
piano	22 5.96%	5 1.36%	1 0.27%	0 0.00%	337 91.33%	4 1.08%	0 0.00%
tabla	0 0.00%	0 0.00%	20 3.91%	0 0.00%	0 0.00%	492 96.09%	0 0.00%
trumpet	0 0.00%	12 17.65%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	56 82.35%

Figure 5. k-Nearest Neighbor classification using MFCCs. Overall accuracy = 88.44%.

Figure 5 shows a confusion matrix for k-Nearest Neighbor classification using MFCCs. Clarinet and Electric Bass score the highest with 100% classification accuracy, while cello does the poorest with 71.26% accuracy.

Figure 6 shows a confusion matrix for SVM classification using second order cosine log scattering coefficients. Overall accuracy is very low compared to MFCCs and our own implementations, but that is because these scattering coefficients are computed with $J = 8$. J is the maximal scaling for scattering – as J increases we get more features per example, and the window size of ϕ is increased. The



Feature: CLS2 Classifier: SVM Overall Accuracy: 62.7141%

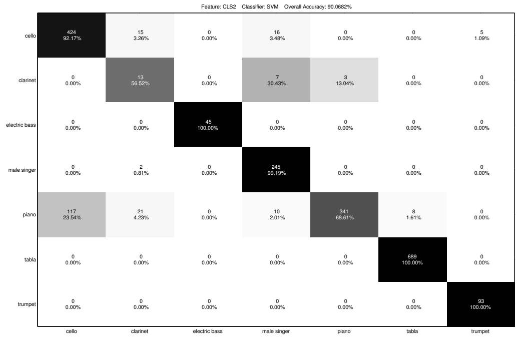
	cello	clarinet	electric bass	male singer	piano	tabla	trumpet
cello	46 13.53%	5 1.47%	0 0.00%	124 36.47%	0 0.00%	158 46.47%	7 2.06%
clarinet	0 0.00%	0 0.00%	0 0.00%	6 37.50%	10 62.50%	0 0.00%	0 0.00%
electric bass	0 0.00%	0 0.00%	33 100.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%
male singer	0 0.00%	1 0.55%	0 0.00%	173 94.54%	0 0.00%	0 0.00%	9 4.92%
piano	26 7.07%	38 10.33%	0 0.00%	99 26.90%	162 44.02%	37 10.05%	6 1.63%
tabla	32 6.27%	0 0.00%	0 0.00%	8 1.57%	0 0.00%	470 92.16%	0 0.00%
trumpet	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	68 100.00%

Figure 6. SVM classification using second-order CLS, $J = 8$. Overall accuracy = 62.71%.

J	Accuracy
4	56.32%
8	62.71%
12	86.15%
16	90.07%

Table 2. Classification accuracy of second order CSL scattering for different values of J.

length of the window $N = a^J$, where $a = 2^{1/Q}$ and Q is the number of filters per octave.



Feature: CLS2 Classifier: SVM Overall Accuracy: 90.0682%

	cello	clarinet	electric bass	male singer	piano	tabla	trumpet
cello	424 92.17%	15 3.26%	0 0.00%	16 3.48%	0 0.00%	0 0.00%	5 1.09%
clarinet	0 0.00%	13 56.52%	0 0.00%	7 30.43%	3 13.04%	0 0.00%	0 0.00%
electric bass	0 0.00%	0 0.00%	45 100.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%
male singer	0 0.00%	2 0.81%	0 0.00%	245 99.19%	0 0.00%	0 0.00%	0 0.00%
piano	117 23.54%	21 4.23%	0 0.00%	10 2.01%	341 68.61%	8 1.61%	0 0.00%
tabla	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	689 100.00%	0 0.00%
trumpet	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	0 0.00%	93 100.00%

Figure 7. SVM classification using second-order CLS, $J = 16$. Overall accuracy = 90.07%.

By increasing J we widen the temporal window ϕ while recovering high-frequency information lost in lower order scattering coefficients due to time averaging. Figure 7 shows the overall accuracy dramatically improves by raising J to 16, jumping up to 90.07%. Table 2 shows the classification accuracies for second order CLS scattering using SVM for different values of J.

7. CONCLUSION

MFCCs have been shown to perform well for instrument classification, but lose high-frequency information when the windowing function ϕ is long. Typically MFCCs are calculated using windows of approximately 20 milliseconds for this reason, but this prohibits the extraction of long-term timbral structures important to the task of instrument classification.

Scattering representations have been shown to recover this information, and theoretically allow for the use of much longer temporal windows in feature extraction. Our results show that for low J values, which generate short time windows and small CLS feature vectors, instrument classification is mediocre at best. Increasing J, however, leads to much stronger classification accuracy.

Unfortunately, scattering coefficients take a long time to compute. Our initial training set was extensive, but was cut down to consist of only five minutes of audio data per instrument in order to save time. With more training data classification would be better, but at a significant cost to computation time. Additionally, raising J raises the computation time as well. The goal of strong classification with both an extensive training set and high J value is a time consuming one indeed.

Future work will be centered around both fixing our own implementation of second-order cosine log scattering coefficients, as well as maximizing classification accuracy while minimizing computation time.

8. REFERENCES

[1] Joakim Andén and Stéphane Mallat. Multiscale scattering for audio classification. In *ISMIR*, pages 657–662, 2011.

[2] Joakim Andén and Stéphane Mallat. Scattering representation of modulated sounds. In *Proc. of the 15th Int. Conference on Digital Audio Effects (DAFx-12)*, 2012.

[3] Joakim Andén and Stéphane Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62:4114–4128, 2014.

[4] Rachel Bittner et al. Medleydb: A multitrack dataset for annotation-intensive mir research. In *ISMIR*, 2014.

[5] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *Information Theory, IEEE Transactions on*, 36(5):961–1005, Sep 1990.

[6] Chih-Wei Hsu and Chih-Jen Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, Mar 2002.

[7] Beth Logan et al. Mel frequency cepstral coefficients for music modeling. In *ISMIR*, 2000.

[8] Stéphane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398, 2012.

[9] Tae Hong Park. *Introduction to Digital Signal Processing, Computer Musically Speaking*. World Scientific Publishing Co., Singapore, 2010.

[10] Lawrence R Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.