WAVELET SCATTERING FOR AUTOMATIC CHORD ESTIMATION

Chris Miller
MARL, New York University
crm313@nyu.edu

Second Author
Retain these fake authors in submission to preserve the formatting

Third Author
Affiliation3
author3@ismir.edu

ABSTRACT

The abstract should be placed at the top left column and should contain about 150-200 words.

1. INTRODUCTION

Much work has been done in the domain of Automatic Chord Estimation (ACE) – one of the so-called "holy grails" of Music Information Retrieval (MIR) research. In general, ACE systems seek to analyze a particular audio file and generate a vector indicating the predominant chord in the song over a particular time frame ("C minor from 1.5 seconds to 4 seconds", for example).

Chord estimation systems developed at NYU's Music and Audio Research Lab (such as those detailed in [2] [4]) use supervised learning approaches to train a chord model on a large chord vocabulary of 157 different chords. Typically, these systems consist of three-part pipeline:

Feature Extraction to transform the audio file into a musically relevant representation

Pattern Matching to match the harmonic content in a particular time frame with a template in a trained chord model

Decoding to determine whether or not a predicted chord makes sense in context

Feature extraction in the ACE systems of [2] [1] uses a Constant-Q transform to create a frequency-space representation on a perceptual log-scale of ten octaves. That is, a pitch representation $\rho_{t,\lambda}$ encodes the energy in temporal window t at a frequency band centered around λ, where each λ corresponds to a pitch in traditional Western twelve-tone music theory. A k-multi-band chroma representation is then created by first windowing the frequency scale with k Gaussians distributed evenly over the support of λ, and then by wrapping each windowed representation $\hat{\rho}_{t,\lambda}$ into a chroma vector. This creates a new representation $\chi^k_{t,q}$ which is invariant to octave transpositions where q is an integer $\in [1, 12]$ representing a chroma (or pitch) value, and k is the windowed band.

Pattern matching is done by creating a chord model for all 157 chords with Gaussian Mixture Models (GMMs). This is a supervised learning approach where features are extracted for a large training set that contains examples of all 157 chords (with annotations) and are then used to train GMM templates for each chord. Data augmentation is employed by rotating each chord through the chroma space so that one example of C minor can also train a model for C# minor, D minor, etc. For any temporal frame t, each of the k chroma representations $\chi_{t,q}$ are matched using GMMs and then fused together via geometric mean. Finally, **decoding** uses Hidden Markov Model (HMM) systems to post-filter the pattern matched data. Here, we use the Viterbi algorithm, which describes the transition probability from one chord C_i state to another – $\mathbb{P}(C_i \rightarrow C_j)$, therefore filtering out chord transitions that are highly unlikely. The matrix of transition probabilities for Viterbi decoding is generated from the labeled training data.

Our experiments looked to replace parts of the existing feature extraction techniques with new ones based on Haar wavelet transforms and deep Haar scattering. We began with the MATLAB system designed by Taemin Cho available in [3] and added the new wavelet-based feature extraction routines to it.

2. EXPERIMENTAL SETUP AND EVALUATION

Evaluation of an ACE system was determined using some of the tools found in the mir_eval library [5] and written into a simple python script called basic_evaluate.py, which compared the estimated chords coming out of the ACE system to the ground truth annotations for all songs in the testing set. All experiments were run on the High Performance Computing (HPC) environment at NYU's Courant Institute.

For each experiment, the script **extractFeaturesAndTrain** is called first to both train a chord model and generate a Viterbi transition probability matrix. extractFeaturesAndTrain takes a list of training files (remaining constant throughout all experiments), as well as a directory to write the features, chord models, and transition matrices to. We adjusted this script to also take in a 'mode' variable and a 'band'. 'mode' is a switch changing the feature extraction method between multiband chroma, Haar wavelet transforms, and deep Haar scattering. 'band' (k) determines the amount of chroma vectors at any given temporal window – equivalent to the number of bands in the multiband chroma representation, and the maximum wavelet scale in the wavelet and scattering representations (i.e. the

number of wavelet coefficients).

Next, **doChordID** is called to perform the chord recognition task on the testing set. doChordID takes a list of test files (again, remaining constant throughout all experiments) as well as the directory containing the chord models and decoding matrices to use (i.e. the output of extractFeaturesAndTrain, and another directory to output the computed results. doChordID was also modified to take in a 'mode' switch – the script automatically picks up the band k from the GMM models in the trained chord models.

Finally, **basic_evaluate.py** is run over the test set to evaluate the results. As per [5], there is "no single right way to compare two sequences of chord labels," and the mir_eval package computes ACE accuracy based along metrics such as root, triads, maj/min, sevenths, inversions, etc. For simplicity, we evaluate all experiments using the mirex measure, which "considers a chord correct if it shares at least three pitch classes in common" [5].

State of the art systems such as those in [2] use a multiband chroma feature extraction with $k = 4$ windowed bands. To begin, some baseline trials were run on a training set consisting of 111 songs from the Beatles discography, 99 RWC pop songs, 224 songs from the Billboard dataset, and 20 Queen songs, for a total of 454 songs. Our testing dataset comprised of 53 Beatles songs that were not part of the training set. Once again, these training and testing sets were kept constant across all experiments. For the $k = 4$ multiband system, the MIREX evaluation is **66.7%**.

3. HAAR WAVELET TRANSFORM

Dating back to 1909, the Haar wavelet ψ is a piecewise constant, real function of compact support, consisting of two steps of equal length and opposite values. Within a discrete framework, it is defined by the following formula:

$$\forall u \in \mathbb{Z},\ \boldsymbol{\psi}[u] = \begin{cases} \frac{-1}{\sqrt{2}} & \text{if } u = 0 \\ \frac{1}{\sqrt{2}} & \text{if } u = 1 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

The "mother" wavelet $\boldsymbol{\psi}[u]$ is translated and dilated by powers of two, so as to produce a family of wavelets $\boldsymbol{\psi_{j,b}}[u] = 2^{j/2}\boldsymbol{\psi}[2^j(u-2b)]$ indexed by the scale parameter $j \in \mathbb{N}$ and the translation parameter $b \in \mathbb{Z}$. After endowing them with the Euclidean inner product

$$\langle \boldsymbol{\psi_{j,b}} | \boldsymbol{\psi_{j',b'}} \rangle = \sum_{u=-\infty}^{+\infty} \boldsymbol{\psi_{j,b}}[u]\boldsymbol{\psi_{j',b'}}[u], \tag{2}$$

the wavelets $\{\boldsymbol{\psi_{j,b}}\}_{j,b}$ form an orthonormal basis of finite-energy real sequences. Moreover, the Haar wavelet is the shortest function of compact support such that the family $\{\boldsymbol{\psi_{j,b}}\}_{j,b}$ satisfies this orthonormality property. On the flip side, it has a poor localization in the Fourier domain, owing to its sharp discontinuities. Elements of the Haar wavelet basis are shown on Figure 2 for various values of j and b.

The wavelet transform coefficients of some finite-energy sequence $\boldsymbol{x} \in \ell^2(\mathbb{Z})$ are defined by $\boldsymbol{W}\boldsymbol{x}[j,b] = \langle \boldsymbol{x} | \boldsymbol{\psi_{j,b}} \rangle$. Since $\boldsymbol{x}[u]$ has a finite length $K = 2^J$, the above decomposition is informative only for indices (j,b) such that $j < J$ and $2^j b < K$, that is $b < 2^{J-j}$. The number of coefficients in the Haar wavelet transform of $\boldsymbol{x}[u]$ is thus equal to $\sum_{j<J} 2^{J-j} = 2^J - 1$. For the wavelet representation to preserve energy and allow signal reconstruction, a residual term

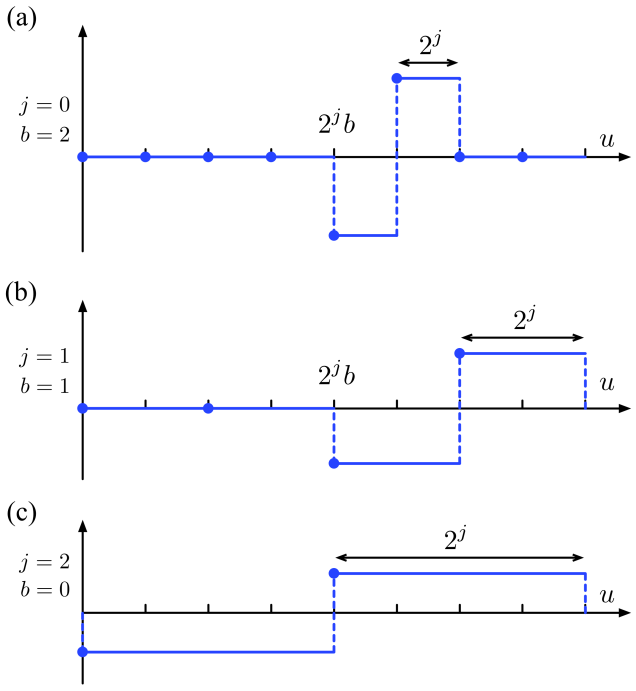

Figure 1. Three elements of the Haar wavelet basis $\{\boldsymbol{\psi_{j,b}}\}$ for various values of the scale index j and the translation index b. See text for details.

$$\boldsymbol{A_J x} = \boldsymbol{x}[0] - \sum_{j,b} \langle \boldsymbol{x} | \boldsymbol{\psi_{j,b}} \rangle \boldsymbol{\psi_{j,b}}[0] = \sum_{u<K} \boldsymbol{x}[u] \tag{3}$$

must be appended to the wavelet coefficients. Observe that $\boldsymbol{A_J x}$ computes a delocalized average of all signal coefficients, which can equivalently be formulated as an inner product with the constant function $\boldsymbol{\phi}[u] = 2^{-J/2}$ over the support $[\![0; K[\![$. Henceforth, it corresponds to the traditional chroma representation, where spectrogram bands of the same pitch class q are summed across all K octaves.

Since the wavelet representation amounts to K inner products in $\mathbb{R}^K$, its computational complexity is $\Theta(K^2)$ if implemented as a matrix-vector product. Implementing these inner products as convolutions would bring the complexity to $\Theta(K(\log_2 K)^2)$ by using the Fast Fourier Transform (FFT) algorithm. To improve this, Mallat has developed a multiresolution pyramid scheme

$$(\boldsymbol{x} * \boldsymbol{h})[u] = \frac{\boldsymbol{x}[u+1] - \boldsymbol{x}[u]}{\sqrt{2}} \tag{4}$$

$$(\boldsymbol{x} * \boldsymbol{g})[u] = \frac{\boldsymbol{x}[u+1] + \boldsymbol{x}[u]}{\sqrt{2}} \tag{5}$$

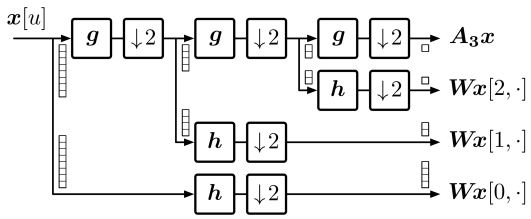

Figure 2. Discrete wavelet transform of a signal of length 8, as implemented with a convolutional pyramid scheme. See text for details.

	operations	memory
Matrix-vector product	$\Theta(K^2)$	$\Theta(K)$
Fast Fourier transform	$\Theta(K(\log K)^2)$	$\Theta(K)$
Convolutional pyramid	$\Theta(K \log K)$	$\Theta(K)$
In-place Haar pyramid	$\Theta(K \log K)$	$\Theta(1)$

Table 1. Computational complexity and memory usage of various implementations of the Haar wavelet transform, for a one-dimensional signal of length K. See text for details.

When applied to the chroma representation $\boldsymbol{X}(t, q, z)$, where t is a temporal window, q a chroma pitch class (an integer between 1 and 12), and z an octave, the Haar wavelet transform encodes the sums and differences in chroma bins across octaves. The number of output "bands" from the Haar wavelet transform, given by k, determines the number of scales J for which the basis Haar wavelet is dilated: $2^J = k$.

The use of the Haar wavelet transform in the feature extraction stage for our ACE system is motivated by the fact that it is stable to octave deformations – something that the multiband representation is not. By coding for chroma relationships across varying octave scales, we are able to capture representations at both low scales (less invariance, more stability) and higher scales (higher invariance, less stability), thereby reducing the variance of the GMMs.

Upon receiving a CQT-derived pitch representation of 8 octaves $\boldsymbol{X}[t, \lambda]$, the lowest and highest octaves are thrown out of a total of 8 octaves (this follows suit from the multiband chroma representation). $\boldsymbol{X}$ is therefore a matrix of size $T \times \Lambda$, i.e. the number of frames in the song T times the perceptual frequency bins $\Lambda = (12 \times 8) = 96$. A chroma representation is then created by reshaping $\boldsymbol{X}[t, \lambda]$ into a tensor of size $T \times Q \times Z$ where $Q = 12$ (number of chroma pitch classes) and $Z = k = 8$ (number of octaves as well as number of wavelet coefficients). In the sequel, we keep the notation $\boldsymbol{X}[t, q, z]$ for simplicity.

The Haar wavelet transform is then computed iteratively over the scale $J = log_2(k)$ and stored into a cell of size k, with each row in the cell containing a chroma matrix of size $T \times Q$ (same as in the multiband case, but with $k = 8$ cell rows instead of 4). At each scale $j \in [\![1, J]\!]$, the chroma tensor $\boldsymbol{X}_{t,q,z}$ is reshaped to a four dimensional tensor g from which the sum g and difference h coefficients are calculated:

$$\chi_{t,q,z} \rightarrow g_{t,q,w,v} \tag{6}$$

where

$$\text{size}(\chi) = (T \times Q \times Z) \qquad \rightarrow \qquad \text{size}(g) = (T \times Q \times 2 \times \frac{k}{2^j}) \tag{7}$$

h takes the difference of g along the third dimension ($\frac{\partial g}{\partial w} = h$). For each $v \in [1, \frac{k}{2^j}]$, the modulus $|h_{t,q,w,v}|$ is stored as a row in the output Haar wavelet representation cell. g is then summed along the third dimension (w) and squeezed down into a three dimensional tensor (singleton dimensions due to sum removed). After iterating at all scales, the resulting g is stored as the last (k'th) row in the output Haar cell array.

	operations	memory
Matrix-vector product	$\Theta(K^3)$	$\Theta(K^2)$
Fast Fourier transform	$\Theta(K^2(\log K)^2)$	$\Theta(K^2)$
Convolutional pyramid	$\Theta(K^2 \log K)$	$\Theta(K^2)$
In-place Haar pyramid	$\Theta(K \log K)$	$\Theta(1)$

Table 2. Computational complexity and memory usage of various implementations of the deep Haar scattering transform, for a one-dimensional signal of length K. See text for details.

4. DEEP HAAR SCATTERING

As opposed to the Haar wavelet transform representation, a Haar scattering representation concatenates sum and difference terms at different scales and then iterates. From the chroma representation $\chi_{t,q,z}$, we reshape to a tensor of dimension $J + 2$ called ϕ:

$$\chi_{t,q,z} \rightarrow \phi_{t,q,\sigma_1,\sigma_2,\ldots,\sigma_J} \tag{8}$$

where the size of ϕ is $T \times Q \times 2_1 \times 2_2 \ldots \times 2_J$. At each scale $j \in [1, J]$, we first create a copy of ϕ called $\hat{\phi}$, and then concatenate the sum of $\hat{\phi}$ along scale σ_j and the difference of $\hat{\phi}$ along σ_j, setting the result as ϕ. We then take the modulus $|\phi|$ and iterate.

5. RESULTS

At first, results were disappointing. As the two Haar methods operated over all octaves (i.e. number of octaves = number of coefficients), k was set to 8 for these trials, and the results for the Haar wavelet representation returned a dismal 44.3 %. Scattering performed better at 53.1 %, but

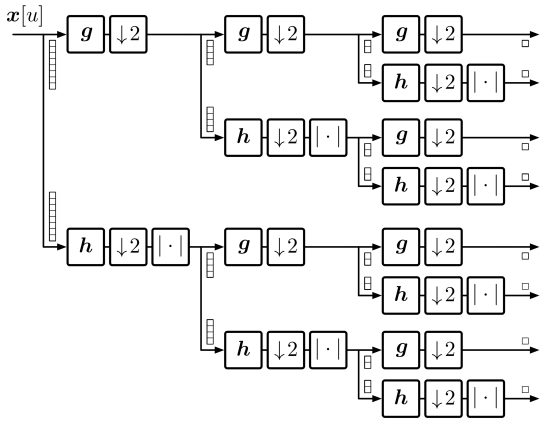

Figure 3. Deep scattering transform of a signal of length 8, as implemented with a convolutional pyramid scheme. See text for details.

neither came close to the multiband representation. However, when increasing the number of bands in the multiband representation from 4 to 8, accuracy for multiband plummeted to 47.6 %. This seemed to indicate a correlation between more coefficients and decreased performance – hinting that more stable representations needed to be used.

Mode	k	mirex measure
Multiband	4	**66.8** %
	8	47.6 %
Haar Wavelet	4	55.7 %
	8	44.3 %
Haar Scattering	4	60.2 %
	8	53.1 %

To decrease the number of coefficients in the wavelet representations from $k = 8$ down to $k = 4$, neighboring octaves were averaged together, and the wavelet coefficients were the constructed over a scale of $J = log_2(k) = 2$. Both Haar wavelet methods benefit from this decrease of scale, as the Haar Wavelet accuracy jumps up to 55.7 % accuracy (an increase of 11.4%) and the Haar Scattering accuracy becomes 60.2 % (an increase of 7.1 %). While still not quite as good as the multiband representation, Haar scattering begins to come a bit closer.

6. DISCUSSION

Haar wavelet analysis does not lead to terrible results, but nor does it achieve the same level of accuracy as the typical multiband chroma approach. Why? One reason is that the wavelet coefficients, both in the Haar wavelet transform and the deep Haar wavelet scattering methods, lack local context. This is done primarily to achieve octave invariance and to stabilize the chord recognition system to deformations, but the Haar representations seem to go too far by completely delocalizing along pitch.

Higher harmonics have significantly different behavior than lower ones. While in lower octaves the fundamental frequencies stand out, at higher octaves the harmonics start to clutter the chroma space at low amplitudes, making detection of the fundamental at upper harmonics very difficult. The multiband chroma representation creates, at every frame, k chroma representations which then vote on a chord using GMMs. Chroma vectors at high octaves are likely poorer voters than those at lower octaves, where fundamentals are more strongly represented. The Haar representations delocalize information completely along pitch, blurring the more useful information at lower octaves with the less useful information at higher octaves – resulting in less accurate voting.

7. REFERENCES

[1] Taemin Cho. *Improved Techniques For Automatic Chord Recognition From Music Audio Signals*. PhD thesis, New York University, 2013.

[2] Taemin Cho and Juan P. Bello. On the relative importance of individual components of chord recognition systems. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, 22(2):477–492, February 2014.

[3] Taemin Cho, Juan P. Bello, and Eric J. Humphrey. tmc-ace-mirex2013. `https://github.com/marl/tmc-ace-mirex2013`, 2013.

[4] Eric J. Humphrey and Juan P. Bello. Four timely insights on automatic chord estimation. In *ISMIR*, 2015.

[5] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. mir_eval: A transparent implementation of common mir metrics. In *ISMIR*, 2014.