Detection & Removal of Words

Preprocess

- We will work with an $n$-Level signal detection method
- Need each side of the image to be divisible by $2^n$

Original Image
Wavelet Transformation

- Apply an $n$-level transformation

Whoa, Whoa, Whoa.
Let’s back up a mile or two.

Padding

| LH | HH |

$L = \text{“low frequency”}$  \hspace{1cm}  $H = \text{“high frequency”}$  

$XY = \text{“Row” “Column”}$

$HL = \text{High Frequency Row, Low Frequency Column}$

Overview

- Signal Processing
- Fourier transforms
- Wavelets

Digital Signal Processing

- Discrete Fourier Transforms
- Discrete Wavelet Transforms

- There’s no reason for the signal to be 2D
- So, let’s begin with 1D

Signals

- Given a sequence of events, find the patterns or compress the signal
  - Ex: Physician’s pharmaceutical prescriptions over time
  - Ex: Number of patient records accessed by employees of the medical center

FYI – Many real sequences are periodic

What’s Going On

- Decomposition of a signal into a sum of sine (and cosine) waves
- Assesses similarity through the wave

Waves with frequency 0, 1, …

- Use the inner product to measure similarity (a.k.a. cosine similarity)
Waves with frequency 0, 1, …n-1

Use the inner product to measure similarity (a.k.a. cosine similarity) between your sequence and the expected frequency of the wave(s).

Distance between sequence / set in vector space

\[ \cos(x, y) = 1 - \frac{\sum_{i=0}^{n} x_i y_i}{\sqrt{\sum_{i=0}^{n} x_i^2 \sum_{i=0}^{n} y_i^2}} \]

Waves with frequency 0, 1, …

Use the inner product to measure similarity (a.k.a. cosine similarity)

Wave 0: No wave

\[ \cos(x, y) = 1 - \frac{\sum_{i=0}^{n} x_i y_i}{\sqrt{\sum_{i=0}^{n} x_i^2 \sum_{i=0}^{n} y_i^2}} \]

\[ \cos(x, y) = 1 - \frac{\sqrt{118.9}}{\sqrt{141 \times 117.67}} \]

\[ \cos(x, y) = 0.915 \]

Waves with frequency 0, 1, …

Use the inner product to measure similarity (a.k.a. cosine similarity)

Frequency f = 0
Intuition

- All basis functions are n-dimensional vectors
- Orthogonal to each other
- Calculate similarity of your sequence through inner product with each of functions
- DFT result – approximately all of the similarities of your sequence with the basis functions

Euler’s Identity

\[ e^{j \theta} = \cos(\theta) + j \sin(\theta) \]

Computation

\[ X_n = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp \left( -j \frac{2 \pi t}{n} \right) \]

Caveat: Euler

\[ e^{jx} = \cos(x) + j \sin(x) \]

Let’s say \( x = \pi \)

\[ e^{j\pi} = \cos(\pi) + j \sin(\pi) \]

We know

\[ \cos(\pi) = -1 \]
\[ \sin(\pi) = 0 \]

So, \( e^{j\pi} = -1 + 0 = -1 \)

DFT

- You’ll find DFT in most math and signal processing software
- Matlab
  - Use the “FFT” (Fast Fourier Transform – we’ll come to this in a second)
  - \( A = [10, 20, 10, 20, 10, 20, 10, 20] \)
  - \( B = \text{fft}(A) \)
  - \( \text{plot}(|B|) \)

- \( X_n’s \) are complex numbers… except
  - \( X_0 \Leftarrow \) this is a real number

- However, we have symmetry

\[ X_f = \overline{X}_{n-f} \]

- \( \overline{X} \) is the complex conjugate: \( x + yj = x - yj \)
Amplitude

- Amplitude Spectrum: \(|X_f|\) vs. \(f=0,1,2,\ldots, n-1\)
- Symmetric
  (so, you only need to plot the first half)

Ex: Impulse

- \(x = \{0, 1, 0, 0\}\)
- \(X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp(-\frac{j 2 \pi f t}{n})\)
- \(X_0 = \frac{1}{\sqrt{4}} \sum_{t=0}^{4-1} x_t \exp(-\frac{j 2 \pi 0 t}{4})\)
- \(X_1 = \frac{1}{\sqrt{4}} \sum_{t=0}^{4-1} x_t \exp(-\frac{j 2 \pi 1 t}{4})\)
- \(X_2 = \frac{1}{\sqrt{4}} \sum_{t=0}^{4-1} x_t \exp(-\frac{j 2 \pi 2 t}{4})\)
- \(X_3 = \frac{1}{\sqrt{4}} \sum_{t=0}^{4-1} x_t \exp(-\frac{j 2 \pi 3 t}{4})\)

Ex: Impulse

- Symmetry!
- \(X_0 = \frac{1}{2}\)  
- \(X_1 = -\frac{1}{2}j\)  
- \(X_2 = -\frac{1}{2}\)  
- \(X_3 = \frac{1}{2}j\)

"Frequency leak": Suggests we need \(n\) numbers in frequency domain to represent 1 number in the sequence domain!
2D DFT

- Allows for two-dimensional coordinates
- Great for images!

\[ X_{f_1,f_2} = \frac{1}{\sqrt{n_1}} \frac{1}{\sqrt{n_2}} \sum_{\alpha=0}^{n_1} \sum_{\beta=0}^{n_2} x_{\alpha,\beta} \exp(-2\pi j f_1 \alpha / n_1) \exp(-2\pi j f_2 \beta / n_2) \]

Tips and Tricks

- Computing the DFT is time consuming!

\[ X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp\left(-\frac{j 2\pi ft}{n}\right) \]

- Basically, for each \( f \), you compute \( n \) steps.
- Since \( f \sim n \), it’s \( O(n^2) \)

FFT

- Fast Fourier Transform (Cooley & Tukey 1965)
- When \( n \) is a power of 2, we can reduce the complexity to \( O(n \log n) \)
- Uses a divide-and-conquer strategy
- Recursively splits DFT into two DFTs with \( O(n) \) multiplications of complex roots
- Split the polynomial into two polynomials
  - One with the even powers
  - One with the odd powers

Points of Contention

- DFT \( \rightarrow \) Great for farming the periodicity in data
- But what about other features in the pattern…
  - Change points?
  - Overlapping patterns?
- Compression of pattern
  - Burst of lots of information
  - Long periods of silence

Short Window Fourier Transform

- Reset or partition the window of analysis according to size of expected sequence

- Question: How can we determine the length of the window?
  - If we know the font size beforehand – great!
  - But what if we don’t?

Overview

- Signal Processing
- Fourier transforms
- Wavelets
Enter the Wavelet
(Discrete Wavelet Transform)

- Use multiple window sizes simultaneously!

<table>
<thead>
<tr>
<th>Time Rep.</th>
<th>DFT</th>
<th>SWFT</th>
<th>DWT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Time</td>
<td>Time</td>
<td>Time</td>
</tr>
</tbody>
</table>

Haar Wavelet

- Subtract the sum of the left half from the right half
- Repeat – recursively – for quarters, eighths, etc.

Building the Haar

- Map each coefficient to the time-frequency plane

Compression of Music
Haar

- Orthonormal bases in the [0,1] range
  - $h_0(x)$, $h_1(x)$, …
  - beyond the Fourier bases

Mother Bases

- $h(x) = \begin{cases} 1 & 0 \leq x < 1/2 \\ -1 & 1/2 \leq x < 1 \\ 0 & \text{o.w.} \end{cases}$

- $h_j(x) = 2^{j/2} h(2^j x - k)$

- $n = 2^j + k$

- $k 2^{-j} \leq x < (k+1)2^{-j}$

- $0 \leq k < 2^j$

Daubechies

- Also uses Orthonormal bases
- But there’s no closed form to explain them
- Created using a cascade algorithm (See Wavelet Theory: Chapter 6, https://onlinelibrary.wiley.com/doi/10.1002/9781119146462.ch6)

- Calculate wavelet coefficients iteratively
- Each scale leads to the coefficient of the next scale
Daubechies

- Also uses Orthonormal bases
- But there's no closed form to explain them
- Created using a cascade algorithm (See Wavelet Theory: Chapter 6 - https://onlinelibrary.wiley.com/doi/10.1002/9781118166652.ch6)
  - Calculate wavelet coefficients iteratively
  - Each scale leads to the coefficient of the next scale

Detection & Removal of Words

- Text Detection
- Text Selected Elimination
- Wavelet Transform
- Preprocess

Preprocess

- We will work with an "n-Level" signal detection method
- Need each side of the image to be divisible by 2^n
- If the image length is not divisible by 2^n
  - Pad the image with black / white

Process

- Transform via Daubechies-4 wavelet basis
- Wavelet transformation results in frequency "bands"

Wavelet Transformation

- Apply an n-level transformation
- Highest level transformer contains at least 8 x 8 pixels in the smallest band of the image
Wavelet Transformation

- LH: Horizontal edges at particular scale
- HL: Vertical edges
- HH: Diagonal edges

- Suggested as best for dealing with separation of sections with and without text
- Use lowest 3 to 4 levels for text detection and elimination

<table>
<thead>
<tr>
<th>LL</th>
<th>HL</th>
</tr>
</thead>
<tbody>
<tr>
<td>LH</td>
<td>HH</td>
</tr>
</tbody>
</table>

3-Level Transformation

Image

HL Band

LH Band

Need to post-process HH Band to avoid elimination of non-text (details in Wang’s chapter – see reading)

Information Retention

- Text eliminated, Arrows Maintained