Mobile Sensing VII – Audio Sensing

Spring 2015
Petteri Nurmi
Learning Objectives

- Understand basics of microphone sampling, data representation and noise characteristics
- What is windowing and why is it important? What is the Short Time Fourier Transformation and how it differs from other Fourier transformations?
- What kind of features are typically extracted in audio sensing applications?
- How to compare audio similarity? What types of similarity measures exist and which tasks they support?
Sensors: Microphone

- Mobile phone microphones optimized for transmitting speech across mobile networks
  - Modulation used to encode signals
  - Microphone frequency response optimized for speech range
- Pulse Code Modulation (PCM)
  - Amplitude of signal sampled at uniform intervals, each sample quantized to nearest value within a range
  - Standard for digital audio
  - 8-bit: Mono, 16-bit: Stereo
  - Sampling rates: 8000kHz, 16000 kHz, 44100 kHz
  - Maximum frequency captured by 44100 kHz is 20 kHz
    - ultrasound covers frequencies > 20 kHz → not feasible on most mobile devices
Audio Noise

- Noise referred to as the residual low level sound that is heard during “silent” periods (hiss and hum)
- Two main types of noise:
  - White noise: random signal with constant power spectral density
  - Pink noise: $1/f$ noise, spectral density of noise inversely proportional to frequency
- In addition to residual noise, many sources of constant ambient noise
  - Airconditioning units, distant chatter, etc.
  - All influence the nature of the captured signal
Frequency Response

- Frequency response is a quantitative measure of the output spectrum of a system in response to stimulus.
- Microphone frequency response indicates how the device captures different frequencies.
  - Typically optimized for human speech ➔ frequencies below 50Hz reduced and higher frequencies amplified.
  - Noise filtering, especially through high-pass/low-pass/band-pass filters influence response.
- Generally ANY two different mobile phone models have different frequency responses.
  - Similarly using built-in microphones vs. headset microphones affects frequency response.
Audio Sensing Pipeline

- General mobile sensing pipeline, but with certain unique characteristics

1. Sensor data extraction
   - Audio signals non-stationary by nature ➔ short frames
   - Short frames cause discontinuities ➔ need to consider overlapping in frame construction

2. Preprocessing
   - Signals consistently corrupted by “hiss and hum”
   - Non-stationary signals cause spectral leakage ➔ windowing required

3. Feature extraction
   - Mainly spectral features due to the nature of signals

4. Classification
   - Non-stationary signals ➔ temporal dependencies need to be modelled, e.g., HMMs widely used
Fourier Transformation Revisited

- Fourier transformation has been designed to capture the periodicity of a signal
  - Assumes periodicity constant over time
  - Does not localize frequencies in time
- Periodicity of speech and most other audio sources varies over time
  - E.g., each spoken phoneme has different format
- Taking discrete FFT of non-periodic signal causes “smearing” of signal energy across a wide-range
  - Referred to as *spectral leakage*
Window functions refer to mathematical functions that are zero-valued outside some interval.

Essential in spectral analysis to reduce spectral leakage and discontinuities at start and end of frame.

Applying a window function on the signal before taking FFT can reduce effects of leakage:
  - Cannot eliminate, actually only change shape of leakage
  - Trade-offs: frequency resolution, amplitude accuracy, and overall extent of leakage

Either multiply the signal by the window, or multiply the FFT of signal by the FFT of the window.

(Most) windows zero at end and start points.
Rectangular Window

- Defined as \( w(n) = 1.0 \) \( (n = 1, \ldots, N) \)
  - Corresponds to a filter that passes through exactly \( N \) points
- Simplest possible window, preserves main frequency accurately, but poor in other aspects
  - Also known as Boxcar and Dirichlet window
- Included in all presentations about windowing mainly due to completeness
  - Suitable for transient signals, otherwise causes discontinuities which degrade DFT
Triangular Window

- Can be seen as convolution of two N/2 width rectangular windows
  - Better against spectral leakage and amplitude preservation, but suffers against discontinuities
- General form: \( w(n) = 1 - |(n - (N - 1)/2) / L/2 | \)
- When \( L = N - 1 \), corresponds to a Bartlett window
  - Triangular window non-zero at ends (0 and L)
  - Bartlett window zero \( \rightarrow \) better against discontinuities

**Matlab:**
- `triang(length)`
- `bartlett(length)`
Kaiser Window

- Window function that attempts to provide trade-off between leakage and additional smearing
- Popular in audio sensing, but also for ECG and other high frequency signals with regular noise
- Good for recovering amplitude, less so for resolving frequencies
- Formally
  - \( N \) is length of sequence
  - \( I_0 \) is zero\(^{th} \) order Bessel function
  - \( \alpha (> 0) \) parameter that determines shape of window

Matlab: kaiser(length, alpha)
Hann(ing) Window

- General purpose window for signal analysis, also known as the raised cosine window
- Fast side lobe decrease → good at reducing the extent of leakage
- Relative large difference between main lobe and side lobes → good frequency resolution
  - But respectively reduces amplitude resolution

\[ W(n) = 0.5 \left(1 - \cos\left(\frac{2\pi n}{N - 1}\right)\right) \]  

Matlab: hann(length)
Hamming Window

- Popular window choice in audio processing
  - Simple to implement
  - Provides good frequency resolution, but only moderate amplitude accuracy and leakage
- Variant of Hann window that minimizes the nearest side lobe
  - And thus provides better frequency resolution for the main peak
  - Formally: \( W(n) = \alpha - \beta \cos((2\pi n / (N - 1))) \)
  - Formally: \( \alpha = 0.54 \)
    \( \beta = 1 - \alpha = 0.46 \)

**Matlab:** `hamming(length)`
Other Windows

- Generalized cosine windows
  - General form: \( w(n) = \sum a_k \cos((2\pi kn)/N) \)
  - If \( K = 1 \), called generalized Hamming windows
  - Mainly used in other applications than spectral analysis
  - Blackman, Nutall, Blackman-Harris, ...
- Tukey window
  - Tapered cosine window
  - Cosine lobe convolved with a rectangular window
  - By controlling a “convolution” parameter, can be mapped back into rectangular or Hann window
- And a large amount of other windows
  - Riesz, Riemann, Parzen, Bohman, Gaussian, etc.
Windowing: Example

- Original data
- Triangular
  - Rectangular
  - Kaiser
  - Hamming
  - Hann
Summary of Window Functions

- Audio sensing: basics
  - Construct frames with overlap to avoid discontinuities caused by preprocessing
  - Use windowing to reduce spectral leakage and to amplify either amplitude or frequency resolution
- Windowing *almost always* performed in audio sensing as a preprocessing step
- No “ultimate” window function, best choice depends on application
  - Filters that are best in some aspect (frequency/amplitude/leakage) are typically bad in another aspect
  - Kaiser, Hann, and Hamming filters generally most widely used due to their balanced nature
Short Time Fourier Transform (STFT)

- Short Time Fourier Transform (STFT) examines frequency and phase content of signals locally
  - Basic tool in audio analysis
  - If frame-size sufficiently small, signal can be assumed to be approximately stationary
- Calculated by using Fourier transform on windowed audio frames
  - Each FT provides spectral information of a single slide of the audio signal provides simultaneously time and frequency information
- Formally: \( \text{STFT}_{f, \omega}(t', \omega) = \int [f(t) - W(t - t')] e^{-j2\pi\omega t} \, dt \)
- Time-frequency trade-off:
  - Long frames: excellent frequency, but bad time separation
  - Short frames: excellent time, but bad frequency separation
Spectrogram formally defined as the squared magnitude of STFT, i.e.,

\[ \text{spectrogram}(t, \omega) = |\text{STFT}_{f, \omega}(t, \omega)|^2 \]

- Used particularly for studying speech and music
  - Pitch, formants, and related features can be observed from the spectrogram contents
- Intuitive description:
  - Take the FFT of each (windowed) frame
  - Rotate the normalized FFT
  - Convert FFT into heatmap based on amplitude
  - Aggregate over time
Spectrogram: Example

**Matlab**

```matlab
matlab: load gong.mat
```

```matlab
matlab: w = 1024;
spectrogram(y,kaiser(w,2),w/2,w,Fs,'yaxis')
```
Feature Extraction

• *Most* audio sensing builds on spectral features, i.e., those extracted from Fourier transformed signals
  • Speech, music, and many other everyday audio signals of interest are periodic
  • Large hardware variations in microphones, combination of multiple spectral features typically better
• The most important time-domain features are related to zero crossings
  • \( \text{ZCR} = \sum |\text{sign}(s_i) - \text{sign}(s_{i-1})| / 2 \)
  • Particularly variation of ZCR informative
Spectral Features

• Power and sub band power
  • Power defined (analogously to before) as the sum of the (windowed and normalized) FFT of a frame: \( P = \sum |F(\omega)|^2 \)
  • Sub band power: power calculated over a specific sub band (i.e., frequency range)
    • Technical note: matlab, consider only first N/2 coefficients of FFT

• Spectral rolloff
  • Frequency bin below which the majority of the power is concentrated: \( \max(\omega_h : \sum |F(\omega)|^2 < \varepsilon) \)

• Low/high energy frame rate
  • Number of frames within a “frame window” that have energy below/above a threshold
    – E.g., Frame width = 32ms and frame window 1-2 seconds
Spectral Features

- Consider normalized FFT magnitudes $p(\omega)$
- Spectral Centroid SC (Brightness)
  - Balancing point (centroid) of the frequency spectrum: $SC = \frac{\sum \omega \ p(\omega)^2}{\sum p(\omega)^2}$
- Bandwidth BW:
  - Width of the range of frequencies occupied by the signal
  - $BW = \frac{\sum [(\omega - SC)^2 p(\omega)^2]}{F(\omega)^2}$
- Spectral flux SF
  - L2 norm of the difference of (normalized) spectral amplitude between two frames: $SF = \sum (p_t(\omega) - p_{t-1}(\omega))^2$
- Spectral entropy H
  - $H = - \sum p(\omega) \log p(\omega)$
1. Apply windowing on the signal, in this case Hamming window
   - Frame-size 400ms, sampling rate 44.1kHz
2. Calculate FFT of the windowed signal and normalize the FFT coefficients
3. Spectral entropy:
   - \[ SE = - \sum p(\omega) \log p(\omega) = 9.87 \]
4. Spectral centroid:
   - \[ \frac{\sum [\omega p(\omega)^2]}{\sum p(\omega)^2} = 1647 \]
5. Spectral roll-off 95%:
   - \[ \max(\omega_h : \sum |F(\omega)|^2 < \varepsilon) = 14582 \]
MFCC (Mel Frequency Cepstral Coefficient)

- MFCC coefficients model spectral energy distribution in a perceptually meaningful way
- Widely used feature in audio analysis, particularly in the analysis of speech signals
- Computed as follows:
  1. Apply window function on frame, take FFT / STFT
  2. Map power of spectrum to mel scale using triangular overlapping windows
     - So-called Mel-spaced filterbank, typically 20-40 windows
  3. Take log of the powers at each of the mel frequencies
  4. Take discrete cosine transform of the mel log powers
  5. MFCCs are amplitudes of resulting spectrum
• MFCC is motivated by so-called Mel scale
• Perceptual scale of pitches judged by listeners to be equidistant from each other
  • Reference level at 1000Hz equals 1000 on mel scale
  • Non-linear scale, after about 500Hz increasingly large intervals are judged to produce equal increments
  • Mel refers to melody, scale based on pitch comparisons

\[ M = 2595 \log_{10} (1 + f/700) \]

Source: [http://en.wikipedia.org/wiki/Mel_scale](http://en.wikipedia.org/wiki/Mel_scale)
Audio Activity Detection

- Audio data contains many “uninteresting” portions that mainly consist of background noise
  - Audio processing CPU intensive ➔ filtering out unwanted periods improves energy-efficiency
- Two main features used:
  - (Log-)Energy: estimates overall level of audio activity
    - Simplest approach is to apply a fixed (or adaptive) threshold on the log-energy
  - Spectral Entropy:
    - Recall that spectral entropy was calculated by mapping the FFT coefficients into probabilities
    - Flat spectrum results in high entropy ➔ can be used to detect frames consisting mainly of noise
Adaptive threshold determination for audio activity detection using log-energy

1. Calculate a smoothed log-energy sequence $e'$
   - Mean-filtering of the log-energy, original article uses sliding window of 9 10ms frames

2. Estimate average and minimum values of the smoothed log-energies over a window of size $W$
   - Denote these by $m_i$ and $a_i$, original article uses $W = 2$ seconds

3. Define threshold $\tau_i = m_i + \max\{2, 0.7 \times (a_i - m_i)\}$
   - Assume audio activity takes place if the smoothed log-energy exceeds the threshold
Example

- Original signal
- Log energy
- Spectral entropy
- Audio activity
- Noise
Audio Similarity: Time-Domain

- Cross-correlation
  - Cross-correlation defined as the similarity of two signals as a function of lag (see Lec III)
  - Can be used as a similarity measure for audio: 
    \[ \text{sim}(x,y) = \max R_{x,y}(m) \]
- Absolute difference / Euclidean distance
  - \[ \text{sim}(x,y) = 1 - \| x - y \| \]
- Pros and cons
  - Frequencies with higher volume dominate the similarity ➔ somewhat resilient against hardware-specific noise characteristics
  - Simple to compute
  - But also easy to “break”, e.g., playing a loud sound

Matlab
Cross-correlation:
\[ [r, lags] = \text{xcorr}(A,B); \]
\[ \text{sim} = \max(r) \]
\[ = 0.09 \]

...or around 0.02 if calculate frame-wise
Audio Similarity: Frequency-Domain

- Correlation and deviation in frequency domain can be used as (simple) alternative to time domain similarity
- Perform FFT of two signals and compare correlation/distance between FFT coefficients
- For weak signals (background noise) easily high similarity
- Similarities can be also combined, e.g., a time-frequency similarity:
  - \( \text{sim}(i,j) = 1 - \sqrt{d_{\text{time}}(i,j)^2 + d_{\text{frequency}}(i,j)^2} \)
Audio Similarity: Fingerprinting

- Audio similarity has been widely studied in the context of music information retrieval
  - Goal is to create a fingerprint that is robust to distortions and noise, and can be compared across devices
  - Many of the techniques have focused on music retrieval so need concise representation of the signal
- Shazam:
  - Similarity calculated by comparing spectrogram peaks
  - Peaks hashed for retrieval purposes
- Waveprint (Google Music):
  - Spectrogram considered as an image and a wavelet representation is fitted on the image
  - Signs of wavelet coefficients considered as hash for retrieval
Audio Similarity: Others

- Music similarity measures
  - Many other music-related similarity measures, which aim at preserving pitch, timbre, rhythm, and other musical expression parameters
- Hamming distance
  - Differences between FFT frequency bands of successive frames converted into a binary vector
  - Bit difference of two binary vectors used as similarity
  - Requires careful synchronization between samples
Summary

• Microphone a highly useful, but complex sensor
  • Different frequency responses, noise characteristics, and preprocessing
• Audio signals typically non-stationary → need windowing and STFT analysis to capture periodicity
• Most audio sensing applications operate using spectral features
• Audio activity detection can be used to improve energy-efficiency by discarding uninteresting parts
• Wide range of similarity measures, optimal choice depends on application
References


References


• Haitsma, J. & Kalker, T., A Highly Robust Audio Fingerprinting System, ISMIR, 2002


• Schürmann, D. & Sigg, S., Secure Communication Based on Ambient Audio, IEEE Trans. Mob. Comput., 2013, 12, 358-370