MP1 - Speech and Speaker Recognition with Nearest Neighbor

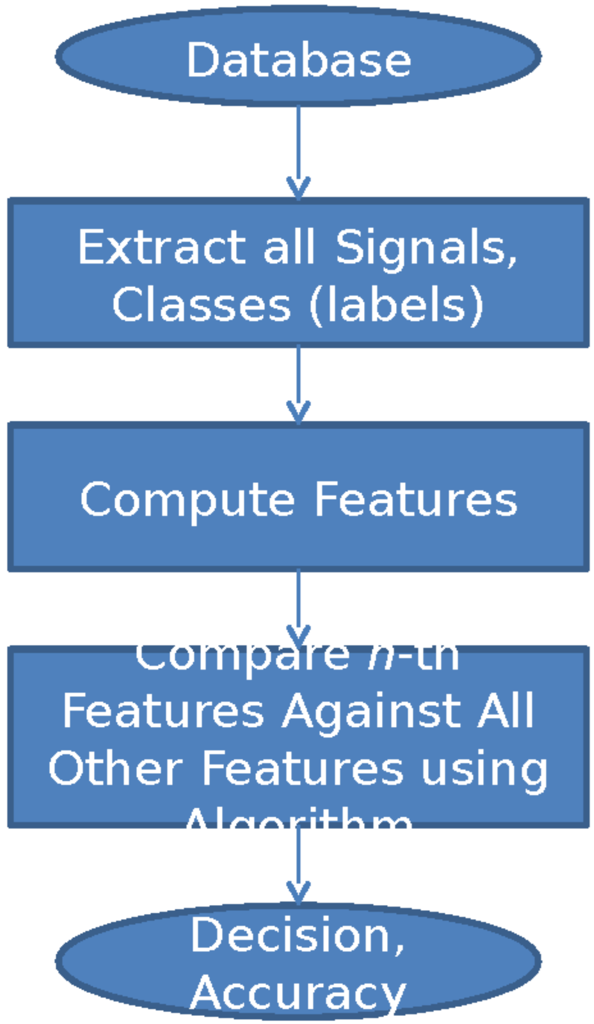
ECE417 – Multimedia Signal Processing

Spring2015

Goals

- Given a dataset of N different audio samples of people speaking, be able to:
  - Extract cepstral features
  - Use nearest neighbor to perform speaker recognition
  - Use nearest neighbor to perform speech recognition

The System: Highly Generalized



100 audio signals

Store all your signals

Compute the cepstrum, using some window size, some overlap

Use nearest neighbor distance metric

The Data

- 100 different audio samples
  - 4 different speakers, labelled A,B,C,D.
  - 5 different spoken digits, labelled 1,2,3,4,5.
  - Various instances, labelled a,b,c,d,e
- File format:
  - Name is [Speaker][Digit][Instance].wav
- Each audio sample is called an *observation.*

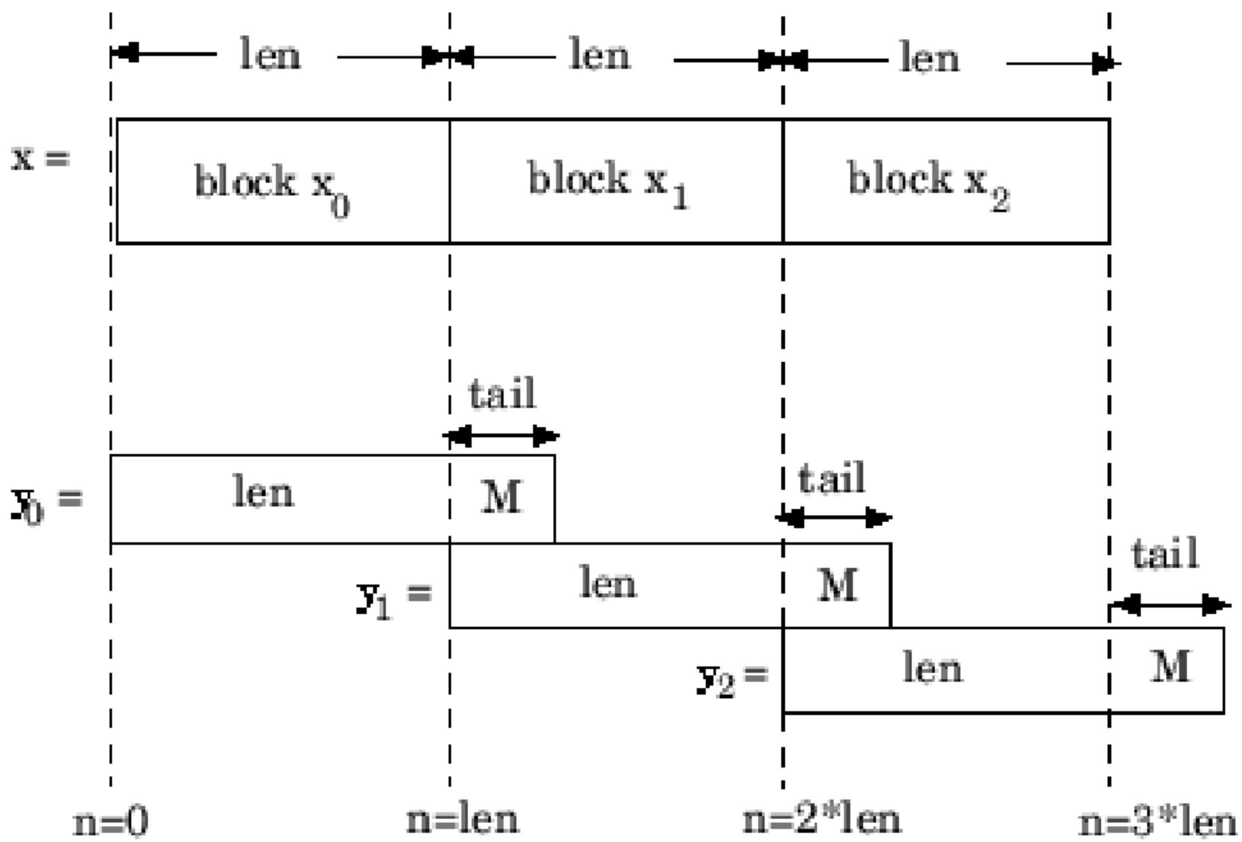
Audio Extraction

- Each file is a .wav file.
- Use audioread (or wavread) to extract the data.
- Result is stereo vector (2 channels).
- Linear interpolation is needed as the audio file may be of slightly different lengths.

Signal Pre-conditioning

- Speech is a non-stationary, time-varying signal.
- For most phonemes, the properties of the speech remain invariant for a short period of time (5-100 ms)
- We want to separate the signal into overlapping frames.
  - 10% Overlap
- Since we are inherently windowing the signal anyway by chopping it up into frames, let’s use a window with small sidelobes.
  - Hamming window

Signal Pre-Conditioning (Continued)



The Cepstrum

- General formula:
  $c[n] = \mathcal{F}^{-1}\{\log_{10}|\mathcal{F}\{x[n]\}|\}$
- The FFT and IFFT function in Matlab's signal processing toolbox will be useful here.
  - Implementation of the FFT itself is best left to courses on the computational aspects of DSP…
- Apply the cepstrum calculation to each frame of windowed data. If there are M frames of length L, you get an LxM matrix.
  - We are only concerned with the first 12 coefficients. This reduces your matrix to 12xM.
- Unroll this into a single column vector that is (12M)x1

K-Nearest Neighbor

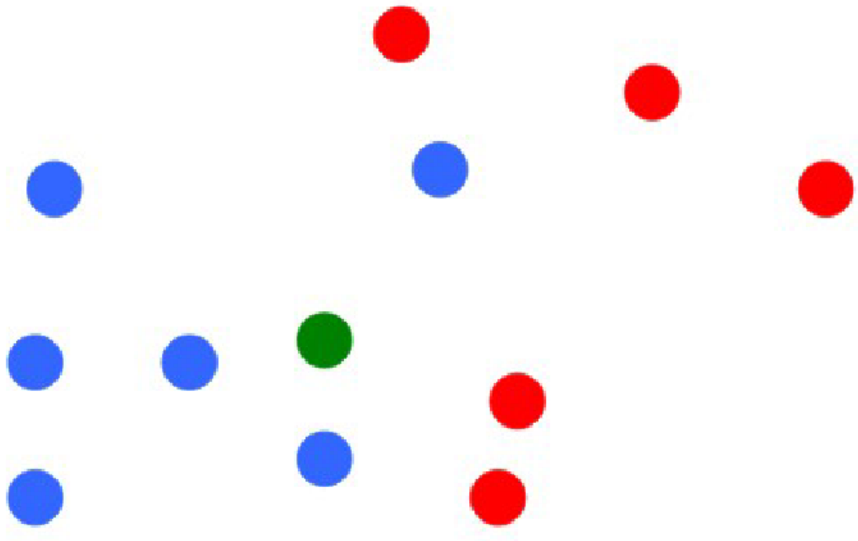
- Non-parametric approach (uses data, not model)
- Steps:
  - Given a data point, find the distance between that point and all other points in the dataset.

$$d_{mn} = \sum_{k=0}^{K-1} |c_m[k] - c_n[k]|^2$$

  - Find the K closest distances and corresponding classes to the given point.
  - Of these, find the class that occurs the most. This class is the one that matches the input data.

Illustration

- The green dot should be classified as what class (blue or red)?



Overall Procedure

- Get all audio samples, resize them so that they are all the same length.
- Compute the cepstrum using 10% overlap, a Hamming window, and various window sizes (100, 500, 10000).
- Perform nearest neighbor:
  - Remove all data corresponding to the same speaker for every sample when doing speech recognition.
  - Remove all data corresponding to the same digit for every sample when doing speaker recognition.

A Few Tricks for Removing Data and Indexing

- Logical Indexing will be your friend in this MP
- Separate labels into arrays of strings
- Store data in cells or arrays
- Use commands like ‘find’ or just use logical indexing to find indices of like classes (labels)
- Some Example Code
  - CurrentData = data{1};
  - CurrentSpeaker = SpeakerLabel(1,:);
  - CurrentDigit = SpeechLabel(1,:);
  - IndToKeep = find(SpeechLabel~=CurrentDigit);
- More Matlab Examples…

Results (Tables)

- Tables for the 1NN and 5NN results for each speaker, digit, and overall accuracy in both cases.
- This must be done for the raw data case and the 3 window-length cases.

Results (Tables)

Speech Recognition

	"Raw Features"	Cepstrum, W=100	Cepstrum, W=500	Cepstrum, W=10k
1-NN	D1: 0% D2: 30% D3: 15% D4: 50% D5: 0%	D1: 40% D2: 75% D3: 90% D4: 65% D5: 80%	D1: 45% D2: 100% D3: 85% D4: 45% D5: 85%	D1: 15% D2: 80% D3: 80% D4: 50% D5: 40%
	Overall: 19%	Overall: 70%	Overall: 72%	Overall: 53%
5-NN	D1: 0% D2: 45% D3: 5% D4: 30% D5: 0%	D1: 55% D2: 65% D3: 80% D4: 50% D5: 80%	D1: 55% D2: 80% D3: 80% D4: 60% D5: 70%	D1: 10% D2: 80% D3: 75% D4: 50% D5: 70%
	Overall: 16%	Overall: 66%	Overall: 69%	Overall: 57%

Results (Graphs)

- Graphs for
  - Recognition Accuracy vs Window Length for 12-Coefficient Cepstrum Using 1-NN (individual digits and overall results overlayed)
  - Recognition Accuracy vs Window Length for 12-Coefficient Cepstrum Using 5 NN (individual digits and overall results overlayed)
  - Speaker Recognition vs Window Length using Nearest Neighbor (individual speakers and overall results overlayed)
  - Speaker Recognition vs Window Length using 5-Nearest Neighbor (individual speakers and overall results overlayed)

Results (Graphs)