MP3 – Speech and Speaker Recognition with Nearest Neighbor

ECE417 – Multimedia Signal Processing

Fall 2017

Goals

- **Given a dataset of N audio files:**
  - **Features**
    - **Raw Features, Cepstral (Hz), Cepstral (Mel)**
  - **Classifier**
    - **k-NN**
  - **Tasks**
    - **Speaker Recognition**
    - **Speech Recognition**

The High-Level System

100 Audio Files (4 speakers*5 words*5 reps).

Do pre-processing (e.g., resizing), and store.

Slide window down the signal stream using some overlap. Compute cepstrum for each windowed segment in each signal (file).

Compute distance between the features in the one signal and features in other signals. Which one(s) is(are) closest?

Report accuracy in tables and graphs.

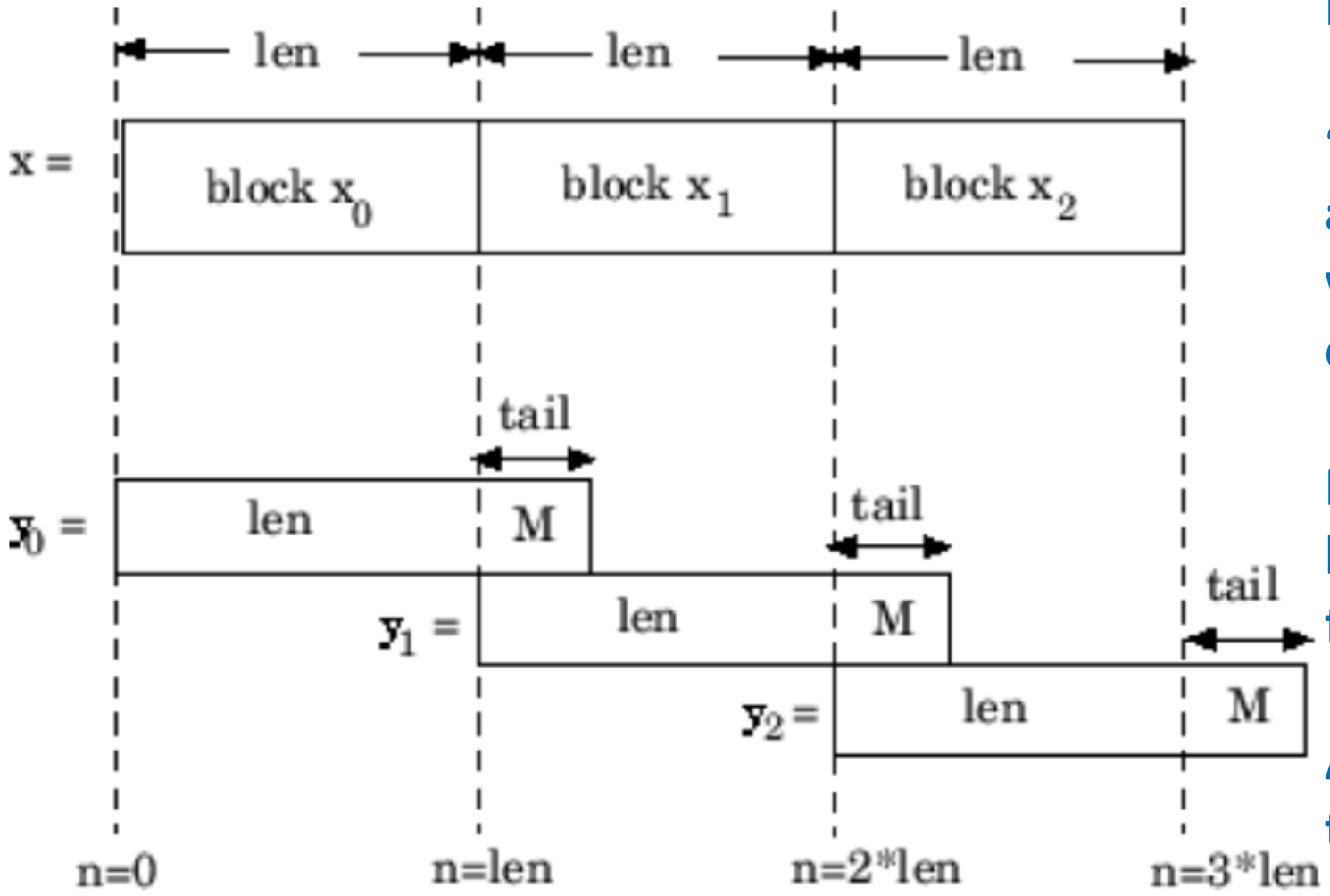
The Data

- **100 different audio files**
  - **4 speakers: A,B,C,D.**
  - **5 digits: 1,2,3,4,5.**
  - **5 utterances (instances) of each digit: a, b, c, d, e.**
- **File format**
  - **[Speaker][Digit][Instance].wav**
- **Example:**
  - **A = {A1a, A1b, A1c, A1d, A1e, A2a, …, A2e, A3a, …, A3e, A4a, …, A4e, A5a, …, A5e}**
  - **|A| = 25**
- **Each audio file is called an observation.**

Audio Extraction and Storage

- **Each file is a .wav file with sampling freq = 22050 Hz.**
- **Use audioread to read in the data.**
  - [data, Fs] = audioread(wav_name)
- **Result is a stereo vector (2 channels); take the left channel.**
  - data(:,1)
- **Use linear interpolation (imresize) to resize each signal to the same length (e.g., T = 10000 samples)**
  - imresize(data(:,1),[T ,1]);
- **Store the resized, left channel of each signal.**

Signal Pre-Conditioning (signal → frames)



Frame/Window size = len + M.

'len' = # samples to advance the frame = # samples which do not overlap between current and next frame

M = # samples that overlap between current and next frame

Apply a hamming window to each frame.

Signal Pre-conditioning (signal → frames)

- **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).**
- **Separate the signal into overlapping frames or windows; slide the frame in time down the length of the signal.**
  - **10% overlap**
  - **Example: If window size = 100 samples, overlap size = 10 samples. Then, (from prev slide) len = 90, M = 10, len + M = 100**
- **Multiply frame with a Hamming window to reduce side lobes.**
- **Complete the missing parts in sig2frames.m**
  - frames = sig2frames(signal, Nw, No) % Nw = window size, No = overlap size

Frame → Cepstrum (Hertz Scale)

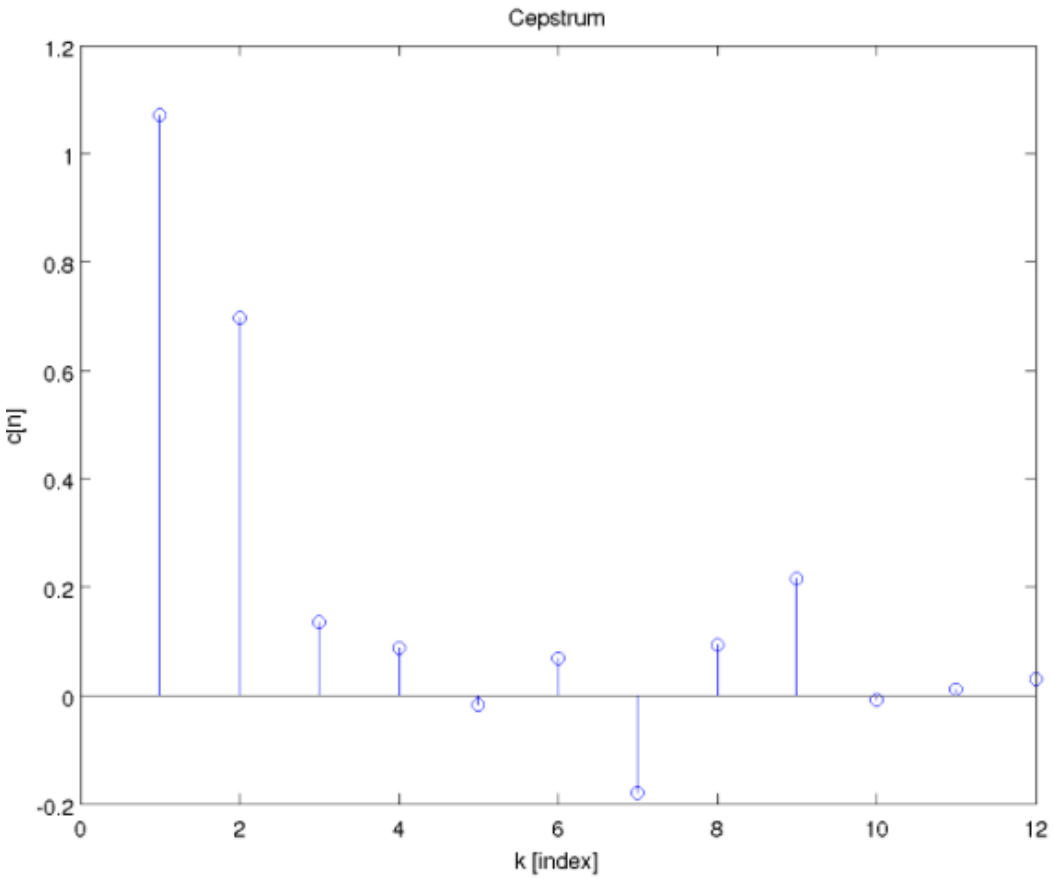
- General formula: $c[n] = \mathcal{F}^{-1}\log|\mathcal{F}(x[n])|$

- Inbuilt **fft**, **ifft** functions in **Matlab** are useful here (you don't need to implement FFT yourself!)
- Calculate the **cepstrum for each frame**. For each signal, you get an **NFFT x Nf matrix** , **Nf = # frames, NFFT = # number of FFT points**.
- But we only care about **the first Ncc = 12 coefficients**, so the matrix reduces to **Ncc x Nf.** Each col. is a vector of cepstral coeffs.
- **Complete the missing parts in cepstrum.m**
  - function CC = cepstrum(signal, Ncc, Nw, No) % here Ncc = 12, CC = [Ncc x Nf]

- Unroll this matrix (CC) into a **single column vector** that is **(Ncc*Nf)x1**.

The Cepstrum (Example Plot)

- Example: Cepstrum with the first coefficient removed.

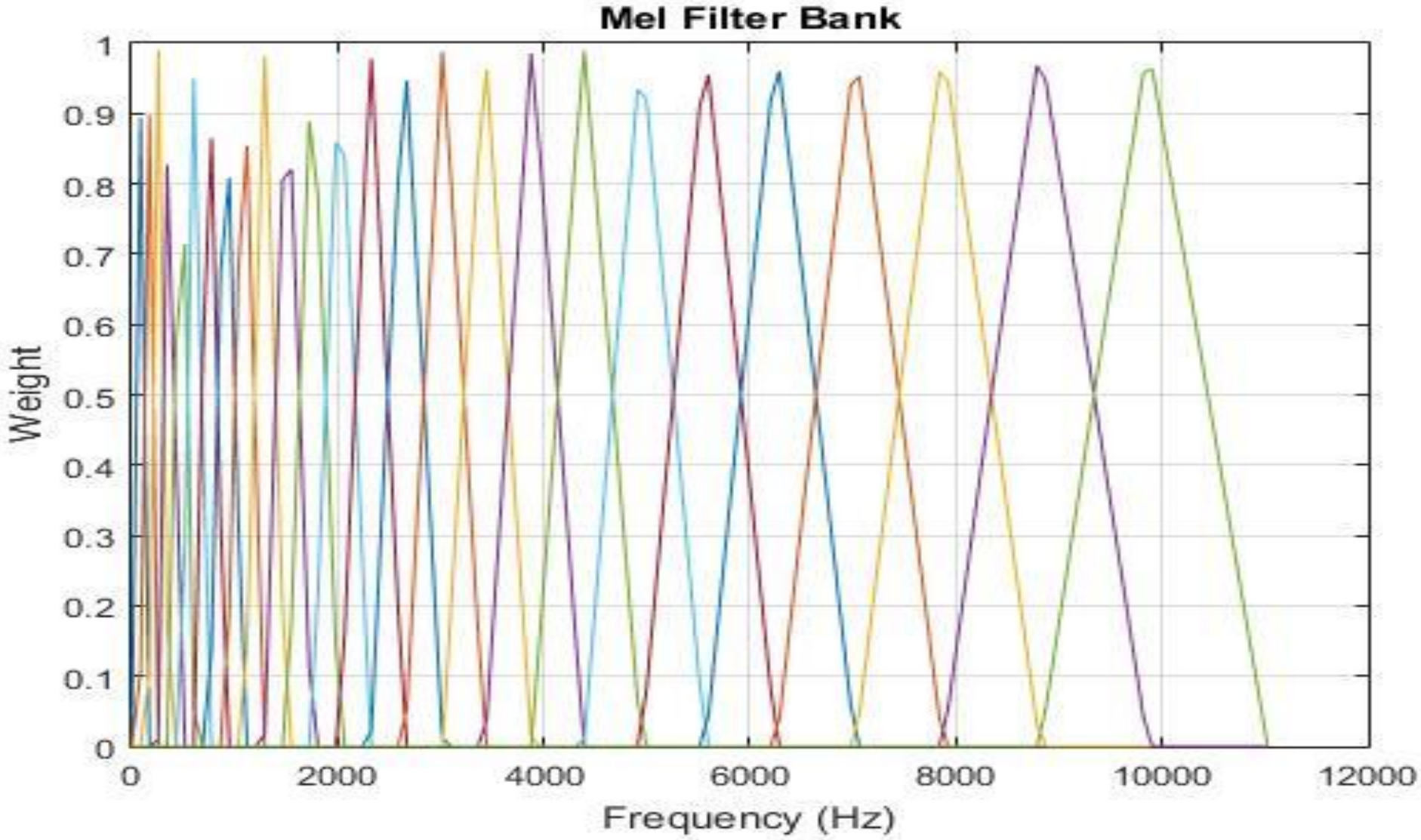


Signal → Mel Frequency Cepstral Coeff (MFCC)

- Pre-emphasize signal and convert resulting signal to frames (sig2frames).
- Compute Mel Filterbank Weights, H
  - H = [M x K], M = # Mel Filters, K = NFFT/2 + 1. Each row of H is a filter.
  - Use [H, f] = melfilterbank(M, K, R, Fs); % M = 26, R = [0 Fs/2].
- Compute Average Magnitude Spectrum for each frame and save in X. (X = [NFFT x Nf])
- Apply H on X to get Y (Mel magnitudes).
  - Make sure to sum the filtered spectrum for each Mel filter. This will give Mel magnitudes.
  - Y = [M x Nf]
- IDCT of log()
  - CC = idct(log(max(1e-6, Y))); % [M x Nf]
- But we only care about the first Ncc = 12 coefficients, so the matrix reduces to Ncc x Nf.
- **Complete the missing parts in mfcc.m**
  - CC = mfcc(signal, Ncc, varargin); % here Ncc = 12, CC = [Ncc x Nf]
  - Example: CC = mfcc(signal, Ncc, 'Nw', Nw, 'No', No, 'M', 26, 'Fs', Fs, 'R', [0 Fs/2]);

- Unroll this matrix (CC) into a single column vector that is (Ncc*Nf)x1.

Mel Filterbank



Overall Procedure

- **Read all audio files** (wavread), and do **imresize** to get uniform lengths.
- **Compute features:**
  - **Cepstrum/MFCC: Apply a sliding window**, **overlap 10%,** and **compute the cepstrum/MFCC** for each frame. Consider different window sizes 100, 500, 10000.
  - Raw Features: One long vector of audio samples from the entire wav file.
- Put all the **cepstra/MFCC/raw features** for each file's frame into a **single feature vector.**
- **For each vector, perform 1-NN and 5-NN using leave-N-out strategy**:
  - *Remove all data corresponding to the test speaker when doing **speech** recognition.*
  - *Remove all data corresponding to the test digit when doing **speaker** recognition*

Overall Procedure (leave-N-out)

- Speech Recognition: (Speaker Independent)
  - Test = A1a.wav. Then, Test Spk = A, Test Digit = 1
  - Train = {B*, C*, D*}. Thus, exclude all A* files (N = 25).
  - Run k-NN. If digit classified by k-NN = 1, then no error. Else, there is an error.
  - Repeat these steps treating every wav file as test datum.
- Speaker Recognition: (Text Independent)
  - Test = A1a.wav. Then, Test Spk = A, Test Digit = 1
  - Train = {A[2-5]*, B[2-5]*, C[2-5]*, D[2-5]*}. Thus, exclude all *1* files (N = 20).
  - Run k-NN. If spk classified by k-NN = A, then no error. Else, there is an error.
  - Repeat these steps treating every wav file as test datum.

Results (Matlab Output)

```
> datadir = '/ws/ece417/hw3/speechdata'
> run(datadir)
```

Cepstrum

Speech Recognition

1-NN

	Raw	W =100	W =500	W =10000
D1:	0	70	65	20
D2:	30	45	50	70
D3:	15	70	65	70
D4:	50	70	65	65
D5:	0	60	70	30
Avg:	19	63	63	51

MFCC

Speech Recognition

1-NN

	Raw	W =100	W =500	W =10000
D1:	0	55	60	35
D2:	30	65	90	90
D3:	15	95	100	90
D4:	50	75	75	60
D5:	0	80	90	50
Avg:	19	74	83	65

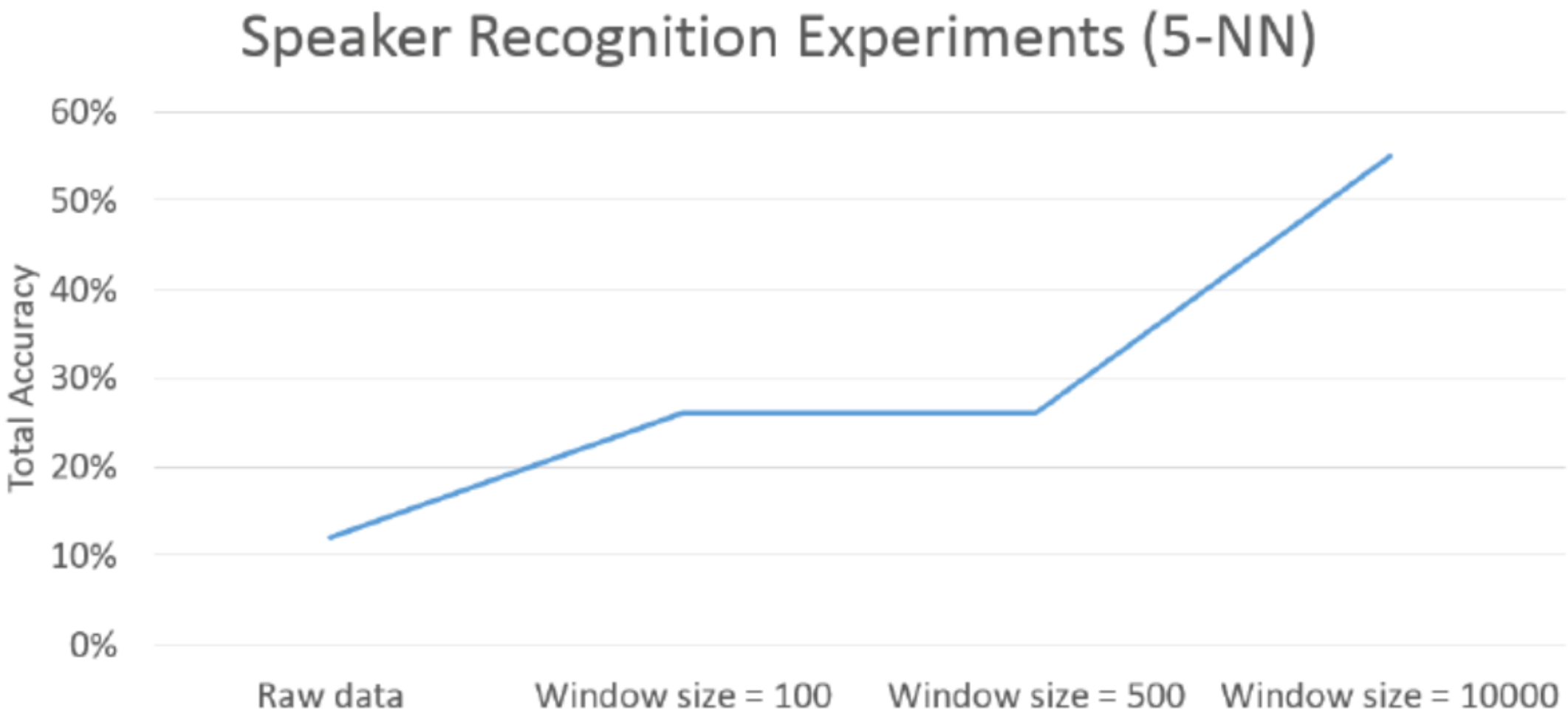
Results (Tables)

- Tables for the 1NN and 5NN results for each speaker, digit, and avg accuracy in both cases.
- Run this for the raw features and cepstrum/MFCC (3 window sizes (100,500,10000)).
- Tabulate results in your write-up.

Results (Graphs)

- Graphs for
  - Speech Recognition Accuracy vs. Window Length for Cepstrum/MFCC Using 1-NN (avg results sufficient; individual digits not reqd)
  - Speech Recognition Accuracy vs. Window Length for Cepstrum/MFCC using 5-NN (avg results sufficient; individual digits not reqd)
  - Repeat the above for Speaker Recognition with avg. results over speakers instead of digits.

Example (Graphs)



Note: Graph is not representative of real results

Turn In

- Your writeup. Mention the 3 features, k-NN classifier, and your analysis in the discussion. All of your experimental results (in table form).

- Your Matlab code
  - run.m: function run(datadir).
  - sig2frames.m, cepstrum.m, melfilterbank.m, mfcc.m
  - Other *.m dependencies (no restriction on i/p or o/p arguments)
  - run_ec.m: function run_ec(datadir). (Extra Credit part. On a separate file this time)

- Upload to Compass:
  - MP3_<NetID>.zip (Matlab code, members.txt. But no audio/image/video files.)
  - MP3_<NetID>.pdf (write up)

Piazza

- (Continue to) Post questions on Piazza.
  - However, I will not be able to respond to direct emails (related to MP*) sent to my Webmail a/c.

- Keep your questions short (e.g. 1-3 lines) and crisp.
  - E.g. of a good question: "Why are we computing the Mel Filterbank weights only over half spectrum (NFFT/2 + 1 points) instead of full spectrum (NFFT) ?"

  - E.g. of a not-so-good question: "I do not understand MFCC from lecture notes or walkthrough although I have been reading this for the last 3 days. Can you provide some hints?"

  (Note: Questions like these are likely to going to have a negative effect if you are working with a Professor or an internship mentor or a manager at your job.)