MP3: Cepstrum and Mel-Frequency Cepstral Coefficients(MFCC)

Group: Akhil Alapaty & Ojus Deshmukh

I. Introduction

As done in previous MP's, we first found the Raw Pixel feature vector, which contains minimally processed data from the original audio files. Next, we calculated the Cepstrum feature vector for the data. This is done by converting the raw pixel feature vector signal into a vector of frames using the sig2frames function. Next, we took the Inverse Fourier Transform of the logarithm of the frames vector's magnitude spectrum. The Cepstrum feature vector is more effective than the Raw Pixel feature vector because it catches features such as pitches and frequencies in a speech signal, since they are part of the logarithmic representation. The cepstrum is generally characterized as containing information that depicts the rate of change in a signal's separate spectrum bands.

The other feature vector we used was MFCC, or Mel-Frequency Cepstral Coefficients. This is calculated by converting the raw pixel feature vector signal into a vector of frames using the sig2frames function. Next, we took the magnitude of the Fourier Transform of the frames. We then multiplied the resulting signal by a precalculated Hamming window. Lastly, we took the Inverse Discrete Cosine Transform of the resulting signal, and took only the first 12 coefficients because they contained the majority of the signal's energy. MFCC is better than a raw feature vector, because on MFC, the frequency bands are spaced on the mel scale instead of linearly, which allows the program to more accurately model human speech(Wikipedia contributors). Once again, the learning metric used in this MP was the kNN classifier, which is used to compare the images after being processed by each of the above feature vectors. It incorporates a "leave one out" strategy, which refers to 1 image being part of the test dataset, while the other 99 audio files are part of the train dataset. This cycles through all 100 images until accuracies are computed for each of them using the nearest neighbor algorithm with values of 1 and 5 for k.

.

II. Methods

After looking at our results, we clearly realized that our data is significantly off from what is expected. The way we implemented the MP is as follows:We read in the 100 audio files using audio read and simultaneously created a class data matrix, which allowed us to store all of the files in a matrix that we would later use to facilitate the K-nearest neighbor algorithm. Next, after finding the left channel of the data, which was of size 10000x100, we called the sig2frames function, which converted the signal into frames. We determined the starting and ending indices of each frame based off the specifications given in the walkthrough. Next, we called the cepstrum function, which as discussed before, performs the inverse Fourier Transform on the log of

the magnitude spectrum. Next, we called the mfcc function in order to get the filtered spectrum. In each of the above sections, we only cared about the top 12 coefficients, so we ended up with 12x100 matrices as we left the function. After returning back to mp3.m, we performed the speech and speaker recognition experiments on each of the data samples, through the use of the K-nearest neighbor distance metric.

The two critical transformations we used in this MP were in calculating the cepstrum and calculating the Mel-Frequency Cepstral Coefficients. In Lines 28-65 of mp3.m, we have three for loops, depending on the Nw value, that run through cepstrum.m and mfcc.m for each .wav file's signal.

The cepstrum is calculated entirely through a cepstrum.m function, which does the following calculations:

$$c[n] = \mathcal{F}^{-1}\{\log|\mathcal{F}\{x[n]\}|\}$$

For each window, the cepstrum is mathematically calculated as such:

$$c[n] = \sum_{n=0}^{N-1} \log\left(\left|\sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi}{N}kn}\right|\right) e^{j\frac{2\pi}{N}kn}$$

The MFCC is calculated entirely in the mfcc.m function. It is mathematically calculated as such:

$$c[n] = \sum_{m=0}^{M-1} C_m cos(\frac{\pi(m+0.5)n}{M})$$

* This equation image was taken from Sheng-Ru Cheng's MP3 Lab Report for ECE 417

III. Results

Cepstrum
Speaker Recognition
1-NN

	Raw	W=100	W=500	W=10000
S1:	50.000000	12.500000	12.500000	12.500000
S2:	50.000000	12.500000	12.500000	12.500000
S3:	50.000000	12.500000	12.500000	12.500000
S4:	75.000000	12.500000	12.500000	12.500000
Avg:	56.250000	12.500000	12.500000	12.500000

MFCC
Speaker Recognition
1-NN

	Raw	W=100	W=500	W=10000
S1:	50.000000	50.000000	50.000000	50.000000
S2:	50.000000	50.000000	50.000000	50.000000
S3:	50.000000	50.000000	50.000000	50.000000
S4:	75.000000	75.000000	75.000000	75.000000
Avg:	56.250000	56.250000	56.250000	56.250000

Cepstrum
Speaker Recognition
5-NN

	Raw	W=100	W=500	W=10000
S1:	175.000000	27.500000	27.500000	27.500000
S2:	175.000000	27.500000	27.500000	27.500000
S3:	200.000000	27.500000	27.500000	27.500000
S4:	200.000000	32.500000	32.500000	32.500000
Avg:	187.500000	29.500000	29.500000	29.500000

MFCC
Speaker Recognition
5-NN

	Raw	W=100	W=500	W=10000
S1:	175.000000	175.000000	175.000000	175.000000
S2:	175.000000	175.000000	175.000000	175.000000
S3:	200.000000	200.000000	200.000000	200.000000
S4:	200.000000	200.000000	200.000000	200.000000
Avg:	187.500000	187.500000	187.500000	187.500000

Cepstrum
Speech Recognition
1-NN

	Raw	W=100	W=500	W=10000
D1:	150.000000	12.500000	12.500000	12.500000
D2:	150.000000	12.500000	12.500000	12.500000
D3:	150.000000	12.500000	12.500000	12.500000
D4:	250.000000	12.500000	12.500000	12.500000
D5:	250.000000	12.500000	12.500000	12.500000
Avg:	190.000000	12.500000	12.500000	12.500000

MFCC
Speech Recognition
1-NN

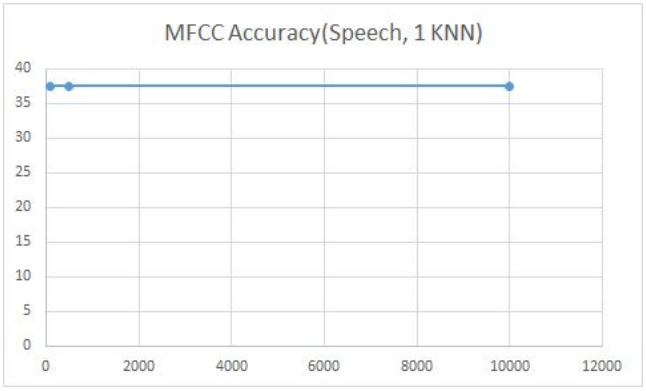
	Raw	W=100	W=500	W=10000
D1:	150.000000	50.000000	50.000000	50.000000
D2:	150.000000	50.000000	50.000000	50.000000
D3:	150.000000	50.000000	50.000000	50.000000
D4:	250.000000	75.000000	75.000000	75.000000
D5:	250.000000	0.000000	0.000000	0.000000
Avg:	190.000000	37.500000	37.500000	37.500000

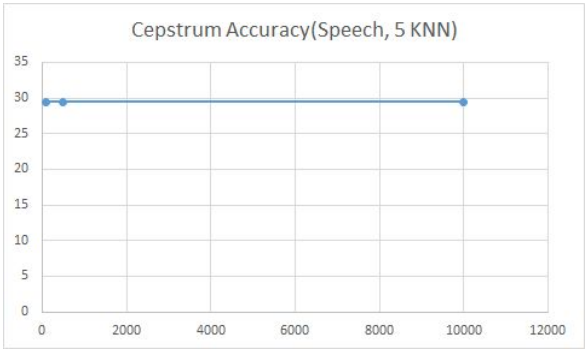
Cepstrum
Speech Recognition
5-NN

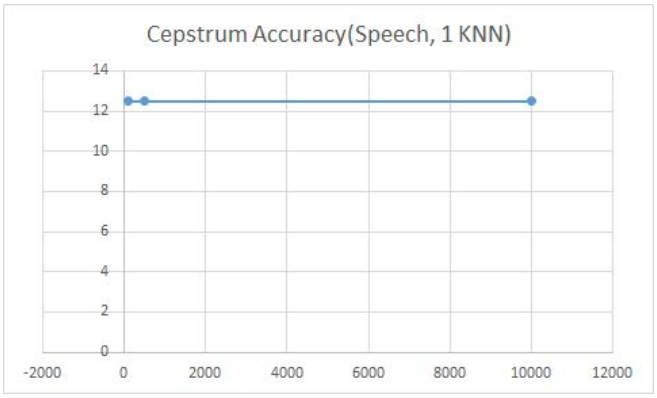
	Raw	W=100	W=500	W=10000
D1:	550.000000	27.500000	27.500000	27.500000
D2:	650.000000	27.500000	27.500000	27.500000
D3:	650.000000	27.500000	27.500000	27.500000
D4:	650.000000	32.500000	32.500000	32.500000
D5:	650.000000	32.500000	32.500000	32.500000
Avg:	630.000000	29.500000	29.500000	29.500000

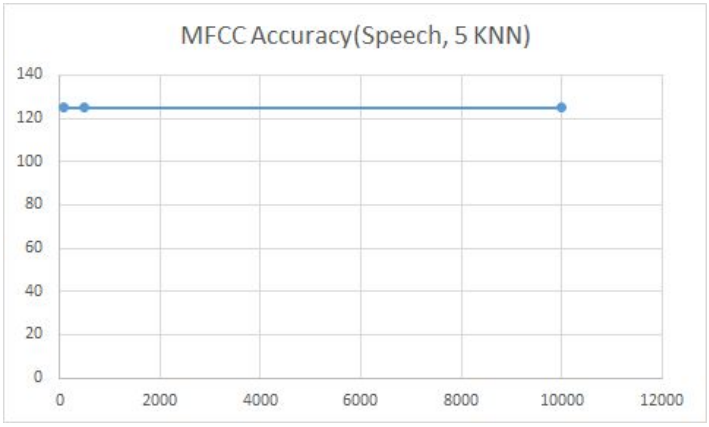
MFCC
Speech Recognition
5-NN

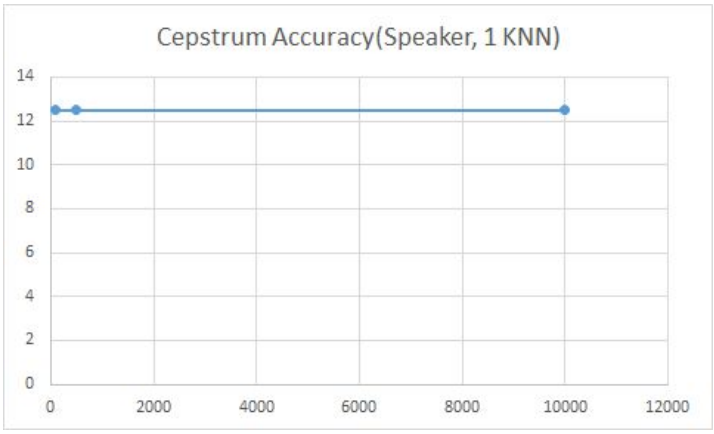
	Raw	W=100	W=500	W=10000
D1:	550.000000	175.000000	175.000000	175.000000
D2:	650.000000	175.000000	175.000000	175.000000
D3:	650.000000	200.000000	200.000000	200.000000
D4:	650.000000	200.000000	200.000000	200.000000
D5:	650.000000	0.000000	0.000000	0.000000
Avg:	630.000000	125.000000	125.000000	125.000000

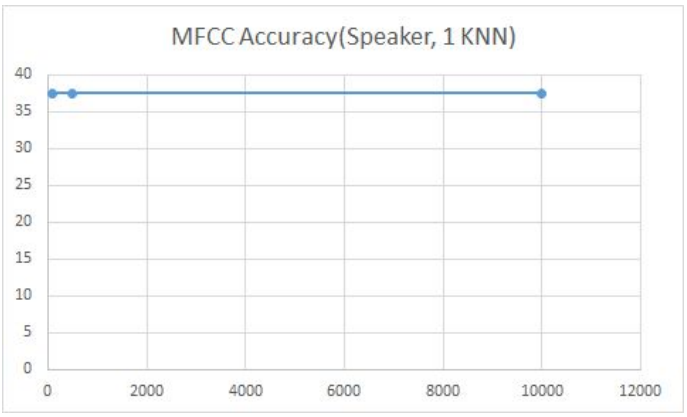


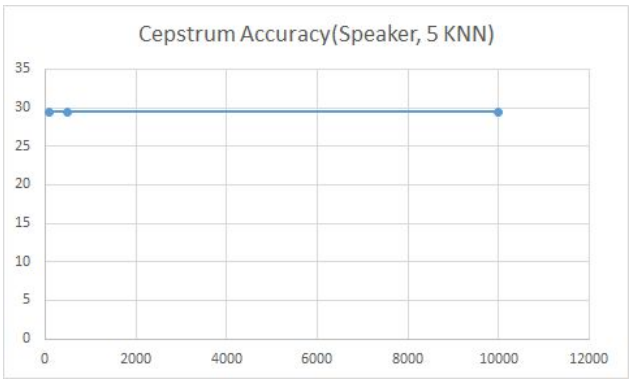


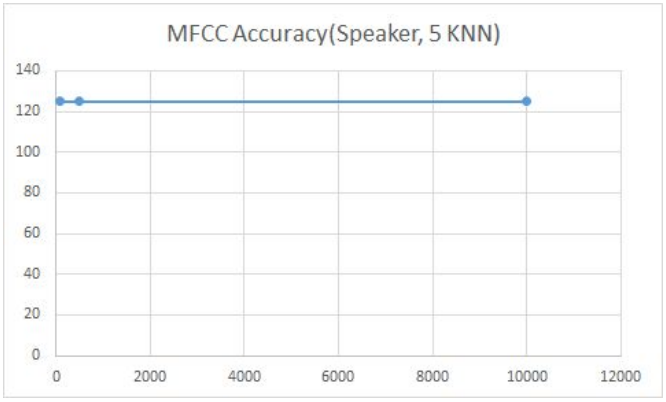












IV. Discussion

As seen above, these completely straight line graphs indicate that there's something missing somewhere in our code, since we did the correct general implementation of the feature vectors and the kNN classifier. Since our data was off, we didn't see any point in analyzing it for the Discussion. Instead, we analyzed the results posted on the Walkthrough and tried to make sense of it conceptually. One thing that stood out was the average for a smaller window was higher than that of a larger window size (63% average for Window Size of 100 vs 51% average for Window size of 10000). Going back to digital signal processing, smaller window sizes meant more samples, which would obviously give a higher accuracy rate since there would be more data to compare and reconstruct.

Wikipedia contributors. "Mel-frequency cepstrum." *Wikipedia, The Free Encyclopedia*. Wikipedia, The Free Encyclopedia, 4 Sep. 2017. Web.
10 Oct. 2017