EE 113DA: Digital Signal Processing Design

Mini-Project #2

Vowel Recognition

Name: Khyle Calpe and Philip Kwan

Lab Section: 1B

OBJECTIVE

The objective of this mini-project is to train a neural network in MatLab to recognize spoken vowels based on the associated Mel Frequency Cepstral Coefficients (MFCCs) obtained through feature extraction on the H7 board.

SYSTEM BLOCK DIAGRAM

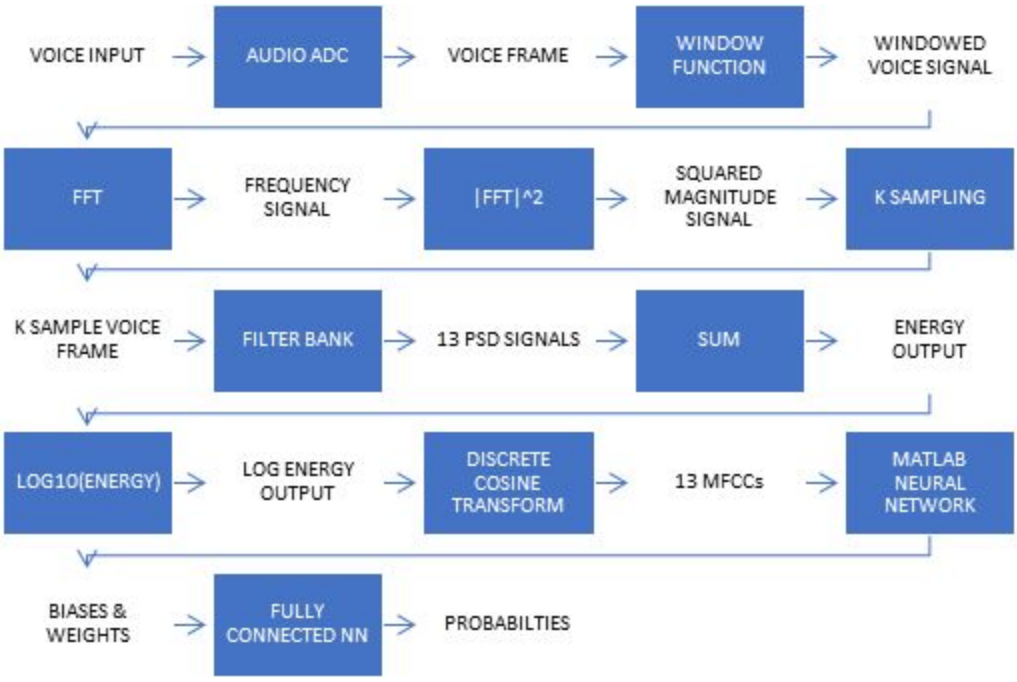

Figure 1. System block diagram of the vowel recognition project.

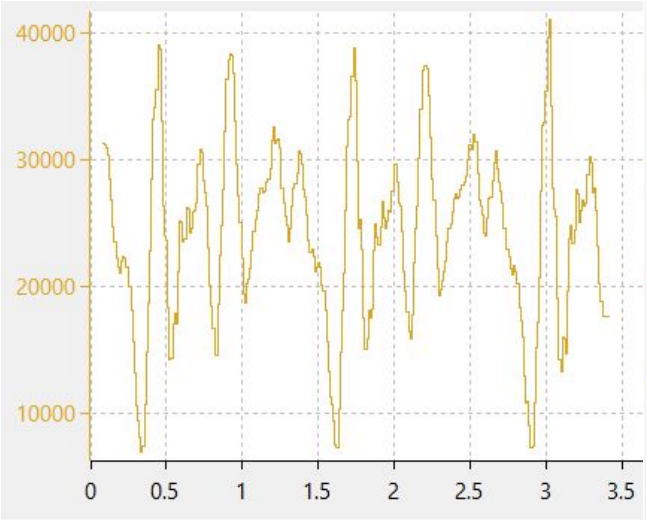

Figure 2. “Ah” Voice Print

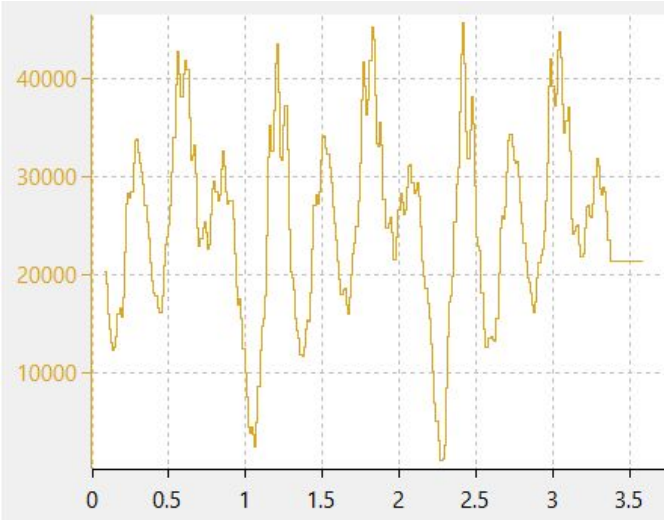

Figure 3. “Eh” Voice Print

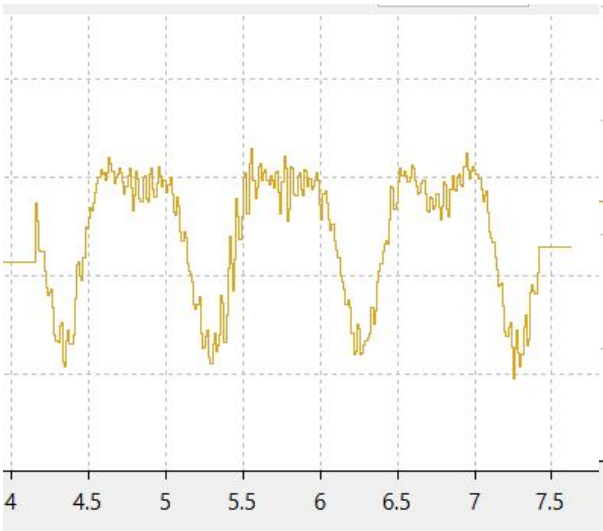

Figure 4. “Ee” Voice Print

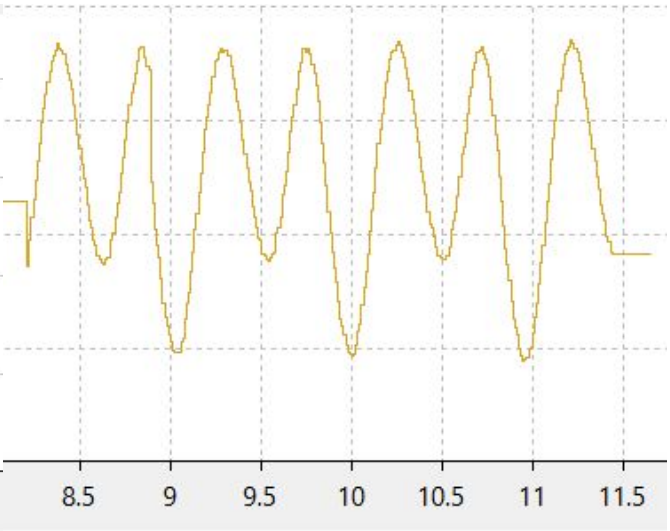

Figure 5. “Oo” Voice Print

DESIGN

The design of the vowel recognition system is based on the computation of the MFCCs of voice samples, which are then fed into a neural network for recognition. The first step of the feature extraction process is to subtract by the DC offset of the audio signal and then filter out the noise from the audio recording via a Hamming window. Then, 513 samples of the squared magnitude of the FFT are fed into 13 triangular bandpass filters. The final step of the feature extraction is to apply a Discrete Cosine Transform based on the logarithmic sums of the energy outputs of each of the 13 signals. 120 audio samples were gathered, with 40 samples from each of our voices, 40 samples from a female voice, and the remaining 40 from voices over a voice call.

We used two fully connected layers for our neural network. For each layer, we created a separate dense function to implement the required activation function since C does not allow directly passing in functions as a function parameter. The input layer has 5 nodes and the output layer has 4 nodes, corresponding with the number of sounds we are training for. The neural network weights, biases, and pre-processing parameters were directly stored in arrays.

In order to speed up the code, care was taken to reduce the number of loops used. When possible, we combined operations done on an array into the same loop. When testing multiple voices in a single session, arrays would need to be reset on every iteration. Thus, most arrays and variables were defined in the loop, so they would be reset on the following run.

DATA

	Ah	Eh	Ee	Oo
Validation Accuracy (%)	100	100	100	100
Test Accuracy (%)	100	100	100	100
New Voice Accuracy (%)	100	100	100	100

Figure 6. Neural Network Performance

RESULTS

The results of the mini-project satisfied all expectations of accuracy for each of the four monophthongs regardless of the gender of the person speaking into the microphone. However, when pronunciation of the sound is off, the accuracy of the neural network degrades. Furthermore, higher pitch EEs over a voice call were not predicted as accurately as the other sounds. We can improve the neural network performance by diversifying our training set to include slightly mispronounced sounds and more female voices over voice calls.

CHALLENGES

The main challenge of this project was from setting up the MFCC code properly. Initially, we were outputting NaN in our values. We later found that this was due to variable type errors. Our recorded signal was stored as an unsigned integer data type but calculations were done assuming signed floating point data types. Thus, when dividing or subtracting, the output values would be treated as 0. This was remedied by typecasting when doing calculations on unsigned integers and storing the output as a float.

Our second issue was getting a large value in the first MFCC term. This error was caused by two things. The first was that we initially did not subtract the DC offset from the voice recording. When done directly on the array, half of the array would go to 0 due to unsigned integers being unable to store negative numbers. We fixed this by doing the DC offset when performing operations where the output would be stored in a float. The second issue was not having correct indices for the calculation using the discrete cosine transform. Due to coding practices, we were indexing starting from 0 while the calculation expected indexing starting from 1. We fixed this by using an offset when using the index as a variable in the calculation.

CONCLUSION

Neural networks are powerful tools that can perform a variety of tasks such as vowel recognition. For simple, periodic sounds, the neural network had good accuracy in predicting the sound. Accuracy can be improved by diversifying the training set with more variety of voices. Using this technique, speech recognition can be implemented, such as the auto-generated captions found in Youtube videos.

APPENDIX

Ah		Eh		Ee		Oo	
1	2	1	2	1	2	1	2
11.382	11.273	11.979	11.876	12.460	12.415	12.747	11.445
11.209	11.119	12.060	11.555	11.877	11.898	11.063	11.904
11.265	11.131	12.599	12.387	10.423	10.475	10.826	10.102
11.829	11.696	12.684	12.350	9.025	9.298	10.981	10.425
12.224	12.242	10.724	11.564	8.565	8.727	10.718	9.893
12.284	12.108	9.570	10.463	8.590	8.906	9.930	9.831
12.471	12.170	10.409	11.030	8.520	8.662	9.585	9.041
12.349	12.052	10.084	10.601	8.723	8.611	9.354	8.445
10.650	10.187	9.961	10.334	8.680	8.480	8.841	8.594
9.853	9.611	10.167	10.283	8.402	8.369	8.861	8.506
9.413	9.357	10.375	10.716	8.472	8.557	8.831	8.160
9.287	9.298	10.954	11.470	8.547	8.544	8.287	7.991
9.499	9.472	10.935	10.501	8.890	8.588	7.862	8.124
9.731	9.369	10.571	10.572	9.119	8.927	8.070	8.247
8.927	8.595	10.414	10.634	9.996	10.420	8.308	8.379
9.440	9.667	10.683	10.461	9.619	10.136	8.279	8.172
9.806	10.013	10.648	10.329	9.839	10.110	8.324	8.361
9.832	10.201	10.517	10.629	10.424	11.277	8.570	8.161
10.155	10.258	10.104	10.016	10.654	10.707	8.036	8.127
9.855	9.835	8.713	8.475	10.239	10.016	7.864	8.339
9.733	9.697	9.003	8.485	8.675	8.645	7.915	8.524
8.794	8.794	8.692	9.189	9.210	8.608	8.252	8.851
8.321	8.931	8.951	9.303	9.704	9.340	8.181	8.819
8.907	8.730	8.918	9.087	9.290	9.189	8.350	9.310
9.078	8.910	9.139	9.561	9.318	9.172	8.497	9.029
9.048	9.092	8.860	9.478	9.401	9.261	8.489	8.452

Figure 7. Log Energy Vectors for Each Monophthong

Ah		Eh		Ee		Oo	
1	2	1	2	1	2	1	2
15.335	14.944	11.686	8.038	0.760	0.754	9.745	7.820
4.092	0.304	-2.165	-2.407	4.761	5.723	11.343	11.292
-2.621	0.700	7.288	9.709	11.964	13.049	8.761	9.040
-1.182	-4.800	9.666	9.681	5.443	4.398	4.403	2.685
-2.082	-5.981	-1.047	-3.378	0.430	2.071	0.541	2.474
0.548	2.554	-3.679	-3.441	3.624	5.070	-0.327	-0.518
-2.165	0.112	-3.791	-2.682	0.804	3.230	-0.230	0.218
-0.636	-0.099	-2.694	-1.812	2.254	2.046	-0.908	0.275
-0.886	1.332	1.655	1.060	2.408	1.711	0.695	0.239
-2.181	-2.146	-2.200	-2.746	-1.267	-1.214	-0.283	2.102
-0.610	0.515	-2.458	-2.929	3.837	3.098	1.151	1.211
0.135	1.667	-0.717	0.888	0.811	0.089	1.390	0.275
1.424	3.020	-0.684	-0.231	-1.145	-0.846	1.628	0.615

Figure 8. MFCC Vectors for each Monophthong