CA4: Spoken digit Recognition

Shruti Hiray

Roll no: 14D070016

1) End Pointing

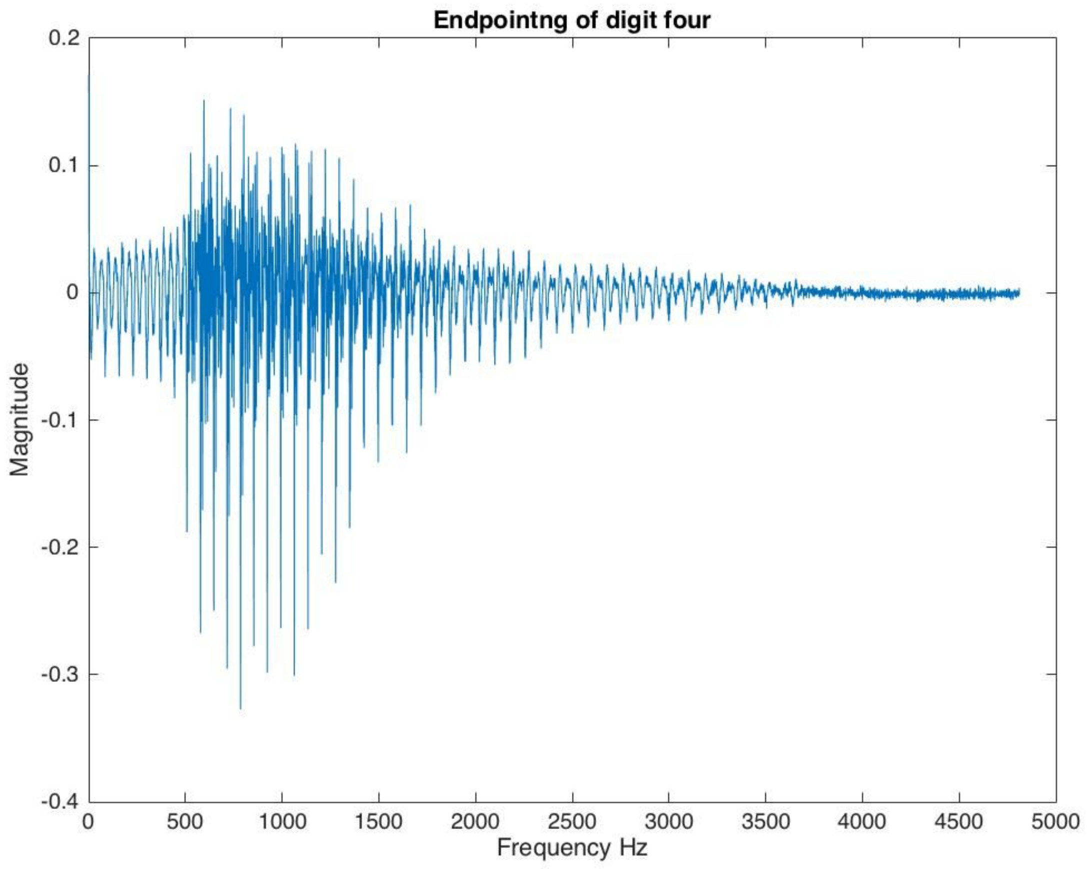
End pointing is done for the sounds to remove the silence time before and after the utterance. Algorithm used for endpointing is as follows:

First, zero crossing rate and energy are calculated at a point using a 10 ms window.
IZCT: Zero crossing rate threshold is calculated using the initial silence before utterance.
IZCT = min(IZC, mean(IZ) + 2*std(IZ))
Where IZC is threshold = 25
Mean (IZ) = mean of zero crossing rate over silence
std(IZ) = standard deviation of zero crossing rate over silence

Energy thresholds are calculated as follows:

Peak energy: IMX
Silence energy: IMN
IMX = max(energy_norm);
I1 = 0.03*(IMX - IMN) + IMN;
I2 = 4*IMN;
ITL = min (I1,I2); (Low energy threshold)
ITH = 5*ITL; (High energy threshold)

N1 be the begin point & N2 be the end point.



To find N1:

1) First point A is calculated from the beginning of signal where it crosses the ITL.
2) Now it is checked that after point A, energy does not go below ITL till it crosses ITH. Then point A is the required N1.
3) If energy falls below ITL, before it reaches ITH then this point is the new point A. Step 2 is repeated until N1 is found.

To find N2:

1) First point A is calculated from the end of signal where it crosses the ITL.
2) Now it is checked that after point A, energy does not go below ITL till it crosses ITH. Then point A is the required N1.
3) If energy falls below ITL, before it reaches ITH then this point is the new point A. Step 2 is repeated until N1 is found.

Also after calculating N1,

A 250 ms interval before N1 is examined, counts the number of intervals where the zero crossing rate exceeds the threshold IZCT. If the number of times the threshold exceed is 3 or more times then N1 is set as first point where threshold exceeded. Otherwise it is kept as N1.

Similar test is used for N2. If there is unvoiced energy in the interval of 250 ms after N2, the endpoint is readjusted using the zero crossing rate.

Matlab Code

1) For calculating threshold of zero crossing rate: IZCT

```
zcr_silence = zeros(1,3800);
for i = 1 : 40
    for n=(-i+1):40
        if normalized_signal(i+n) == 0
            zcr_silence(i) = zcr_silence(i) + 1;
        end

    end
end
for i = 41 : 3800
    for n=-40:40
        if normalized_signal(i+n) == 0
            zcr_silence(i) = zcr_silence(i) + 1;
        end

    end
end
zcr_sil_mean = mean(zcr_silence);
zcr_sil_dev = std(zcr_silence);

IF = 25; % fixed threshold
IZCT = min(IF,zcr_sil_mean + 2* zcr_sil_dev);
```

2) Calculating N1

```
IMX = max(energy_norm);
I1 = 0.03*(IMX - IMN) + IMN;
I2 = 4*IMN;
ITL = min (I1,I2);
ITH = min(5*ITL);

for m=1:divider
    if energy_norm(m) >= ITL
        i=m+1;
        while energy_norm(i)<ITH
            if energy_norm(i) < ITL
                m=i+1;
                break;
            end
            i=i+1;

        end
        if energy_norm(i) >= ITH
                N1 = m-1;
                break;
        end

    end
end
```

3) Calculating N2 and storing pre-emphasized endpointed signal

```
for m=divider:-1:N1
    if energy_norm(m) >= ITL
        i=m-1;
        while energy_norm(i)<ITH
            if energy_norm(i) < ITL
                m=i-1;
                break;
            end
            i=i-1;

        end
        if energy_norm(i) >= ITH
                N2 = m+1;
                break;
        end

    end

end
clipped_sample = sample(N1:N2);
pre_clipped = filter(A,1,clipped_sample);
audiowrite(filename_Vedhas2(k,:),pre_clipped,fs);
```

2) **Feature extractor**

Computing MFCC feature vector for every 10ms of utterance

A total of 39 features are extracted for every frame which comprise of
12 cepstral coefficients
12 delta coefficients
12 delta delta coefficients
1 energy coefficient
1 delta energy coefficient
1 delta delta energy coefficient

This is done as follows:

1) Coefficients are calculated for a frame size of 25ms and with hop of 10ms.
2) Discrete fourier transform is taken of the sample windowed with hamming window. Periodogram based estimate is calculated using DFT .
   $P = |S|^2/N$ where S is the DFT and P is periodogram estimate
3) Mel - spaced filterbank is calculated using 26 filters.
   The upper 4000 Hz and lower frequencies 300 Hz are converted to Mel frequencies using the conversion equation

$$M = 1125\ (1 + \ln(f)/700\ )$$

The corresponding Mel frequencies are divided into equal spaces by adding 26 additional points in between.

These are converted back to Hertz using the equation

$M^{-1} = 700(\exp(m/1125)-1)$

Frequencies are converted to bin numbers f(i) using fft points (512) and sampling frequency (8kHz)

4) Filterbanks are calculated at these points.

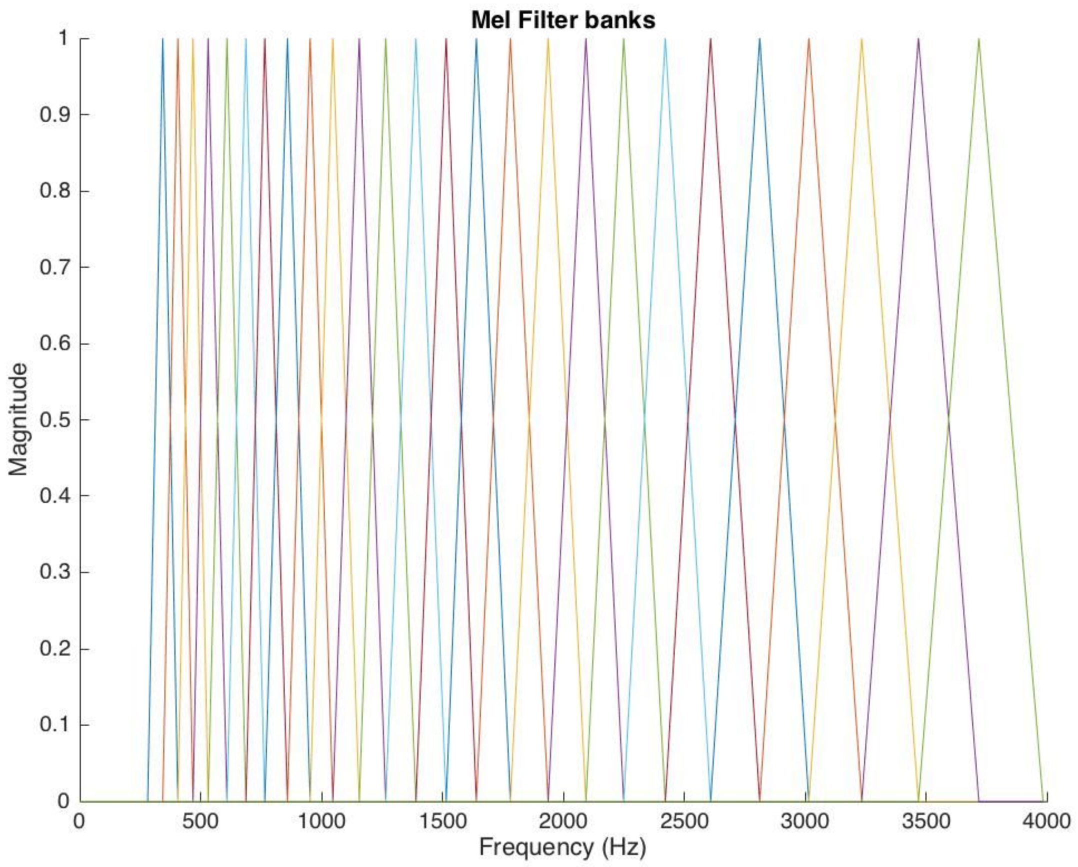
5) Filterbanks are each multiplied with the power spectrum to get the filterbank energy. The log of these 26 filterbank energies gives log filter energy.

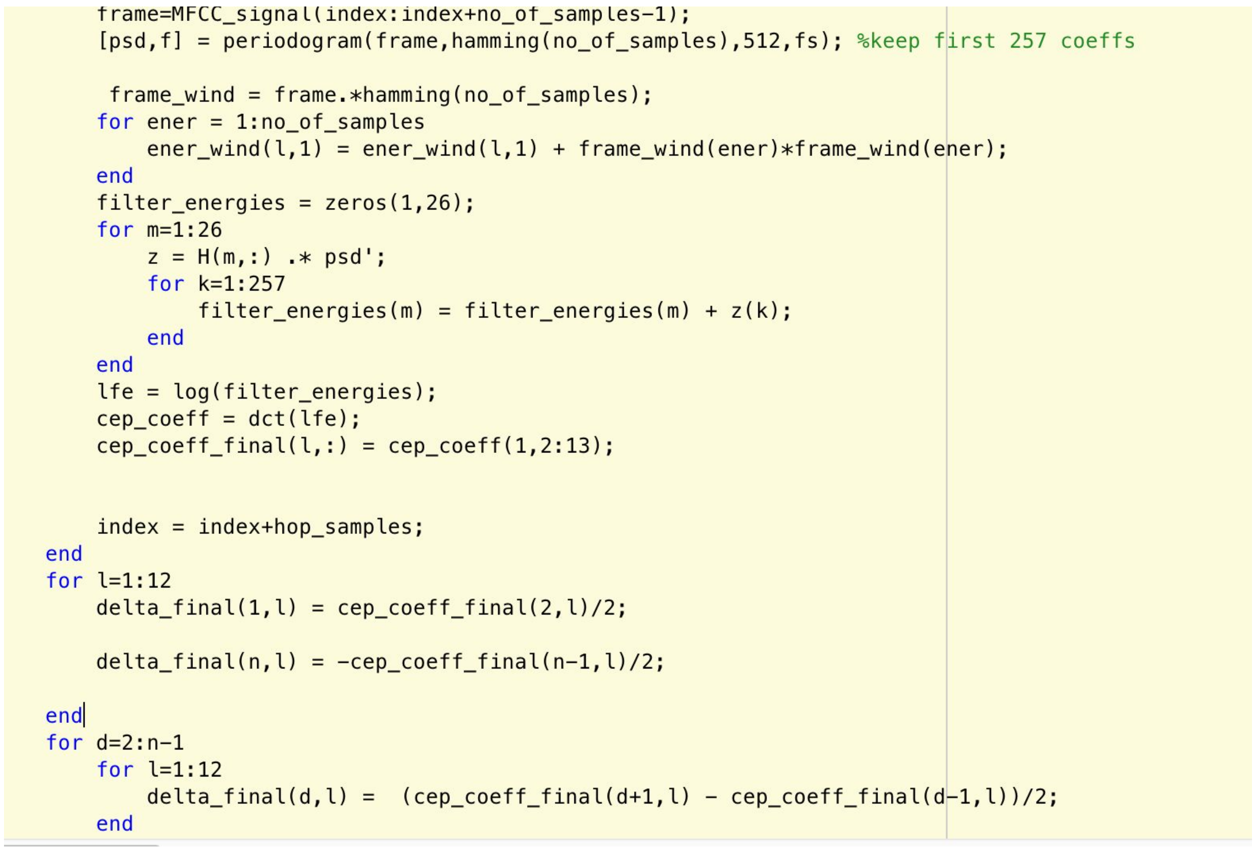
6) Discrete cosine transform of the energies give us 26 cepstral coefficients.

7) 2:13 cepstral coefficients are chosen.
8) Delta for spectral dynamics are calculated to get another 12 cepstral coefficients
$d(t) = (c(t+1) - c(t-1))/2$
9) Similarly delta delta coefficients are calculated from delta coefficients
$dd(t) = (d(t+1) - d(t-1))/2$
Delta , delta delta coefficients helps to capture features across frames.

10) Energy of frame, energy of delta, energy of delta delta are 3 additional features.

11) Thus total 39 features are calculated.
12) For validation without using the spectral dynamics 13 features are kept (12 cepstral coefficients + 1 energy coefficient).


Mel Filter banks
Magnitude
Frequency (Hz)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
500
1000
1500
2000
2500
3000
3500
4000



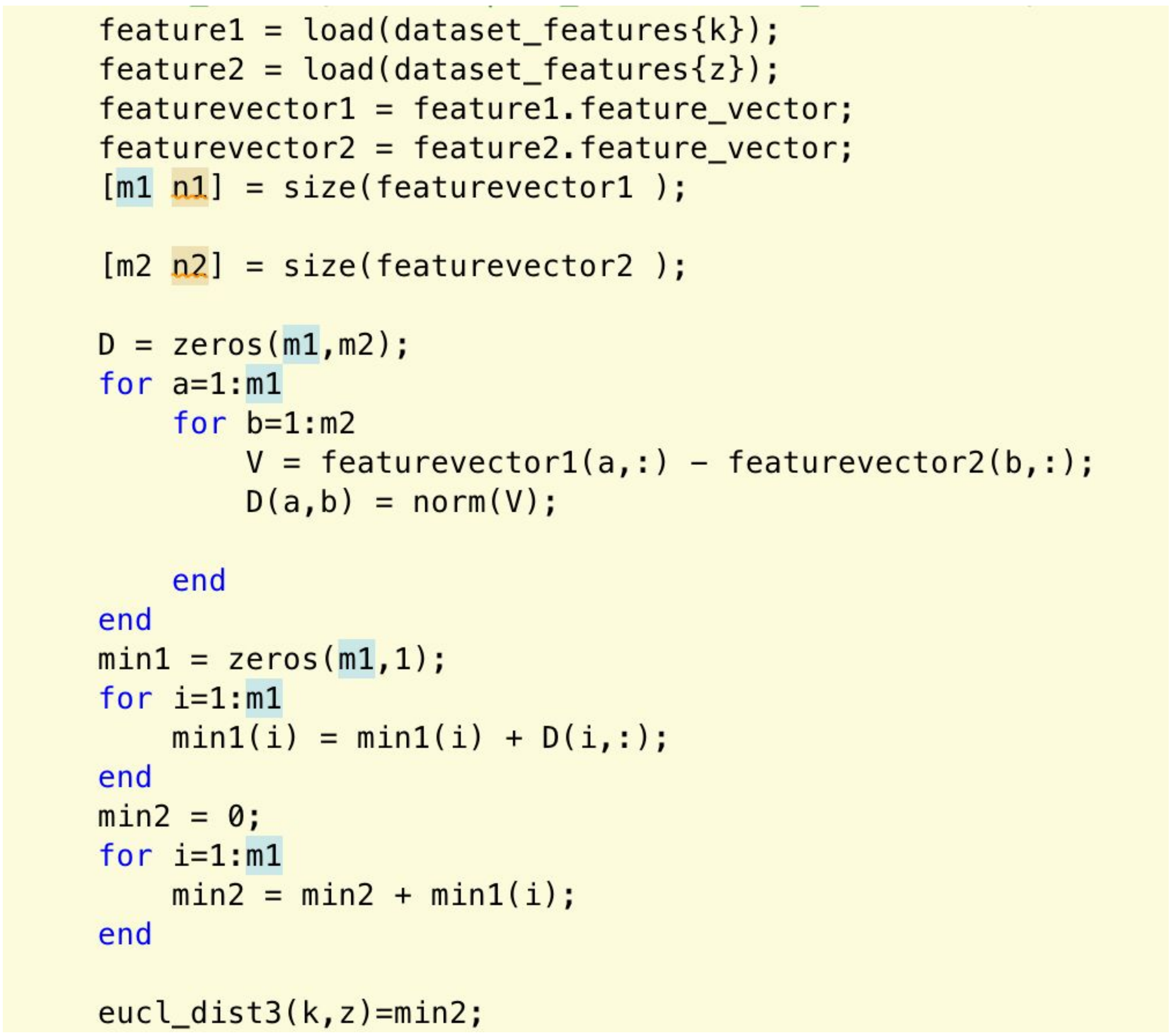
```
        frame=MFCC_signal(index:index+no_of_samples-1);
        [psd,f] = periodogram(frame,hamming(no_of_samples),512,fs); %keep first 257 coeffs

         frame_wind = frame.*hamming(no_of_samples);
        for ener = 1:no_of_samples
            ener_wind(l,1) = ener_wind(l,1) + frame_wind(ener)*frame_wind(ener);
        end
        filter_energies = zeros(1,26);
        for m=1:26
            z = H(m,:) .* psd';
            for k=1:257
                filter_energies(m) = filter_energies(m) + z(k);
            end
        end
        lfe = log(filter_energies);
        cep_coeff = dct(lfe);
        cep_coeff_final(l,:) = cep_coeff(1,2:13);


        index = index+hop_samples;
    end
    for l=1:12
        delta_final(1,l) = cep_coeff_final(2,l)/2;

        delta_final(n,l) = -cep_coeff_final(n-1,l)/2;

    end
    for d=2:n-1
        for l=1:12
            delta_final(d,l) =  (cep_coeff_final(d+1,l) - cep_coeff_final(d-1,l))/2;
        end
```

3) Bag of frames N-fold CV testing

1) Training vectors directly

Each frame has 39 features. If a signal has n frames it has **n** feature vectors each has a dimension of 39. There are **64** samples of a digit (**16** no of speakers x **4** utterances by a speaker).
Thus a typical bag of frames has **64n** feature vectors of dimension 39 (13 for without spectral dynamics) for training.

The testing signal is cross validated with all the bag of frames (feature vectors of testing signal are not included in the bag of frames). The bag of frames for that digit which gives the minimum euclidean distance for the testing signal is the required recognized digit. The word error rate (WER) is as follows:



```
    feature1 = load(dataset_features{k});
    feature2 = load(dataset_features{z});
    featurevector1 = feature1.feature_vector;
    featurevector2 = feature2.feature_vector;
    [m1 n1] = size(featurevector1 );

    [m2 n2] = size(featurevector2 );

    D = zeros(m1,m2);
    for a=1:m1
        for b=1:m2
            V = featurevector1(a,:) - featurevector2(b,:);
            D(a,b) = norm(V);

        end
    end
    min1 = zeros(m1,1);
    for i=1:m1
        min1(i) = min1(i) + D(i,:);
    end
    min2 = 0;
    for i=1:m1
        min2 = min2 + min1(i);
    end

    eucl_dist3(k,z)=min2;
```

WER % = 19.26 (with spectral dynamics)
WER % = 23.12 (without spectral dynamics)
WER% = 15.31 (same gender, with spectral dynamics)
WER% = 18.74 (same gender without spectral dynamics)
Output Matrix:

	0	1	2	3	4	5	6	7	8	9
0	62		6	8	3	11	8			
1		64	2		7	4		1		1
2			56		7					
3				51					2	

4					47					
5						44		1		
6						1	47		1	
7								56		1
8				5			4		61	
9	2					5	5	6		62

One is identified correctly all the times.
Zero has most false positives.
Four, five, six, seven has large errors. This is a result of fricatives present in the sound.
Seven and Nine and commonly confused pairs due to similarly sounding endings.

2) VQ with no of clusters = 4

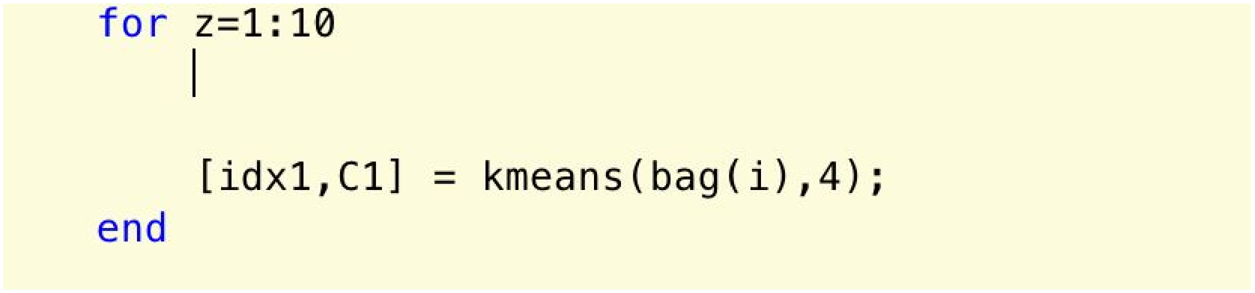
A VQ codebook for each digit is formed by clustering the respective bag of frames of the digit into 4 clusters using k-means clustering. Thus now each feature vector of testing utterance is tested with 4 prototype vectors for a single bag of frames. As before, the bag of frames for that digit which gives the minimum euclidean distance for the testing signal is the required recognized digit. The word error rate (WER) is as follows:

WER % = 23.44 (with spectral dynamics)
WER % = 25.76 (without spectral dynamics)
WER% = 10.29 (same gender, with spectral dynamics)
WER% = 11.74 (same gender without spectral dynamics)

3) VQ with no of clusters = 8

Same implementation as above except there are now 8 prototype vectors for a bag of frames.
The word error rate (WER) is as follows:

WER % = 21.14 (with spectral dynamics)
WER % = 22.67 (without spectral dynamics)
WER% = 9.80 (same gender, with spectral dynamics)
WER% = 11.09 (same gender without spectral dynamics)



```
for z=1:10


    [idx1,C1] = kmeans(bag(i),4);
end
```

This code is followed by distance computation like previous method

	0	1	2	3	4	5	6	7	8	9
0	60		3	7	3	11	8			
1		63	1		9	4		7	7	3
2			55		9					
3	2			51					2	
4					49					
5						44		6		
6						1	47		1	
7								48		1
8			5	5			4		54	
9	2	1				5	5	3		60

One is identified correctly all the times.
Four, five, six, seven has large errors. This is a result of fricatives present in the sound.
Seven and Nine and commonly confused pairs due to similarly sounding endings.
Zero, eight, one have most number of false positives.

Observations:

1) Words with fricatives such as six, five leads to more error as a result of not accurate endpointing.
2) Error with K-clustering is larger than directly training vectors. Each bag of frames has now lesser no of vectors which as a result decreases the accuracy. Benefit of k-means clustering is less computation than training vectors directly.

3) Including spectral dynamics features in the feature vector gives much less error than without using it. This is because spectral dynamic features are across frames thus helps in distinguishing same phoneme occurring in different digits such as /n/ in seven and one, both can have same cepstral coefficients and leads to minimum euclidean distance if spectral dynamics not included.

4) Error with 8 clusters is lesser than error with 4 clusters. Because each testing vector has more vectors for distance computation in a bag of frame leading to more accuracy.

4) Dynamic time warping

DTW is one of the measures of similarity for temporal sequences which can vary in speed.
For a matrix A of k x m dimensions and matrix B of k x n dimension;
The acceptable distance paths from $d_{1,1}$ to end $d_{m,n}$ are:

1) Vertical(i,j) -> (i+1,j)
2) Horizontal (i,j) -> (1,j+1)
3) Diagonal (i,j) -> (i+1,j+1)

At each step the best path is chosen among the three possibilities.

The distance between signals with different number of frames are best calculated using this approach. The word error rate (WER) is as follows:

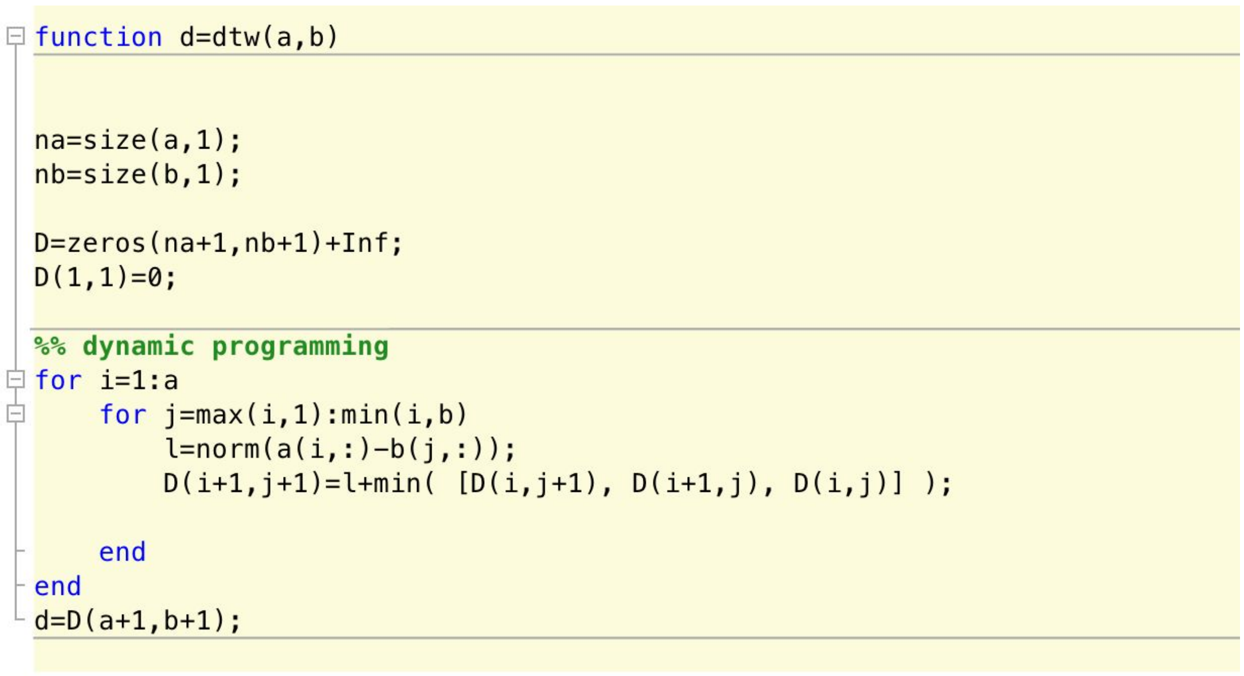
WER % = 11.29 (with spectral dynamics)
WER % = 12.55 (without spectral dynamics)
WER% = 6.43 (same gender, with spectral dynamics)
WER% = 7.12 (same gender without spectral dynamics)

Output matrix:

	0	1	2	3	4	5	6	7	8	9
0	60		1							
1		64				8		1		1
2			62		10					

3				60			3		2	
4					52					
5					1	49				
6	2			2	1	5	57	5	2	
7	2		1					58		1
8				2		1	4		60	
9						1		1		62

Row represent the true value.
Column predicted value.
One is correctly identified all the times.
Accuracy of four, five, six, seven is not good. This is as a result of fricatives present in these sounds.
Six has most number of false positives followed by one.
Nine is predicted as one or seven due to similar sounds at end.



```
function d=dtw(a,b)


na=size(a,1);
nb=size(b,1);

D=zeros(na+1,nb+1)+Inf;
D(1,1)=0;

%% dynamic programming
for i=1:a
    for j=max(i,1):min(i,b)
        l=norm(a(i,:)-b(j,:));
        D(i+1,j+1)=l+min( [D(i,j+1), D(i+1,j), D(i,j)] );

    end
end
d=D(a+1,b+1);
```

5) Conclusion:

Thus dynamic time warping gives the best result, followed by training of vectors directly which is followed by K-means clustering.