Name: Archishman Biswas

Roll no. 180070009

EE 679: Computation Assignment 3

The codes required for the generation of the codebooks required for VQ algorithm is given here

Code for connecting to drive folder (ignore/do not run if not on google colab)

In [9]:

```
from google.colab import drive
drive.mount('/content/gdrive/', force_remount=True)
import os
root_dir = "/content/gdrive/MyDrive/EE 679 Speech Processing Assignments/"
project_folder = "3"

def create_and_set_working_directory(project_folder):
  if os.path.isdir(root_dir + project_folder) == False:
    os.mkdir(root_dir + project_folder)
    print(root_dir + project_folder + ' did not exist but was created.')
  os.chdir(root_dir + project_folder)

create_and_set_working_directory(project_folder)
! pwd
```

```
Mounted at /content/gdrive/
/content/gdrive/MyDrive/EE 679 Speech Processing Assignments/3
```

Importing packages that are required in later sections

In [10]:

```
import numpy as np
import matplotlib.pyplot as plt
from IPython.display import Audio
import scipy.signal as sp
import librosa
import librosa.display
import soundfile as sf
from scipy.signal import find_peaks

def save_as_wav(y_input,file_name,F_samp):
  y_norm = ((y_input-np.min(y_input))/(np.max(y_input)-np.min(y_input))) - 0.5 #making m
ean = 0 and swing = 1
  sf.write(file_name+'.wav',y_norm,F_samp,'PCM_24')

def play_sound(file_name):
  """file_name: the name of the audio file along with the extension"""
  audio = Audio(filename='./'+file_name)
  display(audio)
```

Clustering features to get centroid(forms codebooks)

In [11]:

```
def generating_codebook(feat_vecs,num_centroids,plot_distortions=False):
```

```
def generating_codebook(feat_vecs,num_centroids,plot_distortions=False):
  """feat_vecs: contains all the 39-length feature vectors(as rows) in for a particular u
tterance
  num_centroids: choose between 6 to 64, check the distortion for best match
  Returned values:
  centroids: a num_centroids*39 matrix containing all the estimated centroids from the KM
C algorithm
  sd_vec: a 39 length, capturing the original SD value for all of the features, will be h
elpful in testing part
  """
  from scipy.cluster.vq import vq, kmeans, whiten
  white_feat_vecs = whiten(feat_vecs)

  if plot_distortions:
    distortion = np.zeros(num_centroids)
    for j in range(1,num_centroids+1):
      codebook, distortion_this = kmeans(white_feat_vecs, j)
      distortion[j-1] = distortion_this
    plt.plot(distortion); plt.show()

  codebook, distortion = kmeans(white_feat_vecs, num_centroids)
  sd_vec = np.std(feat_vecs, axis=0) #calculates and returns the sd of each column of dat
a
  #note that, the whitened matrix, each of the column entries feat_vecs is divided by the
sd of same feat_vecs column
  return sd_vec, codebook, distortion #the codebook has the required num_centroids vector
s as rows
```

In [18]:

```
#Reading the .npy files MFCC vectors
all_words_feats_list = np.load("feats_list.npy",allow_pickle=True)
```

In [21]:

```
sd_vec_list = []; codebook_list = [];
for j in range(len(all_words_feats_list)):
  sd_vec,codebook,distortion = generating_codebook(all_words_feats_list[j],128,plot_disto
rtions=False)
  print("Distortion in this case = ", distortion)
  sd_vec_list.append(sd_vec)
  codebook_list.append(codebook)
```

```
Distortion in this case =  4.629619462201386
Distortion in this case =  4.657520945065071
Distortion in this case =  4.722178301139992
Distortion in this case =  4.553997265918201
Distortion in this case =  4.5600498316321385
Distortion in this case =  4.719465298369365
Distortion in this case =  4.6232064974403615
Distortion in this case =  4.583539536643997
Distortion in this case =  4.69763482103644
Distortion in this case =  4.61215263009253
```

In [23]:

```
sd_vec = np.array(sd_vec_list)
codebook = np.array(codebook_list)

np.save("sd_vec_64.npy", sd_vec)
np.save("codebook_128.npy", codebook)
```

In [ ]:

```
```