```
%pip install kapre==0.1.7
%pip install soundfile
%pip install PyPrind
```

```
Collecting kapre==0.1.7
  Downloading https://files.pythonhosted.org/packages/76/c0/0fa10cd05a8368d5bd5381
Requirement already satisfied: numpy>=1.8.0 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: librosa>=0.5 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: tensorflow>=1.15 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: joblib>=0.12 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: decorator>=3.0.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: scikit-learn!=0.19.0,>=0.14.0 in /usr/local/lib/pyt
Requirement already satisfied: six>=1.3 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: scipy>=1.0.0 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: resampy>=0.2.0 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: numba>=0.38.0 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: audioread>=2.0.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: protobuf>=3.9.2 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: astunparse==1.6.3 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: tensorboard<3,>=2.3.0 in /usr/local/lib/python3.6/d
Requirement already satisfied: absl-py>=0.7.0 in /usr/local/lib/python3.6/dist-pac
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.6/dist-p
Requirement already satisfied: tensorflow-estimator<2.4.0,>=2.3.0 in /usr/local/li
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: google-pasta>=0.1.8 in /usr/local/lib/python3.6/dis
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: keras-preprocessing<1.2,>=1.1.1 in /usr/local/lib/py
Requirement already satisfied: grpcio>=1.8.6 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: h5py<2.11.0,>=2.10.0 in /usr/local/lib/python3.6/di
Requirement already satisfied: gast==0.3.3 in /usr/local/lib/python3.6/dist-packag
Requirement already satisfied: setuptools in /usr/local/lib/python3.6/dist-package
Requirement already satisfied: llvmlite<0.32.0,>=0.31.0dev0 in /usr/local/lib/pyth
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-pa
Requirement already satisfied: google-auth<2,>=1.6.3 in /usr/local/lib/python3.6/d
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/pyt
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.6/dis
Requirement already satisfied: importlib-metadata; python_version < "3.8" in /usr/
Requirement already satisfied: rsa<5,>=3.1.4; python_version >= "3" in /usr/local/
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.6/
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.6/d
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packa
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.6/dist
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/loc
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-packages
Requirement already satisfied: pyasn1>=0.1.3 in /usr/local/lib/python3.6/dist-pack
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.6/dist-pa
Building wheels for collected packages: kapre
  Building wheel for kapre (setup.py) ... done
  Created wheel for kapre: filename=kapre-0.1.7-cp36-none-any.whl size=11663 sha25
  Stored in directory: /root/.cache/pip/wheels/80/b0/36/f8ef462364784fcb1b7782e73e
Successfully built kapre
Installing collected packages: kapre
  Found existing installation: kapre 0.1.3.1
    Uninstalling kapre-0.1.3.1:
```

```
      Successfully uninstalled kapre-0.1.3.1
Successfully installed kapre-0.1.7
Collecting soundfile
```

```
import numpy as np
import os
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
from scipy.stats import multivariate_normal as mult_gauss
import librosa as ls
import pyprind
```

Link google drive for file import

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

Two phones chosen are:

1. dha
2. dhu

Compute MFCC and split data into train and test, taking 20% data for testing and rest for training

```
data_dir = '/content/drive/My Drive/SVL-PRASANNA-CV/'   # upload the files in a folder on
os.chdir(data_dir)

def normalize(X):
    temp = X-np.mean(X,axis=0)
    Y = temp/np.std(X,axis=0)
    return Y

dha_mfcc = []
for file in os.listdir('./dha'):
    if file.startswith('dha'):
        path = os.path.join('./dha',file)
        signal,sr = ls.load(path, duration=0.21 ,sr=None)
        #print(ls.get_duration(signal))
        mfccs = ls.feature.mfcc(y=signal, sr=sr, n_mfcc=13, hop_length=int(0.010*sr), n_ff
        dha_mfcc.append(mfccs.T)

dhu_mfcc = []
for file in os.listdir('./dhu'):
    if file.startswith('dhu'):
        path = os.path.join('./dhu',file)
        signal,sr = ls.load(path, duration=0.21, sr=None)
```

```
        signal,sr = ls.load(path, duration=0.21, sr=None)
        mfccs = ls.feature.mfcc(y=signal, sr=sr, n_mfcc=13, hop_length=int(0.010*sr), n_ff
        dhu_mfcc.append(mfccs.T)

# Split data into train and test
dha_mfcc = normalize(np.array(dha_mfcc))
dhu_mfcc = normalize(np.array(dhu_mfcc))
dha_mfcc = np.array(dha_mfcc)
dhu_mfcc = np.array(dhu_mfcc)
#print(dha_mfcc.shape)
#print(dhu_mfcc.shape)
#print(dha_mfcc.shape, dhu_mfcc.shape)
dha_train,dha_test = train_test_split(dha_mfcc, test_size=0.2)      #Separate 2% samples f
dhu_train,dhu_test = train_test_split(dhu_mfcc, test_size=0.2)
print(dha_test.shape, dhu_test.shape)
```

```
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
```

```
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
/usr/local/lib/python3.6/dist-packages/librosa/filters.py:284: UserWarning: Empty
  warnings.warn('Empty filters detected in mel frequency basis. '
```

```
k = 5     #Model size
m_gmm = 3
vect_len = 13


def pdf(x, state, weights):
    wt = weights['w'][state]
    mean = weights['mu'][state]
    var = weights['co_var'][state]

    pdf = 0

    for i in range(m_gmm):
        a = (np.sqrt((np.linalg.det(var[i]) * (2*np.pi)**len(x))))
        b = np.exp((-np.matmul(np.matmul(np.transpose(x-mean[i]) , np.matrix(var[i]).I ),
        pdf = pdf + float(b/a)
    return pdf


def get_alpha(x, weights):
    alpha = np.zeros((x.shape[0], k))

    for j in range(alpha.shape[1]):
        alpha[0][j] = weights['phi'][j] * pdf(x[0],j, weights)

    for i in range(1,alpha.shape[0]):
        for j in range(alpha.shape[1]):
#             print(A[:,j].shape, alpha[i-1].shape)
            alpha[i][j] = np.dot(weights['A'][:,j].reshape(1,-1), alpha[i-1].reshape(-1,1)

    return alpha

def get_beta(x, weights):
    beta = np.zeros((x.shape[0], k))

    for j in range(beta.shape[1]):
        beta[x.shape[0]-1][j] = 1

    for t in reversed(range(0,beta.shape[0]-1)):
        for i in range(beta.shape[1]):
```

```
            temp = 0
            for j in range(k):
                temp += beta[t+1][j]*weights['A'][i][j]*pdf(x[t+1], j, weights)
            beta[t][i] = temp

    return beta


def get_gamma(alpha,beta, weights):
    gamma = []
    for i in range(0,len(alpha)):
        gamma.append((alpha[i]*beta[i])/np.sum(alpha[i]*beta[i]))
    gamma = np.asarray(gamma)

    return gamma

def get_zeta(x, alpha, beta, weights):
    zeta = []
    for t in range(alpha.shape[0]-1):
        temp = np.zeros((k, k))
        for i in range(k):
            for j in range(k):
                temp[i][j] = alpha[t][i]*weights['A'][i][j]*pdf(x[t+1], j, weights)*beta[t
        zeta.append(temp/np.sum(temp))
    zeta = np.array(zeta)
    return zeta


def gmm_gamma(x, gamma, weights):
    gmm_g = np.zeros((x.shape[0], k, m_gmm))
    for t in range(x.shape[0]):
        for i in range(k):
            temp = []
            for m in range(m_gmm):
                a = (np.sqrt((np.linalg.det(weights['co_var'][i][m]) * (2*np.pi)**len(x[t]
                b = np.exp((-np.matmul(np.matmul(np.transpose(x[t]-weights['mu'][i][m]) ,
                temp.append(a/b)
            gmm_g[t][i] = gamma[t][i] * np.squeeze(np.array(temp/np.sum(temp)))
    return gmm_g


def new_params(x, gamma, zeta, gmm_g, weights):
    new_phi = gamma[0]
    new_A = np.sum(zeta, axis = 0)/np.sum(gamma, axis = 0)

    new_w = np.sum(gmm_g, axis = 0)/np.sum(np.sum(gmm_g, axis = 0), axis = 0)

    new_mu = np.sum(np.tile(gmm_g.reshape(gmm_g.shape + (1,)), (1,1,1,x.shape[1])) * np.ti

    new_co_var = np.zeros((k, m_gmm, x.shape[1], x.shape[1]))

    for j in range(k):
        for m in range(m_gmm):
            temp = []
            for t in range(x.shape[0]):
                te = gmm_g[t][j][m] * np.matmul((x[t] - weights['mu'][j][m]).reshape(-1,1)
```

```
                temp.append(te)
            new_co_var[j][m] = np.sum(np.array(temp), axis = 0)/np.sum(gmm_g[:,j,m], axis
    return new_phi, new_A, new_w, new_mu, new_co_var
```

Construct HMM1

```
hmm1_weights = {}
hmm1_weights['phi'] = np.ones(k)/k
print(hmm1_weights['phi'])

hmm1_weights['A'] = np.ones((k, k))/(k)
print(hmm1_weights['A'])

w = np.random.uniform(size = (k,m_gmm))
hmm1_weights['w'] = np.transpose(np.transpose(w)/np.sum(w, axis = 1))
print(hmm1_weights['w'])

hmm1_weights['mu'] = np.random.rand(k, m_gmm, vect_len)
print(hmm1_weights['mu'].shape)

co_var = [np.eye(vect_len, vect_len) for _ in range(k*m_gmm)]
hmm1_weights['co_var'] = np.array(co_var).reshape(k, m_gmm, vect_len, vect_len)
print(hmm1_weights['co_var'].shape)
```

```
[0.2 0.2 0.2 0.2 0.2]
[[0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]]
[[0.31606982 0.51582742 0.16810276]
 [0.2239833  0.48032387 0.29569283]
 [0.07732221 0.39830988 0.52436791]
 [0.32227797 0.36866502 0.30905701]
 [0.14479191 0.47455622 0.38065187]]
(5, 3, 13)
(5, 3, 13, 13)
```

Train HMM1 for phone "dha"

```
epochs = 15


print('training HMM for dha')

for epoch in range(epochs):
    n_phi = np.zeros_like(hmm1_weights['phi'])
    n_A = np.zeros_like(hmm1_weights['A'])
    n_w = np.zeros_like(hmm1_weights['w'])
    n_mu = np.zeros_like(hmm1_weights['mu'])
    n_co_var = np.zeros_like(hmm1_weights['co_var'])

    print('epoch:', epoch)
    bar = pyprind.ProgPercent(dha_train.shape[0])
```

```
    for i in range(dha_train.shape[0]):
        bar.update()
        inp = dha_train[i]
        alpha = get_alpha(inp, hmm1_weights)
        beta = get_beta(inp, hmm1_weights)
        gamma = get_gamma(alpha, beta, hmm1_weights)
        zeta = get_zeta(inp, alpha, beta, hmm1_weights)
        gmm_g = gmm_gamma(inp, gamma, hmm1_weights)
        new_phi, new_A, new_w, new_mu, new_co_var = new_params(inp, gamma, zeta, gmm_g, hm

        n_phi = n_phi + new_phi
        n_A = n_A + new_A
        n_w = n_w + new_w
        n_mu = n_mu + new_mu
        n_co_var = n_co_var + new_co_var
#         print('sample:', i)

    hmm1_weights['phi'] = n_phi/inp.shape[0]
    hmm1_weights['A'] = n_A/inp.shape[0]
    hmm1_weights['mu'] = n_mu/inp.shape[0]
    hmm1_weights['co_var'] = n_co_var/inp.shape[0]
```

```
training HMM for dha
epoch: 0
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 1
[100 %] Time elapsed: 00:00:10 | ETA: 00:00:00
Total time elapsed: 00:00:10
epoch: 2
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 3
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 4
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 5
[100 %] Time elapsed: 00:00:10 | ETA: 00:00:00
Total time elapsed: 00:00:10
epoch: 6
[100 %] Time elapsed: 00:00:10 | ETA: 00:00:00
Total time elapsed: 00:00:10
epoch: 7
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 8
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 9
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 10
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 11
```

```
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 12
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 13
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 14
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
```

Construct HMM2

```
hmm2_weights = {}
hmm2_weights['phi'] = np.ones(k)/k
print(hmm2_weights['phi'])

hmm2_weights['A'] = np.ones((k, k))/(k)
print(hmm2_weights['A'])

w = np.random.uniform(size = (k,m_gmm))
hmm2_weights['w'] = np.transpose(np.transpose(w)/np.sum(w, axis = 1))
print(hmm2_weights['w'])

hmm2_weights['mu'] = np.random.rand(k, m_gmm, vect_len)
print(hmm2_weights['mu'].shape)

co_var = [np.eye(vect_len, vect_len) for _ in range(k*m_gmm)]
hmm2_weights['co_var'] = np.array(co_var).reshape(k, m_gmm, vect_len, vect_len)
print(hmm2_weights['co_var'].shape)
```

```
[0.2 0.2 0.2 0.2 0.2]
[[0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]
 [0.2 0.2 0.2 0.2 0.2]]
[[9.35570768e-01 5.22690637e-03 5.92023252e-02]
 [1.68018622e-01 3.85079949e-01 4.46901429e-01]
 [5.91115549e-01 4.06049961e-01 2.83448988e-03]
 [2.93344893e-01 2.97193148e-01 4.09461959e-01]
 [8.98916552e-06 2.23469043e-01 7.76521967e-01]]
(5, 3, 13)
(5, 3, 13, 13)
```

Train HMM2 for "dhu"

```
print('training HMM2')
for epoch in range(epochs):
    n_phi = np.zeros_like(hmm2_weights['phi'])
    n_A = np.zeros_like(hmm2_weights['A'])
    n_w = np.zeros_like(hmm2_weights['w'])
    n_mu = np.zeros_like(hmm2_weights['mu'])
    n_co_var = np.zeros_like(hmm2_weights['co_var'])
```

```
    n_co_var = np.zeros_like(hmm2_weights['co_var'])

    print('epoch:', epoch)
    bar = pyprind.ProgPercent(dhu_train.shape[0])


    for i in range(dhu_train.shape[0]):
        bar.update()
        inp = dhu_train[i]
        alpha = get_alpha(inp, hmm2_weights)
        beta = get_beta(inp, hmm2_weights)
        gamma = get_gamma(alpha, beta, hmm2_weights)
        zeta = get_zeta(inp, alpha, beta, hmm2_weights)
        gmm_g = gmm_gamma(inp, gamma, hmm2_weights)
        new_phi, new_A, new_w, new_mu, new_co_var = new_params(inp, gamma, zeta, gmm_g, hm

        n_phi = n_phi + new_phi
        n_A = n_A + new_A
        n_w = n_w + new_w
        n_mu = n_mu + new_mu
        n_co_var = n_co_var + new_co_var
#         print('epoch:', epoch, 'sample:', i)

    hmm2_weights['phi'] = n_phi/inp.shape[0]
    hmm2_weights['A'] = n_A/inp.shape[0]
    hmm2_weights['mu'] = n_mu/inp.shape[0]
    hmm2_weights['co_var'] = n_co_var/inp.shape[0]

     training HMM2
     epoch: 0
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 1
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 2
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 3
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 4
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 5
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 6
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 7
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 8
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
     Total time elapsed: 00:00:09
     epoch: 9
     [100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
```

```
Total time elapsed: 00:00:09
epoch: 10
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 11
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 12
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 13
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
epoch: 14
[100 %] Time elapsed: 00:00:09 | ETA: 00:00:00
Total time elapsed: 00:00:09
```

HMM classifier

Computes likelihood for "dha" and "dhu" from hmm1 and hmm2. Selects the one with greater likelihood

```
px_dha = []
px_dhu = []

for l in range(dha_mfcc.shape[0]):
    px_dha.append(np.sum(get_alpha(dha_mfcc[l], hmm1_weights)))     #Correct
    px_dhu.append(np.sum(get_alpha(dha_mfcc[l], hmm2_weights)))     #Error
#print(px_dha)
count = 0
for i in range(dha_mfcc.shape[0]):
    if(px_dha[i]>=px_dhu[i]):
        count += 1
d1=count/float(dha_mfcc.shape[0])
print(count/float(dha_mfcc.shape[0]))     #likelihood for 'dha'

px_dha = []
px_dhu = []

for l in range(dhu_mfcc.shape[0]):
    px_dha.append(np.sum(get_alpha(dhu_mfcc[l], hmm1_weights)))
    px_dhu.append(np.sum(get_alpha(dhu_mfcc[l], hmm2_weights)))
#print(px_dhu)
count = 0
for i in range(dhu_mfcc.shape[0]):
    if(px_dhu[i]>=px_dha[i]):
        count += 1
d2=count/float(dhu_mfcc.shape[0])
print(count/float(dhu_mfcc.shape[0]))     #likelihood for 'dhu'
if(d1>d2):
  print("dha")
else:
  print("dhu")
```

0.8611111111111112
0.7222222222222222
dha