hmm_likelihood

May 1, 2019

[2]:
```
import numpy as np
import os
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
import librosa as ls
```

[16]:
```
zero_mfcc = []
for file in os.listdir('digits_speech/zero/'):
    if file.endswith(".wav"):
        path = os.path.join('digits_speech/zero/',file)
        signal,sr = ls.load(path ,sr=None,  duration=0.21)
        mfccs = ls.feature.mfcc(y=signal, sr=sr, n_mfcc=13, hop_length=int(0.
 ↪015*sr), n_fft=int(0.025*sr))
        zero_mfcc.append(mfccs.T)

seven_mfcc = []
for file in os.listdir('digits_speech/seven/'):
    if file.endswith(".wav"):
        path = os.path.join('digits_speech/seven/',file)
        signal,sr = ls.load(path ,sr=None,  duration=0.21)
        mfccs = ls.feature.mfcc(y=signal, sr=sr, n_mfcc=13, hop_length=int(0.
 ↪015*sr), n_fft=int(0.025*sr))
#         print(mfccs.T.shape)
        seven_mfcc.append(mfccs.T)

zero_mfcc = np.array(zero_mfcc)
seven_mfcc = np.array(seven_mfcc)
print(zero_mfcc.shape, seven_mfcc.shape)
```

```
(200, 15, 13) (200, 15, 13)
```

[17]:
```
temp = zero_mfcc - np.mean(zero_mfcc, axis = 0)
input1 = temp/np.std(zero_mfcc, axis = 0)

temp = seven_mfcc - np.mean(seven_mfcc, axis = 0)
input2 = temp/np.std(seven_mfcc, axis = 0)
```

```
in1_train,in1_test = train_test_split(input1, test_size=0.2)
in2_train,in2_test = train_test_split(input2, test_size=0.2)
print(in1_test.shape, in2_test.shape)
```

```
(40, 15, 13) (40, 15, 13)
```

[57]:
```
n_states = 5
m_gmm = 3
vect_len = 13
d = in1_train.shape[2]

phi = np.ones(n_states)/n_states
print(phi)

A = np.ones((n_states, n_states))/(n_states)
print(A)

w = np.random.uniform(size = (n_states,m_gmm))
w = np.transpose(np.transpose(w)/np.sum(w, axis = 1))
print(w)

mu = np.random.rand(n_states, m_gmm, vect_len)

co_var = [np.eye(vect_len, vect_len) for _ in range(n_states*m_gmm)]
co_var = np.array(co_var).reshape(n_states, m_gmm, vect_len, vect_len)
print(co_var.shape)
# print(co_var[4,2,:,:])
```

```
[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 0.2]]
[[0.43379958 0.34215659 0.22404383]
 [0.38170277 0.5005168  0.11778043]
 [0.42016375 0.38062803 0.19920822]
 [0.18219619 0.58797209 0.22983172]
 [0.16101121 0.27690376 0.56208503]]
(5, 3, 13, 13)
```

[36]:
```
def pdf(x, state):
    wt = w[state]
    mean = mu[state]
    var = 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 ), (x-mean[i]))/2))
        pdf = pdf + float(b/a)
    return pdf
```

[48]:
```
def forward_pass(x):
    alpha = np.zeros((x.shape[0], n_states))

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

    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(A[:,j].reshape(1,-1), alpha[i-1].reshape(-1,1))␣
↪* pdf(x[i], j)

    return alpha
```

[56]:
```
alp = forward_pass(in1_train[0])
# print(alp.shape)
print('likelihood : ', np.sum(alp))
```

```
likelihood :  1.929813136143391e-10
```