Chapter_5_Section_2_Segmentation

February 2, 2019

1 Ch 05: Concept 02

1.1 Segmentation

Import libraries and define hyper-parameters:

```
In [1]: import tensorflow as tf
        import numpy as np
        from bregman.suite import *

        k = 2
        segment_size = 50
        max_iterations = 1000
```

Define functions to get the chromogram and the dataset:

```
In [2]: chromo = tf.placeholder(tf.float32)
        max_freqs = tf.argmax(chromo, 0)

        def get_chromogram(audio_file):
            F = Chromagram(audio_file, nfft=16384, wfft=8192, nhop=2205)
            return F.X

        def get_dataset(sess, audio_file):
            chromo_data = get_chromogram(audio_file)
            print('chromo_data', np.shape(chromo_data))
            chromo_length = np.shape(chromo_data)[1]
            xs = []
            for i in range(chromo_length/segment_size):
                chromo_segment = chromo_data[:, i*segment_size:(i+1)*segment_size]
                x = extract_feature_vector(sess, chromo_segment)
                if len(xs) == 0:
                    xs = x
                else:
                    xs = np.vstack((xs, x))
            return xs
```

As required for the k-means algorithm, specify the assignment and re-centering code: Look in the clustering notebook for explanation of the these functions

```
In [3]: def initial_cluster_centroids(X, k):
            return X[0:k, :]


        def assign_cluster(X, centroids):
            expanded_vectors = tf.expand_dims(X, 0)
            expanded_centroids = tf.expand_dims(centroids, 1)
            distances = tf.reduce_sum(tf.square(tf.subtract(expanded_vectors, expanded_centroi
            mins = tf.argmin(distances, 0)
            return mins


        def recompute_centroids(X, Y):
            sums = tf.unsorted_segment_sum(X, Y, k)
            counts = tf.unsorted_segment_sum(tf.ones_like(X), Y, k)
            return sums / counts
```

Given a chromogram, extract a histogram of sound frequencies as our feature vector:

```
In [4]: def extract_feature_vector(sess, chromo_data):
            num_features, num_samples = np.shape(chromo_data)
            freq_vals = sess.run(max_freqs, feed_dict={chromo: chromo_data})
            hist, bins = np.histogram(freq_vals, bins=range(num_features + 1))
            return hist.astype(float) / num_samples
```

In a session, segment an audio file using k-means:

```
In [5]: with tf.Session() as sess:
            X = get_dataset(sess, 'TalkingMachinesPodcast.wav')
            print(np.shape(X))
            centroids = initial_cluster_centroids(X, k)
            i, converged = 0, False
            prev_Y = None
            while not converged and i < max_iterations:
                i += 1
                Y = assign_cluster(X, centroids)
                if prev_Y == Y:
                    converged = True
                    break
                prev_Y = Y
                centroids = sess.run(recompute_centroids(X, Y))
                if i % 50 == 0:
                    print('iteration', i)
            segments = sess.run(Y)
            for i in range(len(segments)):
                seconds = (i * segment_size) / float(20)
                min, sec = divmod(seconds, 60)
                time_str = str(min) + 'm ' + str(sec) + 's'
                print(time_str, segments[i])
```

```
/home/damianos/miniconda3/envs/tfCPU_Py2/lib/python2.7/site-packages/bregman/features_base.py:3
  mxnorm = P.empty(self._cqtN) # Normalization coefficients
/home/damianos/miniconda3/envs/tfCPU_Py2/lib/python2.7/site-packages/bregman/features_base.py:3
  for i in P.arange(self._cqtN)])


('chromo_data', (12, 633))
(12, 12)
('iteration', 50)
('iteration', 100)
('iteration', 150)
('iteration', 200)
('iteration', 250)
('iteration', 300)
('iteration', 350)
('iteration', 400)
('iteration', 450)
('iteration', 500)
('iteration', 550)
('iteration', 600)
('iteration', 650)
('iteration', 700)
('iteration', 750)
('iteration', 800)
('iteration', 850)
('iteration', 900)
('iteration', 950)
('iteration', 1000)
('0.0m 0.0s', 0)
('0.0m 2.5s', 1)
('0.0m 5.0s', 0)
('0.0m 7.5s', 1)
('0.0m 10.0s', 1)
('0.0m 12.5s', 1)
('0.0m 15.0s', 1)
('0.0m 17.5s', 0)
('0.0m 20.0s', 1)
('0.0m 22.5s', 1)
('0.0m 25.0s', 0)
('0.0m 27.5s', 0)
```