信息搜索与人工智能大作业

题目描述：任选某类图像为训练样本，编程实现其基于 SML 算法的类模型。要求图像的 GMM 为 6 个分量，类模型的 GMM 为 10 个分量。两级 GMM 模型的初值均由 k-means 算法获得。

要　　求：写出实验步骤，并给出收敛后的类模型

1. 基础知识

1.1 k-means

k-means 是无监督的聚类算法。它是对样本点进行聚类分析，使得每个样本点与聚类中心的距离最小。数学描述为，给定数据集 $D = \{x_1, x_2, \dots, X_n\}$，每个样本有 $x_1 \in \mathbb{R}^d$，划分 $k(k \le n)$ 个聚类集合 $S = \{S_1, S_2, \dots, S_k\}$，使得

$$arg \min_S \sum_{i=1}^{k} \sum_{x \in S_i} ||x - \mu_i||^2$$

其中，μ_i 是集合 S_i 的聚类中心。

周志华的《机器学习》中关于 k-means 的伪代码是这样的：

输入：样本集 $D = \{x_1, x_2, \dots, X_n\}$

　　聚类簇数 k

过程：

1：从 D 中随机选择 k 个样本作为初始的均值向量 $\{\mu_1, \mu_2, \dots, \mu_k\}$

2：**repeat**

3：　令 $S_i = \emptyset (1 \le i \le k)$

4：　**for** $j = 1,2, \dots, n$ **do**

5：　　计算样本 x_j 与各均值向量 $\mu_i (1 \le i \le k)$ 的距离：$d_{ji} = \left||x_j - u_i|\right|_2$

6：　　根据距离最近的均值向量确定 x_j 的簇标记：$\gamma_j = arg \min_{i \in \{1,2,\dots,k\}} d_{ji}$

7：　　将样本划入相应的簇：$C_{\gamma_j} = C_{\gamma_j} \cup \{x_j\}$

8：　**end for**

9：　**for** $i = 1,2, \dots, k$ **do**

10：　　计算新均值向量：$\mu_i' = \frac{1}{|C_i|} \sum_{x \in C_i} x$

11：　　**if** $\mu_i' \ne \mu_i$ **then**

12：　　　将当前均值向量 μ_i 更新为 μ_i'

13：　　**else**

14：　　　保持当前向量均值不变

15：　　**end if**

16：　**end for**

17：**until** 当前均值向量均为更新

1.2 GMM

高斯混合模型可以看作是由多个单高斯模型组合而成模型。一般来说，混合

模型可以是任何概率分布，但高斯分布具备良好的数学性质以及计算性能。对于多个高斯混合模型，由于它的参数无法像单高斯分布那样用极大似然的方法获得，常用 EM 算法迭代求解。其 likelihood 函数是

$$\log L(\theta) = \sum_{j=1}^{N} \log P(x_j|\theta) = \sum_{j=1}^{N} \log \left(\sum_{k=1}^{K} \alpha_k \, \phi(x|\theta_k) \right)$$

其中，x_j是第 j 个观测数据，K 是高斯混合模型的数量，α_k 是观测数据属于 k 个子模型的概率，$\phi(x|\theta_k)$是 k 个子模型的高斯分布密度概率，$\theta_k = (\mu_k, \sigma_k^2)$。

2. 实现过程

先对图像进行 YBR 空间处理，将图像分割成8×8的小块（相邻重叠 2），进行 DCT 降维，获得的特征 $x = [x^Y, x^B, x^R]$。基于该特征向量进行 k-means 聚类，获得 GMM 的初始值，然后进行 GMM 聚类。

2.1 数据预处理

假设共有 M 个语义类，每个语义类有 N 幅图片。对于同一个语义类集合 $D_i(i = 1,2,\dots,M)$ 的每幅图片 $I_j(j = 1,2,\dots,N)$ 。

1) 读取图片，转到 YCbCr 空间；

```
# 遍历文件夹里所有图片
for i in range(10):
    # opencv 读取数据建议路径不要出现中文
    temporary_img = cv2.imread(path + str(i + 1) + '.jpg')
    # 转 BGR 为 YCrCb
    img[i, :, :, :] = cv2.cvtColor(temporary_img, cv2.COLOR_BGR2YCrCb)
```

2) 分块：将图片分解为相互重叠的区域。大小为 8×8，相邻重叠为 2，相当于将一个 8×8 大小的窗隔 6 移动一次。假设图片 I_j 可以分成 L 块，每一块记作 $S_k(k = 1,2,\dots,L)$

```
# 计算该分成多少块，8*8大小，重叠2
row_num = int((img.shape[1] - 2) / 6)
col_num = int((img.shape[2] - 2) / 6)
num = col_num * row_num

split_img = np.zeros((10, num, 8, 8, 3))
# 第 j 张图片
for j in range(10):
    # 切割第 i 个小块
    for i in range(num):
        row = (int(i / row_num))*6      # 计算行的像素起始点
        col = (i % row_num)*6           # 计算列的像素起始点
        temporary_img = img[j, row:row + 8, col:col + 8, :]
```

3) 补零：对于不完整的 8×8 图片块进行补 0

```
# 对不满 8*8 的图片块进行补0
if temporary_img.shape[0] != 8:
    add_sum = 8 - temporary_img.shape[0]
    temporary_img = np.pad(temporary_img, ((0, add_sum), (0, 0), (0, 0)), 'constant')
if temporary_img.shape[1] != 8:
    add_sum = 8 - temporary_img.shape[1]
    temporary_img = np.pad(temporary_img, ((0, 0), (0, add_sum), (0, 0)), 'constant')
```

4) 在 YBR(YCbCr) 空间计算各区域的 DCT，获得特征 $S_k = [S_k^Y, S_k^B, S_k^R]$。

```
# 转换数据类型，之后就可以用 opencv 进行 dct 变换
temporary_img = temporary_img.astype(np.float32)
# 对每个通道进行 dct 变换
for k in range(3):
    temporary_img[:, :, k] = cv.dct(temporary_img[:, :, k])
```

5) 接着需要将 8*8*3 的数据降为一维，根据一些资料查找，在进行 dct 变换后，新生成的 8*8 的数据，会在左上角集中高频数据，右下角集中低频数据。查看输出的数据确实如此。可以用 zigzag 扫描的方式，将 8*8 的数据变成一维的列表

2.2 图像特征提取和语义类建模

基于同一个语义类的数据集，先做该语义类的特征抽取。在对某语义类 D_i 的每幅图片 I_j 进行分块后，每一小块相当于一个样本点，共有 L 个样本，基于该 L 个样本进行 GMM 聚类，图像的 GMM 为 6 个分量，GMM 的初始值基于 k-means 获得。

获得各个分量的参数 $\{\pi, \mu, \sum\}$ 即概率值、均值和协方差。将这样的高斯参数作为一组数据，每幅图像可得到 6 组数据。每一个语义类有 N 张图片，可得到 $6N$ 组数据。基于这些数据再进行图像的语义类建模，每个语义类用 10 个分量。

1) 图像特征提取，并获得高斯参数组，这里用 sklearn 实现

```
# 混合高斯模型个数 6 , 模型初始化参数的方式 k-means, 协方差类型 每个分量有各自不同对角协方差矩阵
gmm = GaussianMixture(n_components = 6, init_params = 'kmeans', covariance_type = 'diag').fit(img[k, :, :])
weight = gmm.weights_                # 每个混合模型的权重, 即pi, 维度 (6,)
mean = gmm.means_                    # 每个混合模型的均值, 即mu, 维度 (6,192)
covariances = gmm.covariances_       # 每个混合模型的协方差, 即sigma, 维度 (6,192)
```

2) 获得同一个语义类图像特征的高斯参数组，构造语义类的数据集

```
# 将 pi, mu, sigma 按列拼接
image_feature[6*k:6*k+6] = np.c_[weight, mean, covariances]
```

3) 基于语义类的数据集进行高斯聚类，语义类模型用 10 个高斯分量

```
# 类模型
print('Class model processing using GMM method ...')
start_time = time.time()
class_params_1 = GaussianMixture(n_components = class_num, init_params = 'kmeans', covariance_type = 'diag').fit(image_feature_1)
```

2.3 标注与模型保存

对于一个测试图片，用同样的方法抽取用 6 个高斯分量表示的特征，将这些特征参数在每个语义类模型中进行对数似然性，选择对数似然最大的那个进行标注。

1) 图像特征的抽取

```
test_img = load_picture(data_path + 'test/', test_num, row_pixel, col_pixel)
split_dct_test_img = split_and_2dct(test_img)
test_image_feature = feature_GMM(split_dct_test_img, feature_num)
```

2) 计算每个类模型下的对数似然

```
# 对于一个测试图片，在每个语义类模型中进行评价，并挑选极大似然最大的那个作为标注
result1 = class_params_1.score_samples(test_image_feature[i*feature_num: i*feature_num+feature_num]).sum()
```

3) 标注为最大值的类模型标签

```
# 获取最大值的索引
maxindex  = np.argmax(result)
print('第' + str(i+1) + '张图片的标注：', label[str(maxindex+1)])
```

4) 保存类模型

```
with open (model_path + '2.pickle', 'wb') as f:
    pickle.dump(class_params_2, f)
```

2.4 测试结果

1) 我的数据集共四个语义类：建筑、森林、天空和道路。测试了四张图片，理想的标注结果是分别是：天空、建筑、建筑、森林、道路。

```
label = {   "1":"建筑", "2":"森林",
            "3":"天空", "4":"道路"}  # 标签
```


1.jpg


2.jpg


3.jpg


4.jpg


5.jpg

2) 可以看到结果还是可以的

```
标注方式：6 组高斯分量表示的特征，在各个模型下对数似然求和
第1张图片的标注：  天空
模型一 1197.4422763003656
模型二 -896.8800026383701
模型三 2717.949628250184
模型四 3.137033027909524
第2张图片的标注：  建筑
模型一 557.4130672975268
模型二 -1653.874231574853
模型三 -10109.124582568207
模型四 -3112.8582202692533
第3张图片的标注：  建筑
模型一 -4527.559826374473
模型二 -4863.940533019403
模型三 -102093.91835569213
模型四 -119886.71883401176
第4张图片的标注：  森林
模型一 -22358.16557938116
模型二 -4698.904224254716
模型三 -2226830.6503871186
模型四 -117425.93083001938
第5张图片的标注：  道路
模型一 804.0704438560432
模型二 -510.69946489610743
模型三 -1096.9960419299596
模型四 1836.304124291269
```

2.5 存在的问题

1) 这里的只标注了一个标签，因为语义类比较少，但是其实有些图片，可能不只含有一个语义类。如图片 2，恰好四个成分都有，从对数似然结果来看，森林和道路的成分也确实是不小的。
2) 对于有些混合成分图片，仍然无法决策出理想的主要成分
3) 标注时没有完全按照 PPT 的公式来做。不可知每个语义类的概率，没有减去特征概率