

基于K-Means算法的图像分割

王铭嵩

2023年11月20日

1 K-Means算法概述

1.1 原理

K-Means算法是一种无监督分类算法，该算法的任务是将无标签数据集聚类成 k 个簇，利用贪心策略求得近似解。具体步骤为：

1. 在样本集中随机选取 k 个样本作为初始各簇中心点 μ_i 。
2. 计算所有样本点与各个簇中心之间的距离，并将其划入最近的簇中。
3. 根据簇中已有的样本点，重新计算簇中心 $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$
4. 重复2, 3.

1.2 优化

K-Means++优化了各簇中心点的选取：

1. 随机选取一个样本点作为第一个簇中心 μ_1 ,
2. 计算剩余样本点与所有簇中心的最短距离， $D(x^{(i)}) = \min[dist(x^{(i)}, C_j) | j = 1, 2, \dots, n]$ ，某样本点被选为下一个簇中心的概率为 $\frac{D(x^{(i)})^2}{\sum D(x^{(j)})^2}$,
3. 重复2，直到选出 k 个簇中心。

2 实验过程

2.1 图片输入

输入图片：

crane.jpg



已将原始图片的宽高等比缩放之原来的25%。

ramen.jpg



2.2 代码实现

仅列出代码实现原理或部分代码，详细代码请参考源代码。

imgHandler实现利用pillow库从路径获取图片，并转为二维数组，其中第一维为像素点（按行排列），第二维为RGB信息；RGB信息统一除以255以作归一化处理并转为float类型。返回值为图像原始宽高以及numpy转化过后的二维数组。

```
# in imgHandler.py:  
def img_to_array(path):  
    file=open(path,"rb")  
    img=pilImage.open(file)  
    imgData=[]  
    ...  
    return img_width,img_height,np.array(imgData)
```

此后利用K-Means++方法的思想来选出k个中心。在Version_1的实现并未利用概率来进行下一个点的选择，而是直接选择距离当前所有聚簇中心最远的点作为下一个点。在后来的评估中认为这么做可能会使某个孤立点单独成簇而缺失掉一个聚簇；在Version_2的实现中修复了该问题。

```
# in KMeans.py:  
def _get_center_and_dist(data, clusters):  
    # loop: choose the farthest point from all the  
    # centers  
    return centers, dist_to_centers  
  
def _get_center_and_dist_v2(data, clusters):  
    # K-Means++ implementation  
    return centers, dist_to_centers
```

之后进行迭代，每次将所有点归类之离其最近的聚簇中心，并重新计算聚簇的平均值作为下一次迭代的相应聚簇中心。迭代终止条件为迭代数达到目标值或所有聚簇中心的改变(delta)小到某一个特定值。返回值为一个

掩码数组，每个位置代表当前像素的所属分组。

```
# in KMeans.py:  
while epoch<iterations:  
    # epoch++  
    # updating centers  
    # if centers don't change, then break  
    # update labels  
return labels
```

最后将原始图片的数组配合掩码分别生成对应组的子图片，并存于同一目录下。

```
# in imgHandler.py  
def array_to_imgs(data, mask, path):  
    # init img group  
    # put pixels  
    # generate outputs
```

2.3 结果展示

聚类结果：

crane.jpg

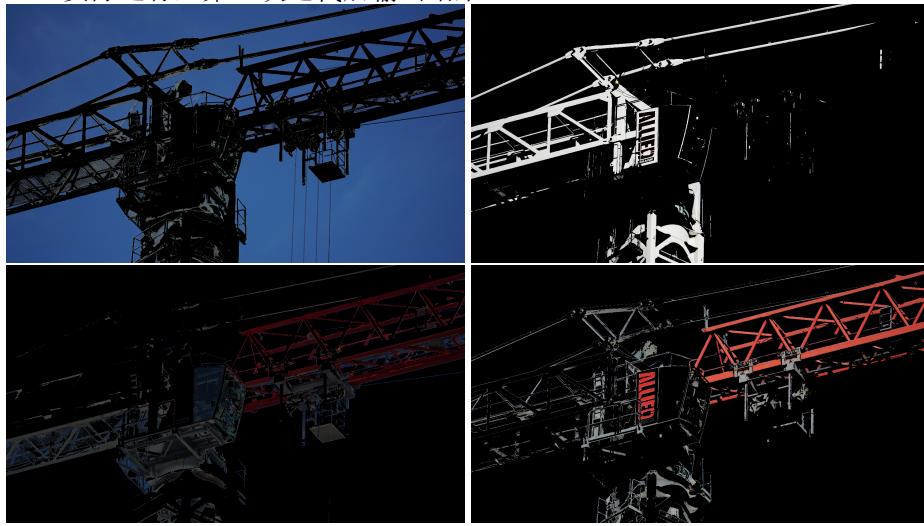
2.3.1 Version=1, clusters=3, iterations=50, delta=0.0001

实际运行至第9次迭代后输出结果。



2.3.2 Version=1, clusters=4, iterations=50, delta=0.00001

实际运行至第17次迭代后输出结果。



2.3.3 Version=2, clusters=3, iterations=15, delta=0.0001

实际运行至第15次迭代后输出结果。



ramen.jpg

3 思考

仅将RGB值的相似度作为聚簇依据，而没有考虑图片距聚簇中心的距离的因素。使得分出的类只有数值意义而实际意义不大。在考虑距离时或许应该构造空间信息和颜色信息的一个函数，从而使聚类更有实际意义。