K-Means聚类 苏潇 PB14011003

环境及主要依赖的库

OS:windows10

python version 2.7.13

modules:numpy,scipy,PIL,random

基本思想

kmeans算法的思想很简单,首先在所有样本中随机选择k个作为初始的center.之后对样本中每个点,通过计算其与各个center之间的距离,将其归入距离最近的那个center对应的cluster.归类完毕后,更新各个center的值,用其对应的cluster中所有向量的均值代替原center.然后再进行下一轮迭代.这样子不断迭代,直到所有的center都不再变化,即完成聚类.

算法最初调试时,发现运行速度相比调用标准的kmeans算法慢了很多,后来发现是因为用了很多for循环,之后把代码改成向量化计算,只保留了两个对k的for循环,因为k一般比较小,对性能影响不会很大,所以也就保留了.另外,为了保证不因为过度追求完全收敛而不断迭代,设置了一个迭代深度,超过迭代深度即使不收敛也停止迭代.这样改进后,算法的运行速度有了很大提高,基本和scipy的cluster模块提供的kmeans算法在同一量级了.

kmeans初始中心的选取是使用random库随机选取.

图像处理使用python的Pillow库,操作比较方便.对图像,用rgb方式打开,每个像素点对应一个三维向量(r,g,b).这样子,只需要把所有像素点数据转换成一个行向量,将这些行向量拼成的矩阵作为输入,就完成图像到算法输入的转化.

算法运行完毕后,把被归为同一cluster的像素点全都改成该cluster的center的值,再save就得到了经kmeans压缩后的新的图片。

实验结果

这里k选取了2,4,8,16,32五种情况.效果如下:

原图

k=2

k=4

k=8

k=16

k=32

通过简单肉眼对比,可以看出:

当k较小时(k=2,4),图片还原度较低,只能大致区分出潜水员,鱼和周围环境的轮廓.

从k=8开始,有了较高的还原度,但细节上还不够准确.当k=32时,基本可以做到较高程度的图像还原了,压缩效果也比较好,60KB的原图压缩后只有24KB.

可见,kmeans应用时,需要选取一个适中的k值,以保证运行速度和还原效果.

(注:k很大时似乎由于初始中心的选取问题以及迭代深度的设置,有时似乎会出现NaN问题,这个bug目前还没有想到太好的修复方法,不过对大小适中的k是没有问题的)