**《模式识别原理》课程**

**实验报告**

题 目 K均值聚类分析与图像分割

单 位 一院

专业班 智胜班

学 号 50620201003

姓 名 孙文垚

2022年9月

1. 实验目的

1.通过K均值算法的实现加深对聚类算法理论的理解；

2.通过聚类的具体实现和应用提升学员的工程实践能力；

3.初步掌握聚类分析的性能评价。

二、实验条件

硬件：电脑：联想拯救者Y7000P2020H

处理器：Intel(R) Core(TM) i7-10875H CPU @ 2.30GHz 2.30 GHz

机带RAM：16.0 GB

系统类型：64 位操作系统, 基于 x64 的处理器

操作系统：Windows 10 家庭中文版

软件：PyCharm Community Edition 2021.3.2

三、实验原理

K均值算法是一种经典的非监督机器学习算法，其基本思路是通过迭代将所有数据分配到与它距离最小的聚类中心来实现数据类簇的自动划分。终止条件是没有对象被重新分配到其它聚类中心，或者所有聚类中心不在发生变化，此时误差平方和最小。利用K均值聚类算法可以实现对图像数据的分割，如航拍或卫星照片的地物地貌分割、自然场景中的目标分割。

四、实验内容

（1）编程实现K均值聚类算法

采用python编程语言。

类别数K、数据集矩阵img和最大迭代次数iteration等参数设置为全局变量。使用不同的函数完成对数据的处理。

randcenter()函数生成初始的K个中心点，cal\_distance()函数计算距离矩阵，cal\_label()函数判断样本点属于哪一类，cal\_center()函数根据类别计算新的类中心点。

距离采用的是简单欧式距离。

**def** K\_means():

    randcenter()

    index **=** 0  #初始迭代次数

**while** True:

        old\_center **=** center

        cal\_distance()  #计算distance

        cal\_label()  #计算标签label

        cal\_center()  #算出新的中心点center

        index **+=** 1

        print('当前迭代次数为：', index)

**if** index > iteration **or** old\_center **==** center:

**break**

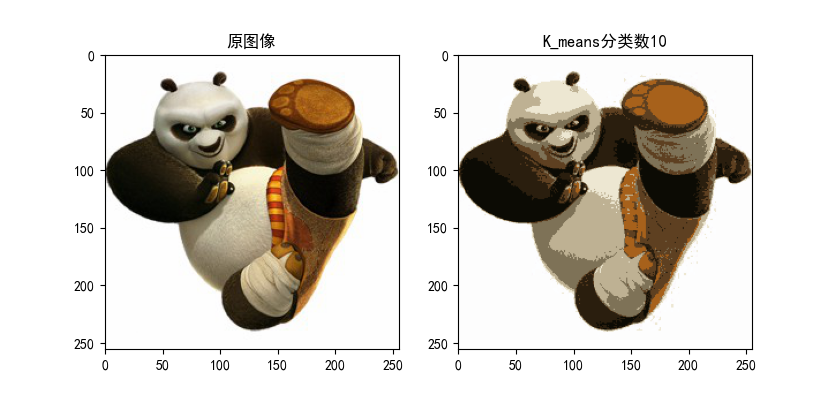
    print('最终迭代次数为：', index)

**return** 0

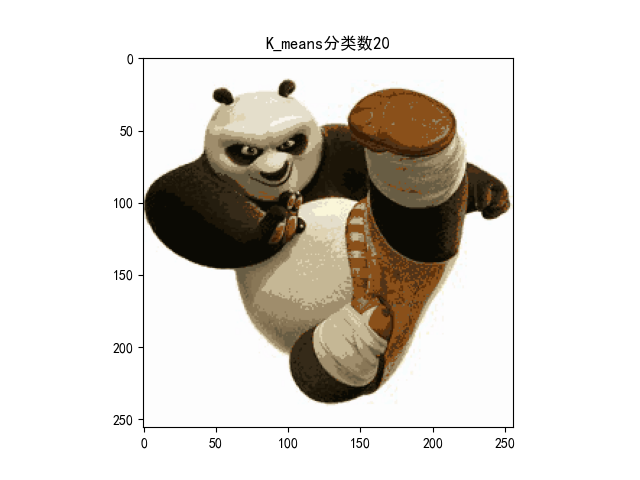
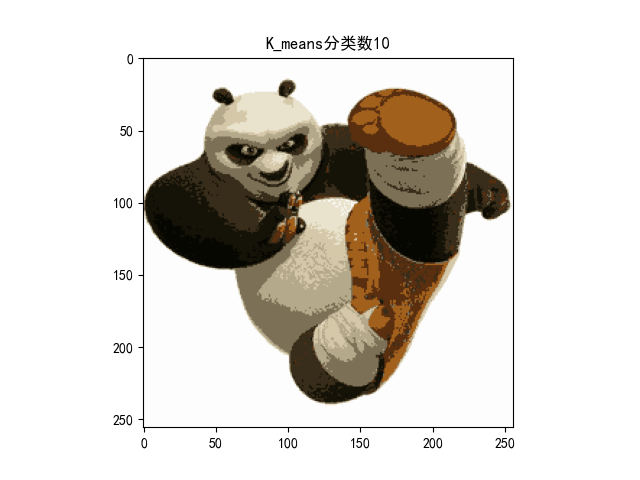
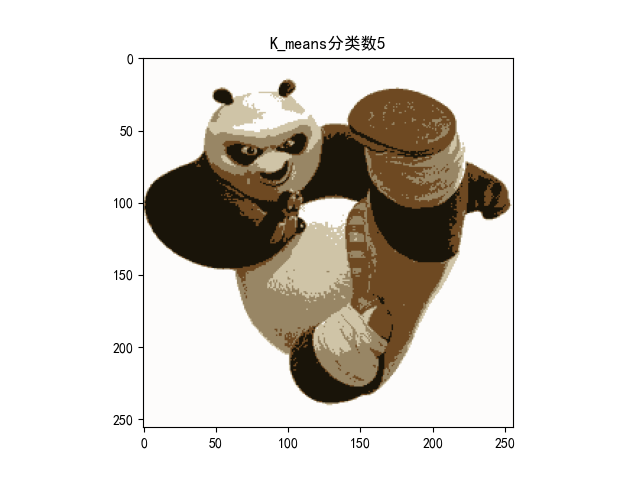
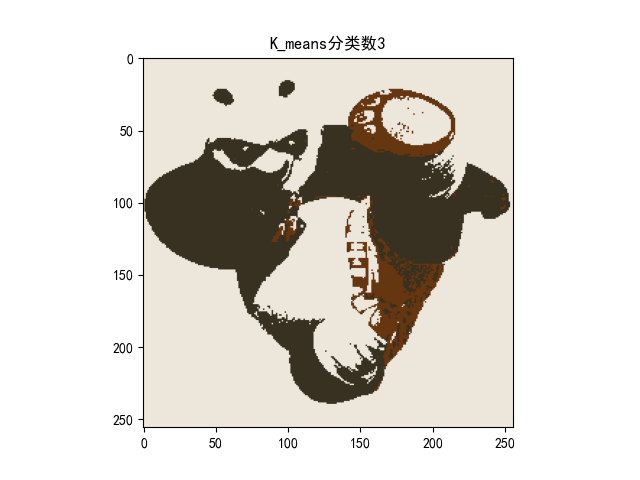
（2）基于K均值算法实现彩色图像分割

实验采用256\*256的jpg格式的图片。通过matplotlib.pyplot.imread()函数读取图片数据转化为256\*256\*3的数组格式。通过K\_means方法找出聚类中心，将同一类的像素点都赋为中心点的像素。最终通过matplotlib.pyplot.imshow()函数输出分割前后两张图片。

五、实验结果及分析



采用不同的K值，处理结果如下：



可以看到K越大越精细，符合预期。

六、程序代码

**import** random

**import** numpy as np

**import** matplotlib.pyplot as plt

img **=** plt.imread('picture.jpg')  #导入图片数据

shape **=** img.shape

#plt.imshow(img)

#plt.show()

K **=** 10  #聚类个数

iteration **=** 5  #最大迭代次数

center **=** []  #聚类颜色中心位置，RGB三元组

label **=** [[**-**1 **for** j **in** range(0, 256)] **for** i **in** range(0, 256)]  #类别号（标签）

distance **=** [[[0 **for** z **in** range(K)]**for** j **in** range(0, 256)] **for** i **in** range(0, 256)]

#各图像点到不同聚类中心的距离

**def** randcenter():  #生成初始聚类中心位置

    center.append(img[random.randint(0, shape[0] **-** 1), random.randint(0, shape[1] **-** 1)])

**for** i **in** range(1, K):

**while** 1:  #确保聚类中心不同

            color **=** img[random.randint(0, shape[0] **-** 1), random.randint(0, shape[1] **-** 1)]

            flag **=** 1  #无重复为1，重复为0

**for** j **in** range(i):

**if** (color **==** center[j]).all():

                    flag **=** 0

**break**

**if** flag **==** 1:  #无重复则跳出，追加写入

**break**

        center.append(color)

**return** 0

**def** cal\_distance():

**for** i **in** range(shape[0]):

**for** j **in** range(shape[1]):

**for** z **in** range(K):

                distance[i][j][z] **=** (int(img[i][j][0]) **-** center[z][0])**\*\***2 **+** \

                                    (int(img[i][j][1]) **-** center[z][1])**\*\***2 **+** \

                                    (int(img[i][j][2]) **-** center[z][2])**\*\***2

                #print(distance[i][j][z])

**return** 0

**def** cal\_label():

**for** i **in** range(shape[0]):

**for** j **in** range(shape[1]):

            min **=** 196608

**for** z **in** range(K):

**if** distance[i][j][z] < min:

                    min **=** distance[i][j][z]

                    label[i][j] **=** z

**return** 0

**def** cal\_center():

    sum **=** [[0, 0, 0]**for** i **in** range(K)]

    count **=** [0 **for** i **in** range(K)]  #统计第K类有多少个

**for** i **in** range(shape[0]):

**for** j **in** range(shape[1]):

            count[label[i][j]] **+=** 1

            sum[label[i][j]][0] **+=** int(img[i][j][0])

            sum[label[i][j]][1] **+=** int(img[i][j][1])

            sum[label[i][j]][2] **+=** int(img[i][j][2])

**for** i **in** range(K):

        center[i][0] **=** sum[i][0] **/** count[i]

        center[i][1] **=** sum[i][1] **/** count[i]

        center[i][2] **=** sum[i][2] **/** count[i]

**return** 0

**def** K\_means():

    randcenter()

    print('center = ', center)

    print('中心点个数：', len(center))

    index **=** 0  #初始迭代次数

**while** True:

        old\_center **=** center

        cal\_distance()  #计算distance

        cal\_label()  #计算标签label

        cal\_center()  #算出新的中心点center

        index **+=** 1

        print(index)

**if** index > iteration **or** old\_center **==** center:

**break**

    print('最终迭代次数为：', index)

**return** 0

**def** change\_img():

**for** i **in** range(shape[0]):

**for** j **in** range(shape[1]):

            img[i][j] **=** center[label[i][j]]

**return** 0

K\_means()

plt.rcParams['font.sans-serif'] **=** ['SimHei']

plt.rcParams['axes.unicode\_minus'] **=** False

plt.subplot(121), plt.title("原图像")

plt.imshow(img)

change\_img()

plt.subplot(122), plt.title("K\_means分类数10")

plt.imshow(img)

plt.show()