**机器学习实验报告02**

**算法1 kmeans**

import numpy as np

import cv2

import matplotlib.pyplot as plt

img = cv2.imread('./cloud/cloud3.jpeg')

data = img.reshape((-1,3)) *# n行 3列*

data = np.float32(data)

criteria = (cv2.TERM\_CRITERIA\_EPS + cv2.TERM\_CRITERIA\_MAX\_ITER, 10, 1.0)

K =2

ret, label, center=cv2.kmeans(data, K, None, criteria,10, cv2.KMEANS\_RANDOM\_CENTERS)

center = np.uint8(center)

res1 = center[label.flatten()] *# 根据索引来取值，最后结果的大小同索引的大小*

res2 = res1.reshape((img.shape))

plt.subplot(121)

plt.imshow(img[:,:,::-1])

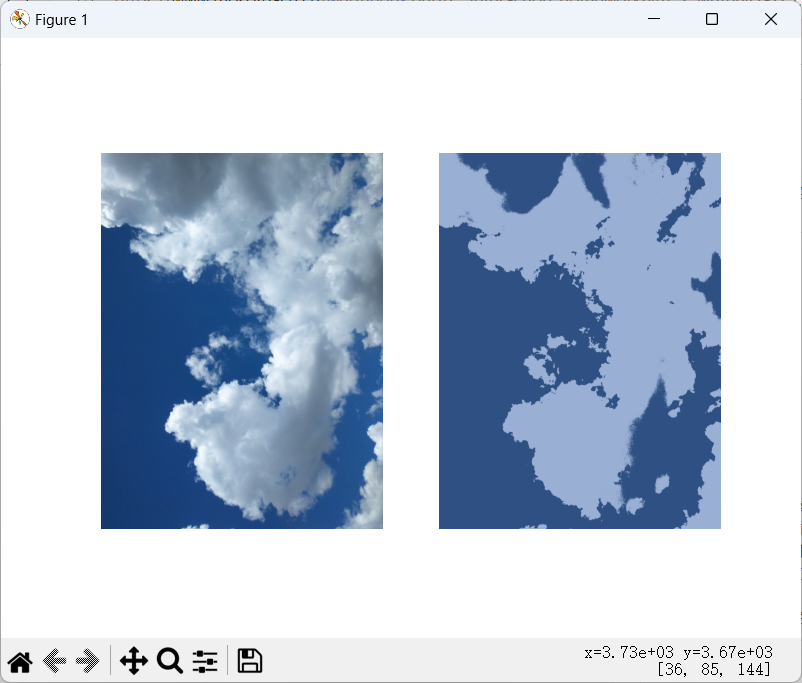
plt.axis('off')

plt.subplot(122)

plt.imshow(res2[:,:,::-1])

plt.axis('off')

plt.show()



**算法流程**

1. 读取待处理图像 cloud3.jpeg。
2. 使用 reshape 将图像中每个像素点的 RGB 值作为一个单元处理，将图像数据变为一个大小为 (n, 3) 的数组，其中 n 是图像的像素总数。
3. 将图像数据转换为 kmeans 函数可以处理的数据类型，即将数据类型转换为 np.float32。
4. 设置 K 值为 2，即希望将图像分为 2 类。
5. 定义聚类的停止条件为迭代次数达到最大值 10 或者精度达到一定阈值。
6. 调用 cv2.kmeans 函数进行 K 均值聚类，传入图像数据、K 值、停止条件等参数，返回聚类结果和聚类中心。
7. 将聚类中心的数据类型转换为 np.uint8。
8. 根据聚类结果中的标签，将每个像素点替换为其对应分类的中心点像素值，得到替换后的图像结果 res1。
9. 使用 reshape 调整替换后的图像的形状为原始图像的形状，得到最终的图像结果 res2。
10. 使用 matplotlib 显示原始图像和处理结果的对比。

**算法2 *大津算法***

import cv2

import numpy as np

from matplotlib import pyplot as plt

image = cv2.imread("./cloud/cloud3.jpeg")

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

plt.subplot(131), plt.imshow(image, "gray")

plt.title("source image"), plt.xticks([]), plt.yticks([])

plt.subplot(132), plt.hist(image.ravel(), 256)

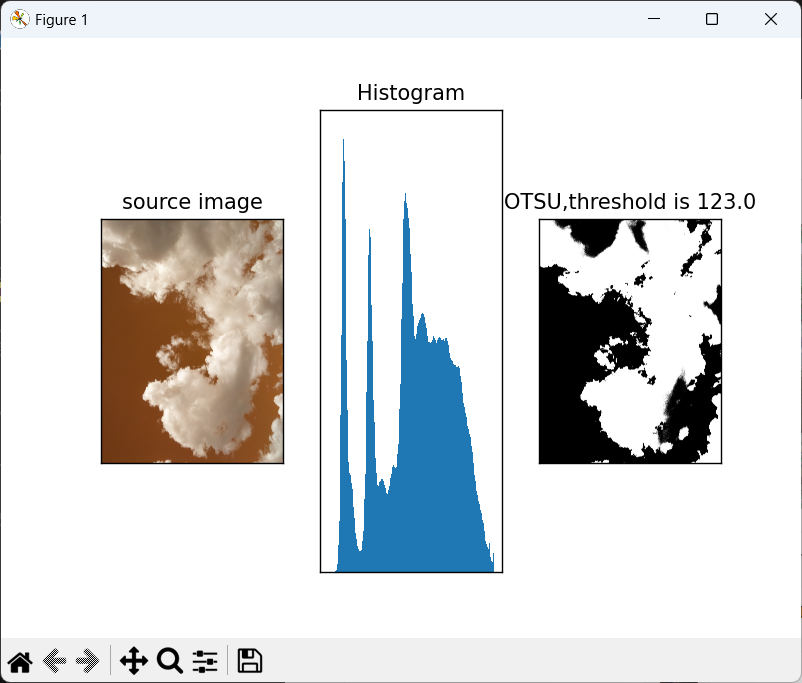
plt.title("Histogram"), plt.xticks([]), plt.yticks([])

ret1, th1 = cv2.threshold(gray, 0, 255, cv2.THRESH\_OTSU) *#方法选择为THRESH\_OTSU*

plt.subplot(133), plt.imshow(th1, "gray")

plt.title("OTSU,threshold is " + *str*(ret1)), plt.xticks([]), plt.yticks([])

plt.show()



**算法流程**

1. 读取待处理图像 cloud3.jpeg。
2. 将图像转换为灰度图像，使用 cv2.cvtColor 函数将彩色图像转换为灰度图像，得到 gray。
3. 绘制三个子图：

* 第一个子图显示原始图像。
* 第二个子图显示原始图像的直方图。
* 第三个子图显示使用OTSU自动阈值法处理后的二值化图像。

1. 使用 cv2.threshold 函数对灰度图像进行二值化处理，传入灰度图像、初始阈值0、最大像素值255和方法选择为OTSU（cv2.THRESH\_OTSU）。
2. 获得OTSU自动阈值法确定的最佳阈值 ret1 和二值化图像 th1。
3. 显示绘制的三个子图。