代码交接文档

Gen\_color\_pic 存放训练集

Tests 存放预测集

color\_pic\_gen.py 产生训练集

rgb\_enclidean.py 计算RGB的欧式距离

rgb\_euclidean\_lab.py RGB加权欧氏距离

color\_HSV\_rxy.py HSV的欧氏距离

color\_HSV+LAB.py HSV加权欧氏距离

hsv\_distributed.py HSV颜色分布法

假设有n个train，m个test

lab\_cie2000.py 时间复杂度n\*m的CIEDE2000

lab\_cie2000\_optimize.py时间复杂度n+m的CIEDE2000

代码最后统计是否匹配是按文件名。

例如test\_blue.jpg和blue.jpg 中’blue’ in ‘test\_blue’中，所以会匹配。

因此文件名尽量不要有交集，例如blue 在 test\_blue 也在 deep\_blue中这样会重复计算。

lab\_cie2000\_optimize.py

CIEDE算法代码中

1. 计算所有测试集的LAB向量进行CIEDE算法的实现。
2. 计算所有训练集的LAB向量进行CIEDE算法的实现。
3. 计算CIEDE △E
4. 计算正确率

初始测试为400\*400 计算量较大 现改为50\*50 准确率在现有样本下不影响。后续根据结果调整，越大越准确，越小越快。

还有改进思路是改进代码，使用numpy或多线程。

color\_pic\_gen.py

产生train图片。

颜色空间为BGR。若产生蓝色则：

img\_blue = np.zeros([400, 400, 3], np.uint8)  
 img\_blue[:, :, 0] = np.zeros([400, 400]) + 255

Img[:,:,0]为B

img[:,:,1]为G

img[:,:,2]为R



Train产生图片通过这个程序或者自写程序

Test需网上一个个下载，然后改名字。目前的判断方式是

代码最后统计是否匹配是按文件名。

例如test\_blue.jpg和blue.jpg 中’blue’ in ‘test\_blue’中，所以会匹配。

因此文件名尽量不要有交集，例如blue 在 test\_blue 也在 deep\_blue中这样会重复计算。

256种颜色的话用0-255表示