电工导 C 第十四次实验报告

姓名: 宋士祥　　学号:521030910013　　班级:F2103001

2022 年 12 月 21 日

1　实验概览

本次实验介绍了 LSH 的相关知识。LSH 的基本想法是先根据哈希值对图片进行分类, 再通过比较特征向量来确定其最终的相似度。由于我们这样处理比直接一一比较特征向量会快很多，因此会大大节省我们遍历的时间。

本次实验我们需要完成一个类似的流程，建立特征向量并对其进行汉明码分析和哈希值计算。

2　实验环境

本次实验采用所需的实验环境如下：

- Docker 中的`sjtumic/ee208` 镜像
- Python3（使用 VSCode 编译）
- `numpy` 扩展。
- `OpenCV` 环境。(均在 SJTUEE208 镜像中给出)

3　问题重述和主要代码说明

3.1　建立特征向量

由题设描述，特征向量即为图片左上，左下，右上，右下四个方位的颜色直方图，套用 lab9 的代码即可得出，对应源码：

```
# 获取特征向量
def make_color_histogram(img):
    # 读取图片
    blue1, green1, red1 = 0, 0, 0
    blue2, green2, red2 = 0, 0, 0
    blue3, green3, red3 = 0, 0, 0
    blue4, green4, red4 = 0, 0, 0
    # 记录数值RGB
    for i in range(len(img)//2):
        for j in range(len(img[0])//2):
            blue1  += img[i][j][0]
            green1 += img[i][j][1]
            red1   += img[i][j][2]
        for j in range(len(img[0])//2,len(img[0])):
            blue2  += img[i][j][0]
            green2 += img[i][j][1]
```

```
            red2   += img[i][j][2]
    for i in range(len(img)//2,len(img)):
        for j in range(len(img[0])//2):
            blue3  += img[i][j][0]
            green3 += img[i][j][1]
            red3   += img[i][j][2]
        for j in range(len(img[0])//2,len(img[0])):
            blue4  += img[i][j][0]
            green4 += img[i][j][1]
            red4   += img[i][j][2]
    v1, v2, v3, v4,  = [blue1, red1, green1],[blue2, red2, green2],[blue3, red3, green3]
                                                ,[blue4, red4, green4]
    v1, v2, v3, v4 = normalize(v1), normalize(v2), normalize(v3), normalize(v4)
    v = np.concatenate((v1,v2,v3,v4))
    return v
```

需要注意的是，normalize 是我们的自定义函数，用于对向量的归一化。

3.2 获取汉明码

在 ppt 中给出了无需利用汉明码获得哈希值的方法，但鉴于 python 有比较强大的字符串功能，我们可以直接利用字符串获取其汉明码，对应代码

```
def normalization(v):
  v= v.copy()
  for i in range(12):
      if v[i] < 0.3:
          v[i] = 0
      elif v[i] < 0.6:
          v[i] = 1
      else:
          v[i] = 2
  return v

# 获得汉明码
def Hamming(v):
  ans = str()
  for i in v:
      if i == 0:
          ans += '00'
      elif i == 1:
          ans += '10'
      else:
          ans += '11'
  return ans
```

3.3 利用局部敏感哈希进行检索

局部敏感哈希即直接取汉明码的某些位获得其哈希函数，对应代码

```
def g(p, index):
  ans = ''
```

```
    for i in index:
        ans += p[i]
    return ans
```

3.4 特征向量的比较

注意到我们所有的特征向量的模长均为 4，因此我们只需要对这些特征向量进行点乘，然后取最大的特征向量即可。

4 结果分析与不同的投影集合的比较

4.1 不同投影集合的比较

我们首先取全集汉明码分析，也即`key=list(range(0,24))`，得到的结果如下：

```
The similar images are:
12
38
The most fit is 38
```

这两张图片都是沙漠中的树，可以说结果较好。

通过对比其分布，我们发现，第 4 位，第 8 位，第 12 位，第 18 位，第 22 位的 01 分布差异较为明显，因此我们尝试更换 key，得到结果

```
The similar images are:
5
8
12
25
38
40
The most fit is 38
```

我们发现这一情形下基本带有黄色的图片都被视为了相似图片。

考虑我们的汉明码是按照 BGR 的顺序，又考虑到目标图片上面是蓝色，下面是黄色，我们可以尝试 2，8，18，24 这四位比较，对应的结果

```
The similar images are:
12
17
23
26
36
37
38
40
The most fit is 38
```

在这样的情形下，我们发现，基本上天空为蓝色，地面为黄色的图片都被算入在中间。

可以看出，我们的特征向量其实略显粗糙，但是基本可以满足我们的需求

4.2 时间对比

由于我们是对每一个数均求的汉明码，并且由于 python 对字符串的效率并没有很好的优化，因此我们的效果并不明显，如果我们将时间从开始比较算起，二者的耗时分别为 0.01s 和 0.014s。这一方面由于我们是计算出了完整的汉明码导致最后结果的差异反而并不明显。

但是对比 CNN 网络，取 extract feature 的时间约为 6s，而我们的算法仅为 2s，虽然我们的略显粗糙，但是时间更快。

值得一提的是，CNN 的五个最相关为 38 12 13 16 39，和我们略显粗糙的比较竟有很大的相似之处。

5 拓展思考

5.1 本练习中使用了颜色直方图特征信息，检索效果符合你的预期吗？检索出的图像与输入图像的相似性体现在哪里？

检索效果承上所述，基本符合我们的预期。因为我们的特征信息其实略显粗糙，但是基本上能够排除很多无关的图片。

特别的，如果我们探索全集，我们甚至可以意外的发现结果相当的好。

检索出图片的相似性首先体现在颜色上。由于我们的 target 偏冷色调，因此我们检索出来的图片以蓝色，黄色为主，可以说基本上体现了主色调。事实上，我们对于 target 选取均为偶数位的重要原因即为其能够反映该位置的主色调

5.2 能否设计其他的特征？

一方面，基于本次实验的特点，我们可以缩小范围，例如画成 3×3 九份的空间，或者更高。这样汉明码相对更长，也更能体现特征。避免一些下方为豹子的图片出现。

又或者，我们可以利用之前课程所学习的 SIFT 等检索特征点，利用特征点的**相对于全局**的坐标进行比较。（比如横坐标的 0.38，要归一化）。

又或者，我们可以尝试提取矩阵的特征向量等。

对于图像识别这一领域而言，图像提取特征算法是目前的一个重要方向。本位仅粗略的对其进行分析。