



浙江大學

数字媒体资源管理实验报告

**Digital Asset Management
(Experiments)**

姓 名	李沛瑶
指导老师	张宏鑫
学 号	3180101940
专业班级	数媒 1801

二〇二〇年

秋冬学期

实验四

实验名称： 图像相似性检测

指导老师： 张宏鑫

成绩： _____

一、实验内容和要求

Image similarity computing

- Given: 10 images, I_1, I_2, \dots, I_{10}
- Goal:
 - compute similarities between I_1 and $I_i (i = 2, \dots, 10)$
 - find the most similar image to I_1

1. 输入十张图像，分别计算每张图像的特征。
2. 对其他九张图像分别计算其与第一张图像的相似度，选出与第一张图像相似度最高的图像。

二、实验原理分析

本实验的重点在于图像特征的表达，利用不同算法计算，用不同的形式表达两张图像之间的相似度。

1、颜色矩算法（实验中实现在 RGB、HSV、LAB 空间分别计算图像相似度）

主要思想：将一张图像特征数据转化为一个向量，如下图所示，对两个图像，我们分别得到了一个对应的向量，接下来只需要计算两个向量的夹角，使用余弦公式即可。

- Image Feature vector: $I_i \rightarrow \mathbf{F}_i$
- RGB based moments, 9 float numbers
- Similarity:
 - dot product:
$$\frac{\mathbf{F}_i \cdot \mathbf{F}_j}{\|\mathbf{F}_i\| \|\mathbf{F}_j\|}$$

具体实现：

三个颜色矩的数学定义如下：

$$\mu_i = \frac{1}{N} \sum_{j=1}^N p_{i,j}$$

$$\sigma_i = \left(\frac{1}{N} \sum_{j=1}^N (p_{i,j} - \mu_i)^2 \right)^{\frac{1}{2}}$$

$$s_i = \left(\frac{1}{N} \sum_{j=1}^N (p_{i,j} - \mu_i)^3 \right)^{\frac{1}{3}}$$

其中， p_{ij} 表示彩色图像第 j 个像素的第 i 个颜色分量， N 表示图像中的像素个数。

以 YUV 颜色空间为例，图像的三个分量 Y,U,V 图像的前三阶颜色矩组成一个 9 维直方图向量，即图像的颜色特征表示如下：

$$\mathbf{F}_{color} = [\mu_Y, \sigma_Y, s_Y, \mu_U, \sigma_U, s_U, \mu_V, \sigma_V, s_V]$$

接下来对于两个图像计算得到 $\mathbf{F}_1, \mathbf{F}_2$ 两个向量，我们可以通过余弦公式（如下图）得到他们之间的相似度的一种表达方式。结果在 0~1 之间，数值越高，相似度越高。

$$\frac{\mathbf{F}_i \cdot \mathbf{F}_j}{\|\mathbf{F}_i\| \|\mathbf{F}_j\|}$$

本次实验中实现了使用 RGB、HSV、LAB 三种颜色空间的颜色矩计算图像相似度。

2、图像余弦相似度

把图片表示成一个向量，通过计算向量之间的余弦距离来表征两张图片的相似度。

本实验中用到的方法，主要是把图像的每一个像素，在三个颜色空间求均值，将每个像素的计算结果连接成一个向量，通过余弦公式，计算两张图像对应的向量夹角的大小，以此来得到两张图像的相似度。

但这种算法，显而易见的计算量，非常大，而实验中，该算法也确实运行速度较慢。

3、SSIM（结构相似度量）

SSIM 是一种全参考的图像质量评价指标，分别从亮度、对比度、结构三个方面度量图像相似性。SSIM 取值范围[0, 1]，值越大，表示两张图像相似度越高。

在实际应用中，可以利用滑动窗将图像分块，采用高斯加权计算每一窗口的均值、方差以及协方差，然后计算对应块的结构相似度 SSIM，最后将平均值作为两图像的结构相似性度量，即平均结构相似性 SSIM。

SSIM 原理及数学公式如下：

假设我们输入的两张图像分别是 x 和 y，那么：

$$SSIM(x, y) = [l(x, y)]^{\alpha} [c(x, y)]^{\beta} [s(x, y)]^{\gamma} \quad (1)$$

其中：

$$\alpha > 0, \beta > 0, \text{ and } \gamma > 0.$$

$$l(x, y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1},$$

$$c(x, y) = \frac{2\sigma_{xy} + c_2}{\sigma_x^2 + \sigma_y^2 + c_2},$$

$$s(x, y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

其中 $l(x, y)$ 是亮度比较 (luminance) , $c(x, y)$ 是对比度比较 (contrast) , $s(x, y)$ 是结构比较 (structure)。

μ_x 和 μ_y 分别代表 x, y 的平均值, σ_x 和 σ_y 分别代表 x, y 的标准差。 σ_{xy} 代表 x 和 y 的协方差。

而 c_1, c_2, c_3 分别为常数, 避免分母为 0 带来的系统错误。

其中:

$$c_1 = (k_1 L)^2, \quad c_2 = (k_2 L)^2$$

$L = 2^B - 1$, 为像素值的范围。

$k_1 = 0.01, k_2 = 0.03$ 为默认值

在实际工程计算中, 我们一般设定 $\alpha = \beta = \gamma = 1$, 以及 $c_3 = c_2/2$, 可以将 SSIM 简化为下:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

SSIM 性质:

- 1、对称性, 即 $SSIM(x, y) = SSIM(y, x)$
- 2、SSIM 是一个 0 到 1 之间的数, 越大表示输出图像和无失真图像的差距越小, 即图像质量越好。当两幅图像一模一样时, $SSIM = 1$ 。

4、通过直方图计算图像相似度

首先对图像进行统计，得到图像的直方图。

然后对两张图像的直方图的相似度进行计算，计算方法有四种：

(1) 相关性计算(CV_COMP_CORREL)

$$d(H_1, H_2) = \frac{\sum_I (H_1(I) - \bar{H}_1)(H_2(I) - \bar{H}_2)}{\sqrt{\sum_I (H_1(I) - \bar{H}_1)^2 \sum_I (H_2(I) - \bar{H}_2)^2}}$$

$$\text{其中: } \bar{H}_k = \frac{1}{N} \sum_J H_k(J)$$

(2) 卡方计算(CV_COMP_CHISQR)

$$d(H_1, H_2) = \sum_I \frac{(H_1(I) - H_2(I))^2}{H_1(I)}$$

(3) 十字计算(CV_COMP_INTERSECT)

$$d(H_1, H_2) = \sum_I \min(H_1(I), H_2(I))$$

(4) 巴氏距离计算(CV_COMP_BHATTACHARYYA)

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2} \sum_I \sqrt{\bar{H}_1(I) \cdot \bar{H}_2(I)}}}$$

本次实验中主要用到的是巴氏距离。

5、通过哈希值差异计算图像相似度

具体算法流程：

- 1) 将输入图片统一压缩成 8*8 的小图
- 2) 将图片转化为灰度图

- 3) 计算图片的 hash 值，这里的 hash 值是 64 位 01 字符串。或者 16 位 hash 值。
- 4) 通过 hash 值来计算汉明距离

哈希值分三种：

- 1) aHash：平均值哈希。速度比较快，但是常常不太精确。
- 2) pHash：感知哈希。精确度比较高，但是速度方面较差一些。
- 3) dHash：差异值哈希。精确度较高，且速度也非常快

aHash：对 8×8 的灰度图像，先求平均值，然后对于每一个像素的灰度值，如果大于平均值就是 1，否则为 0，得到一个长度为 64 的 01 字符串，即为该图像的 aHash 值。

dHash：对 8×8 的灰度图像，对于每一个像素的灰度值，如果大于上一个像素的值就是 1，否则为 0，得到一个长度为 64 的 01 字符串，即为该图像的 dHash 值。

pHash：平均哈希算法过于严格，不够精确，更适合搜索缩略图，为了获得更精确的结果可以选择感知哈希算法，它采用的是 DCT（离散余弦变换）来降低频率的方法

步骤：

- 1) 缩小图片： 32×32 是一个较好的大小，这样方便 DCT 计算
- 2) 转化为灰度图
- 3) 计算 DCT：利用 Opencv 中提供的 `dct()` 方法，注意输入的图像必须是 32 位浮点型，所以先利用 numpy 中的 `float32` 进行转换
- 4) 缩小 DCT：DCT 计算后的矩阵是 32×32 ，保留左上角的 8×8 ，这些代表的图片的最低频率
- 5) 计算平均值：计算缩小 DCT 后的所有像素点的平均值。

- 6) 对 DCT 结果的每一位：大于平均值记录为 1，反之记录为 0。
- 7) 得到信息指纹：64 个信息位，每 4 位合成一位，转化为 16 位的 hash 值。
- 8) 最后比对两张图片的指纹，获得汉明距离即可。

汉明距离 Hamming_distance:

汉明距离表示两个（相同长度）字符串对应位不同的数量，我们以 $d(x,y)$ 表示两个字 x,y 之间的汉明距离。对两个字符串进行异或运算，并统计结果为 1 的个数，结果就是汉明距离。

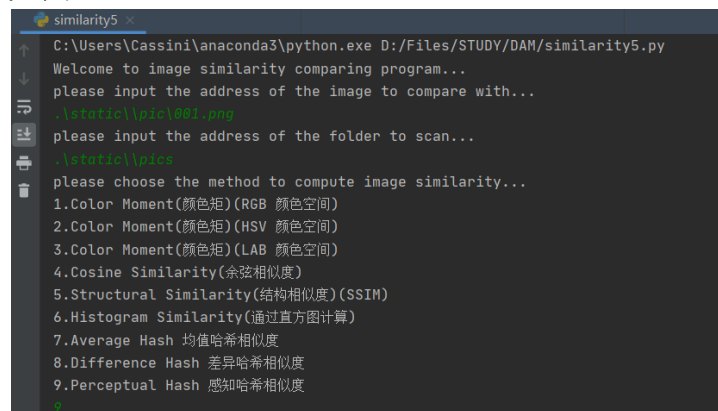
三、源代码与分析

源代码及注释和分析详见 similarity.py 文件

四、运行效果展示

1、本实验分别提供了具有十张、一千张、一万张图像的图像集以供比较，相对路径分别是".\static\pics"、".\static\picss"和".\static\picsss"。选择的源图像是".\static\pic\001.png"。图像集目录下有对源图进行缩小放大和旋转等操作的结果图像。编号分别为01001、01002、01003（不同图像集中编号可能有所不同）。

2、 请根据程序提示输入被比较的源图像和用于比较的图像集所在目录，（请使用相对路径）。本实验提供了一个具有一千张图像的图像集以供比较，相对路径是".\static\pics"。图像集目录下有对原图进行缩小放大和旋转等操作的结果图像。如图：



```
similarity5
C:\Users\Cassini\anaconda3\python.exe D:/Files/STUDY/DAM/similarity5.py
Welcome to image similarity comparing program...
please input the address of the image to compare with...
.\static\pic\001.png
please input the address of the folder to scan...
.\static\pics
please choose the method to compute image similarity...
1.Color Moment(颜色矩)(RGB 颜色空间)
2.Color Moment(颜色矩)(HSV 颜色空间)
3.Color Moment(颜色矩)(LAB 颜色空间)
4.Cosine Similarity(余弦相似度)
5.Structural Similarity(结构相似度)(SSIM)
6.Histogram Similarity(通过直方图计算)
7.Average Hash 均值哈希相似度
8.Difference Hash 差异哈希相似度
9.Perceptual Hash 感知哈希相似度
```


3、选择使用的图像相似度计算方法，程序中一共提供了九种方法。（如上图）

4、运行后程序会输出相似度最高的三张图像的路径以及他们的相似度，并将源图像和相似度最高的三张图像以及他们的相似度展示在一个新打开的窗口中。结果展示如下图：

```
.\static\pics\01001.png
1.0
.\static\pics\01002.png
1.0
.\static\pics\01003.png
0.9375

Process finished with exit code 0
```

