多媒体技术基础第二次实验报告

计 15 班 申喆 2011011313

一、实验任务

For each combination:

Task1: for one query image

Return an ordered list of images (from smallest distance to largest distance)

Calculate the precision of first 30 returned images

Task2: for all query images

Based on task 1, calculate the average precision over all query images

二、代码结构及算法

1、Distance metrics

本次实验中采用了四种距离度量方法，对于向量 $P=(p_1,\ldots,p_n)$以及 $Q=(q_1,\ldots q_n)$，计算方法分别如下所示(下列公式中若分母为 0 则当前 i 值不参与运算)：

- Euclidean(L2) 对应代码中的 `double dis_L2(double[] a, double[] b, int len)` 函数；

$$D(P,Q)=\sqrt{\sum_i (p_i-q_i)^2}$$

- Histogram Intersection(HI) 对应代码中的 `double dis_HI(double[] a, double[] b, int len)` 函数(由于 ppt 中给出的函数可以理解为两个直方图相交的面积，因此两图的距离算法应该为 1 减去所得值)；

$$D(P,Q)=1-\frac{\sum_i \min(p_i,\ q_i)}{\sum_j q_j}$$

- Bhattacharyya (Bh) 对应代码中的 `double dis_bh(double[] a, double[] b, int len)` 函数；

$$D(P,Q)=\sqrt{1-\sum_i \sqrt{p_i q_i}}$$

- Chi-Square(Ch) 对应代码中的 `dis_ch(double[] a, double[] b, int len)` 函数；

$$D(P,Q) = \sum_i \frac{(p_i - q_i)^2}{p_i + q_i}$$

2、获得图片直方图向量

这部分对应于代码中的 `double[] getBins(String path, int flag)` 函数；

对于图像中的每个像素点，通过系统函数获得每个像素点的 r、 g、 b 的值，分别根据 16bins 和 128bins 不同的比例获得当前直方图所对应的 index 的值，并将直方图向量的这个 index 处的值加一，即可获得一幅图像的直方图向量。

3、主函数算法流程

首先读入 QueryImages.txt 以及 AllImages.txt 获得待查询的图像的名称和所有的图像的名称信息。对于每个 Query Image，遍历所有的 Image 并依据四种距离计算方法分别计算它们两两之间的距离。

最后分别依据不同的距离从小到大排序，将这部分的结果存在 ./Image_Retrieval/ans_file 目录下的 16bins_orderedLists 目录下以及 128bins_orderedLists 目录下的对应的距离计算方法的文件夹下，其中前 30 个筛选出来的结果存在./Image_Retrieval/ans_file 目录下的 16bins_selectedLists 以及 128bins_selectedLIsts 目录下的对应的距离计算方法文件夹下。

其中前 30 个的平均准确度以及算法的的平均准确度存在 ./Image_Retrieval/ans_fil 目录下的 16bins_selectedLists 目录下或者 128bins_selectedLists 目录下对应的距离计算方法文件夹下的 res_overall.txt 文件中。

三、实验环境

编程语言：Java

操作系统：Windows 32 位

运行方法：直接用 Eclipse 运行该工程即可。当然也可以新建立工程添加 exp2.java 源代码文件之后运行。

四、实验结果及数据分析

1、实验结果

实验结果数据见目录./ Image_Retrieval/ans_file 目录下

Table1

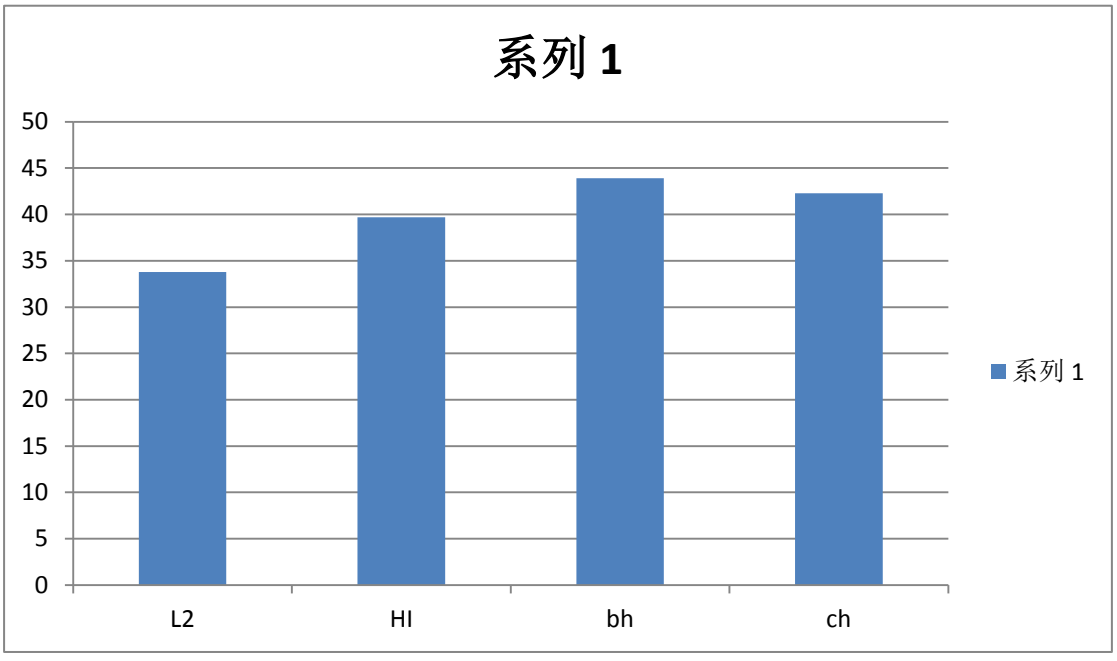
分区方法				距离度量方法			
bins	R	G	B	L2	HI	bh	ch
16	2	4	2	29.75%	33.70%	38.77%	36.79%
128	4	8	4	37.78%	45.68%	49.01%	47.78%

由 Table1 中每种度量方法的精度计算得到每种距离度量方法的平均值，可以得到如下表格

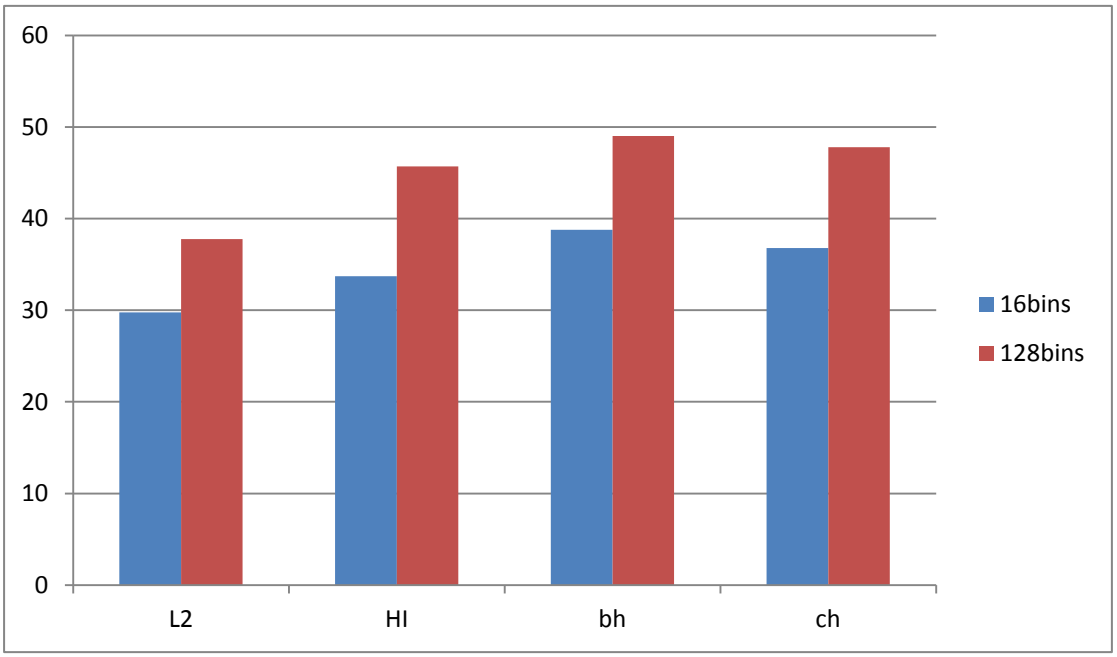
Table2

距离度量方法	L2	HI	bh	ch
平均精度	33.77%	39.69%	43.89%	42.29%

由每个距离的精确度，可得如下直方图



由同一距离度量方法不同的 bins 可得如下直方图



2、实验结果分析

由该表可以看出，若以精确度作为度量好坏的标准的话，距离度量方法从优到差依次为 bh、ch、HI、L2。而对于同一种度量方法，128bins 的准确度要高于 16bins 的准确度。

故如果对精确度要求比较高的话，最好采用 bh 即 Bhattacharyya 距离度量方法并且采用较高的 bins。

3、错误样例分析与提高方法

观察结果，可以看出有一些 query Image(128bins 情况下)四种距离度量方法的准确度都很低，以 elephantx/511.jpg 为例，该图片如下所示

其中三个算法判断除其本身之外距离最近的图片都为 horses/718.jpg，该图片

如下图所示

对于这种，图片主色调占了很大部分且二者主色调相同的图片，这个个人理解是是直方图度量方法本身的缺陷，只有注意到颜色信息而忽略了纹理等一些可能更为重要的信息，所以导致了结果准确度的降低。

另一个四种度量方法准确度都较低的图片为 beach/112.jpg，该图片如下所示

四种算法判断与其最相近的图片(除该图片本身)均为 mountains/805.jpg，该图片如下所示

判断与之次相近的图片有 tribe/25.jpg 和 buses/312.jpg，两幅图片分别如下所示

至此，图片与 query Image 肉眼判断差距就很大了。

因此综合以上两例，要增加准确度，首先可以对图片分区处理，增加纹理等信息。其次对于某个主色调图片很大部分的情况，会导致 L2 算法使图片距离减小或者增大，对于这种情况以及综合考虑各个算法的适用情况(下一小节中有讨论)，可以通过实验对于某些不同种类的图片对每种算法确定一个权值，使用加权之后的结果作为最终的度量结果。

4、拓展联想与分析

实验中，L2 度量方法主要是计算颜色直方图欧式空间的距离向量，而这种计算平方后再开平方的算法很有可能由于某一维的误差而导致二者距离的增大，甚至有可能大于多维均不一样的图片，例如{3,3,3}如果是待查询图片的话，那么{5,1,6}相较于{3,3, 9}可能后者和原图更相似一些，但是该算法则会判断前者更为相似，从而导致了误差的出现。

而对于 HI 算法，则是将所有颜色相同点的个数求和，其效果是可以得到两个图片中相同颜色点的个数，并以此来判断两个图片的距离。这个算法对于某些需要找色调比较相似的情景可能更为适用。

而 ch 算法则可以看做是综合了 HI 和 L2 算法，结果相较于两个单独的算法也显得更好一些。

最后，对于精确度最高的 bh 算法，则可以看做是比较两幅图片的主色调的相似程度。这种算法对于图片有较为明显的主色的情况可能会表现的更好一些。

而对于不同数目的 bin，直观上的理解就是在 RGB 比例相同的情况下，数目越大则图片直方图更加细腻，信息量越大，结果自然也就更加准确了一些。

五、实验总结与收获

通过本次实验，更加具体的了解了图片颜色直方图对于图片基于内容检索的作用，对当前生活中的一些搜索引擎的以图搜图的方法有了初步的认知。其中在 HI 算法中，开始计算准确率只有 2%，经过与同学讨论发现了 HI 算法与图片之间距离一个取反的关系，及时改正过来得到了正确的答案。

最后感谢张正同学给我讲这次实验的算法，感谢老师上课的倾力讲授，感谢助教辛苦的批改作业并找出我们的不足让我们可以继续提高！