

多媒体技术基础 第二次实验

Content Based Image Retrieval

计 24 2012011335 柯均洁

一、 实验内容:

1. 对每张输入图片，建立它的 Histogram (16 bins 或 128 bins)
2. 对于每个 query 的图片，比较其和数据库中图片的距离，输出 30 个距离最小的图片
3. 根据图片的分类，计算这 30 个图片分类的准确率
4. 计算总的准确率
5. 实现图形界面的 Content Based Image Retrieval

二、 实现方法

● 计算 Histogram

通过 `double[] calImageBins(String filename)` 函数来计算直方图。

16 bins 和 128 bins 的分割方式为：

- 16 bins: R : G : B = 2 : 4 : 2
- 128 bins: R : G : B = 4 : 8 : 4

首先获得每个像素点的 r、g、b 值，然后根据分割方式来获得直方图对应的 index，并将 index 处数量加 1

● Distance metrics

对于向量 $P=(p_1, p_2, \dots, p_n)$ 和查询向量 $Q=(q_1, q_2, \dots, q_n)$ 。四种距离度量方法如下：

1. Euclidean (L2)

对应代码中的 `double calL2(double[] p, double[] q)` 函数，距离度量方法为：

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

2. Histogram Intersection (HI)

对应代码中的 `double calHI(double[] p, double[] q)` 函数，距离度量方法为：

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

实验 ppt 中所给的公式是两个直方图相交的面积，因此两图的 HI 距离应该取反（用 1 减去所得值）

3. Bhattacharyya (Bh)

对应代码中的 `double calBh(double[] p, double[] q)` 函数，距离度量方法为：

$$D(P, Q) = \sqrt{1 - \sum_i \sqrt{p_i q_i}} \quad (\text{for percentile histogram})$$

4. 扩展距离函数：Chi-Square (Chi)

对应代码中的 `double calChi(double[] p, double[] q)` 函数，距离度量方法为：

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

- 主函数运行流程：

1. 命令行程序：

首先根据 bins 的数目来计算 AllImages.txt 的 Histogram。然后逐行读入 QueryImages.txt，对于每个 Query Image，计算其 Histogram 并算出其与 AllImages.txt 中所有 Image 的四种距离，并排序。将前 30 个结果放入 16bins/ (或 128bins/) 中对应的距离方法的文件夹下。根据前 30 个分类结果计算准确度，并存入 16bins/ (或 128bins/) 中对应的距离方法的文件夹下的 res_overall.txt 文件中

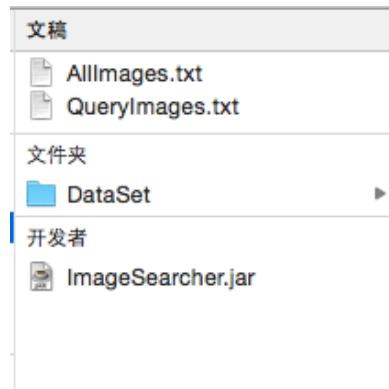
2. 图形界面程序：

用户选择 Query Image 及距离函数，在点击 run (16 bins) 或 run (128 bins) 后，程序读入 AllImages.txt 中的所有图像并建立 Histogram 数据库（只有第一次运行时要建立，接下来的运行可直接使用），然后读入 Query Image，建立其 Histogram 并计算相应的距离。将排序后的前 20 个图像展示在右侧，并显示其距离

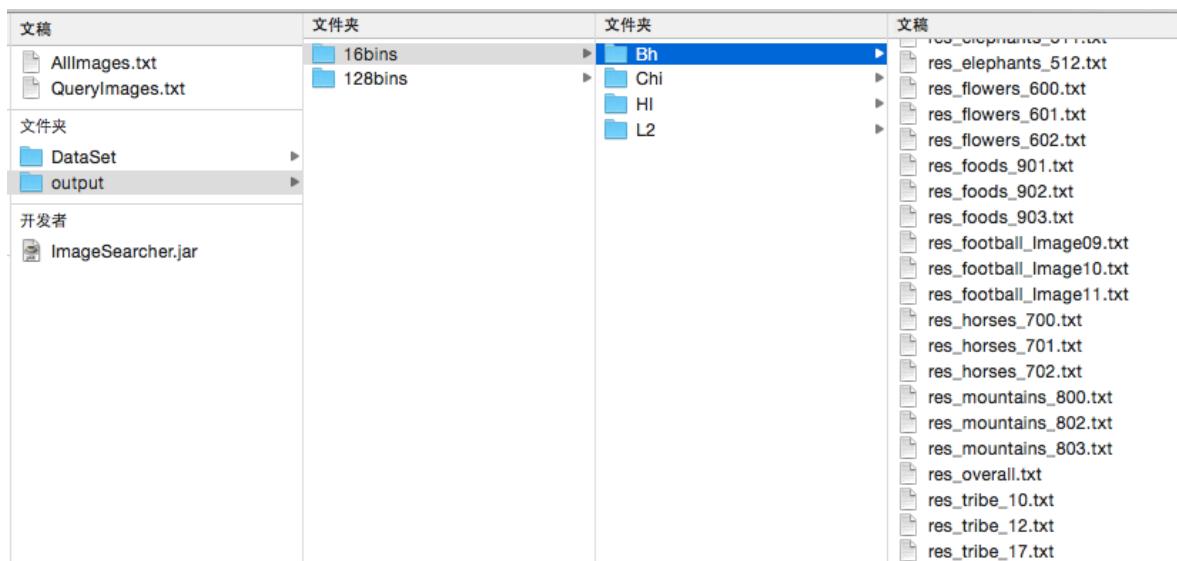
三、运行方法

- 命令行程序运行方法:

1. 将 ImageSearcher.jar 与 DataSet 文件夹、AllImages.txt、QueryImages.txt 放在同一个目录下

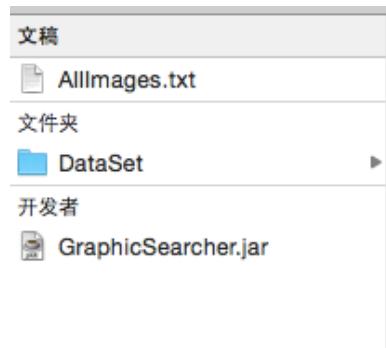


2. 双击 ImageSearcher.jar 运行，就会运行所有 QueryImage.txt 中的查询，并分别应用 16bins 和 128bins 及对应的四种距离函数，生成 output 文件夹。结果按照 16bins/128bins 以及 L2/HI/Bh/Chi 四种距离函数进行分类输出。每个结果文件中输出了前 30 个结果的名称及距离，res_overall.txt 中计算了每个 query 的准确度及该方法总的准确度

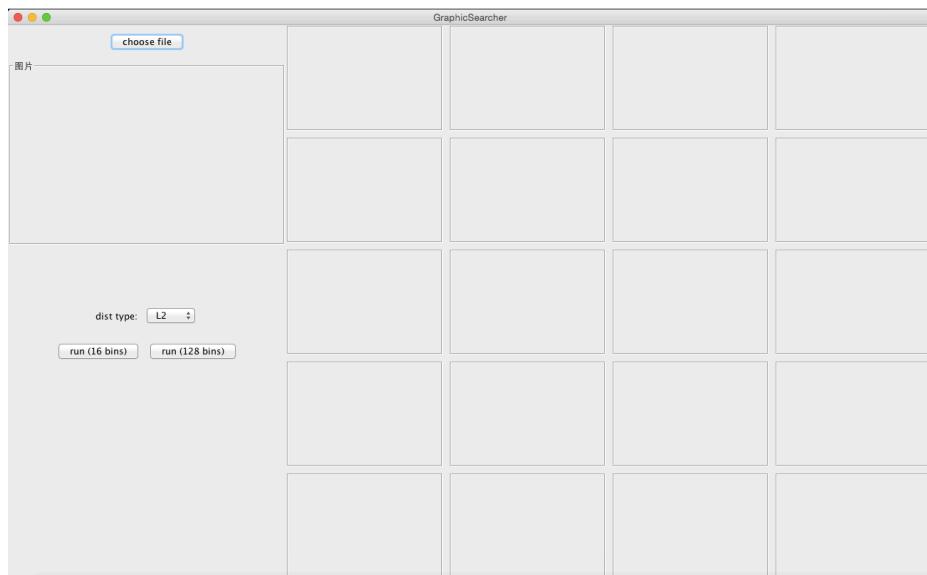


- 图形界面运行方法:

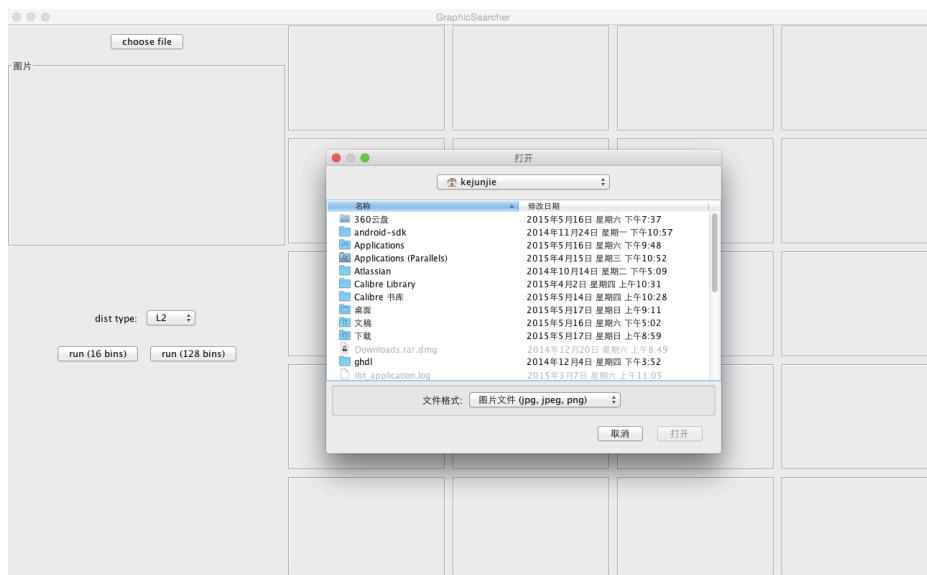
1. 将 GraphicSearcher.jar 和 DataSet 文件夹、AllImages.txt 放在同一个目录下



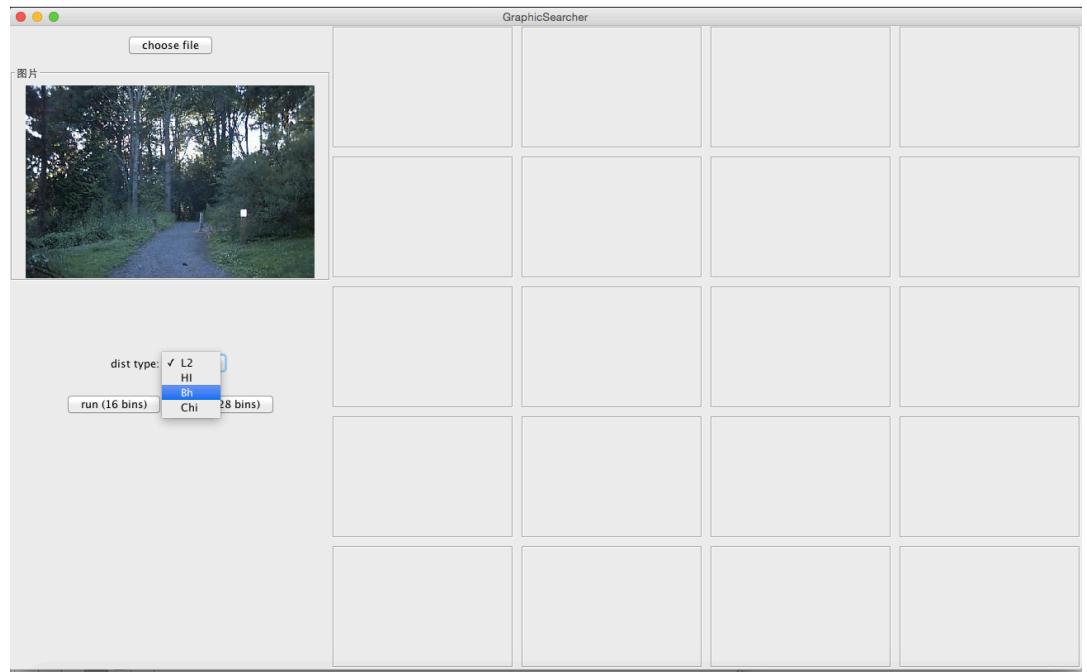
2. 双击运行 GraphicSearcher.jar，进入主界面



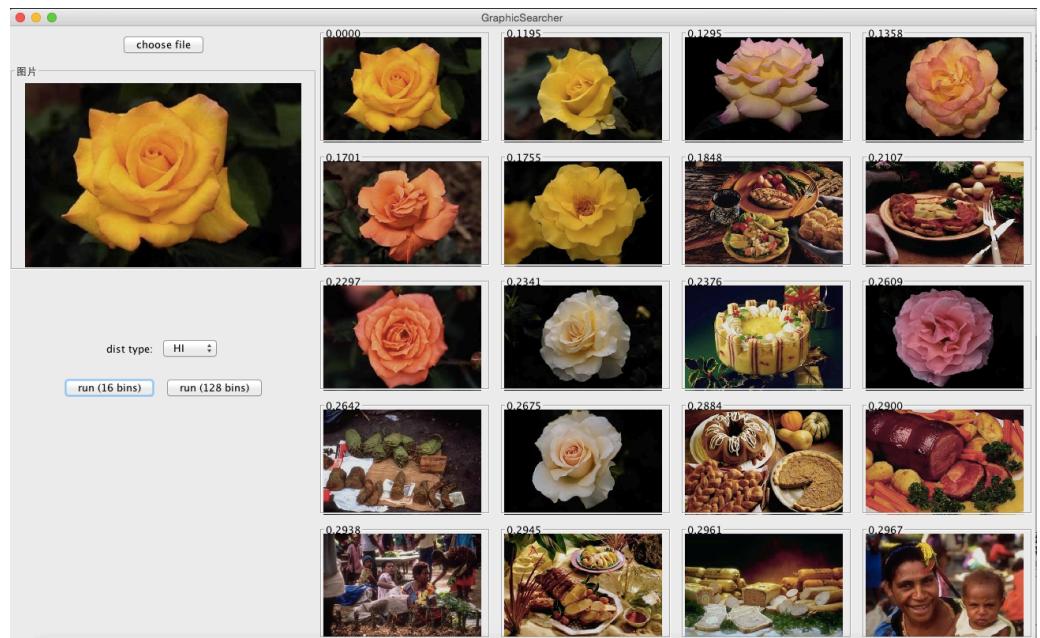
3. 点击左上角的 choose file 选择一张 query 图片

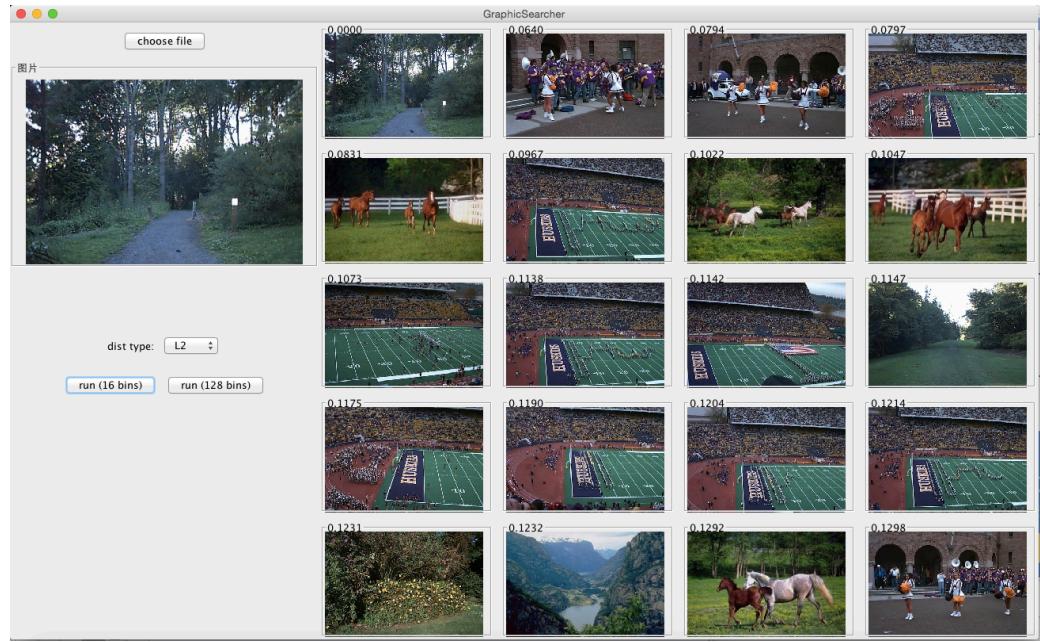


4. 选择 dist type



5. 点击左下方 run (16 bins)或 run (128 bins)按钮进行图片搜索。右边显示前 20 个结果的图片，每个图片的左上角会显示对应的距离（第一次运行的时候会建立 16bins 或 128bins 的数据库，因而要等几秒，第二次搜索会很快）。

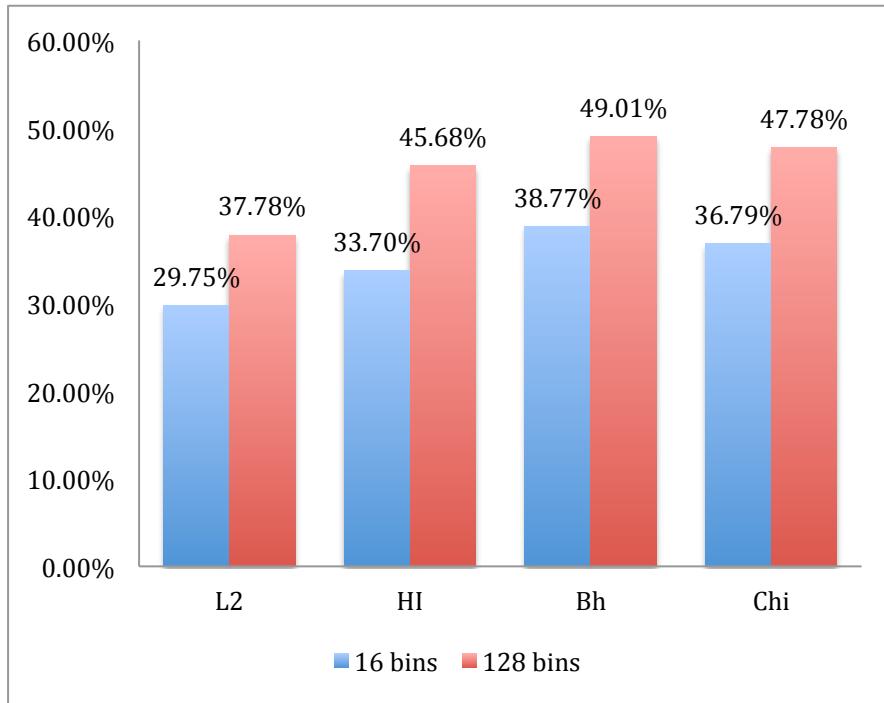




四、实验结果及数据分析

● 实验结果

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



● 实验结果分析

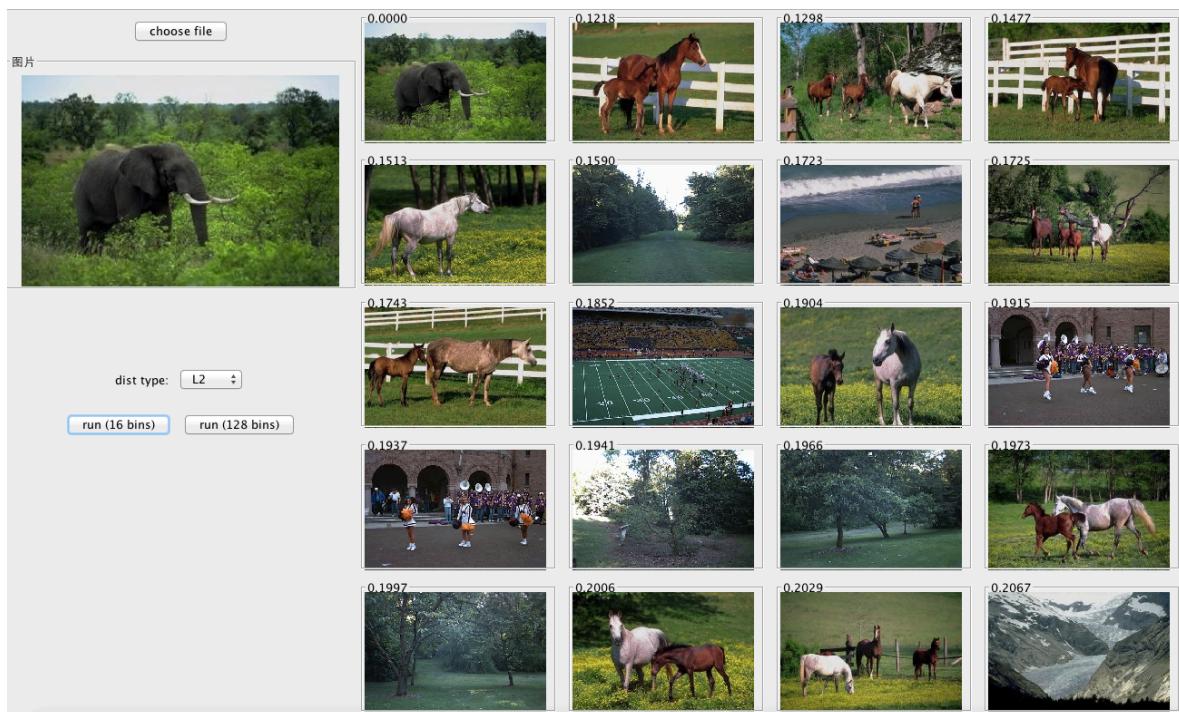
由上图可以看出，对于同一种距离度量方法，128bins 的精确度都要高于 16bins。因为数目越多的直方图更加细腻，信息量也越大，从而更加精细地反映了图片的颜色分布，所以自然准确率更高。

而对于同一种 bins 划分方法，距离度量方法的准确度从低到高依次为 $Bh > Chi > HI > L2$ 。

$L2$ 方法之所以效果不太好是因为欧式距离并不能将颜色分布考虑在内，因而当某一维相差较大的时候会导致 $L2$ 距离很大，而某些图片 R、G、B 三维可能都不太相似，但是总体的距离反而小，因此 $L2$ 距离小图片有时候主色调相差很大。 HI 算法可以得到两个图片中相同颜色的个数，更加适用于寻找色调相近的图片。 Chi 算法综合了 $L2$ 和 HI 算法，结果比后两者都好一些。精度最高的 Bh 算法是比较两个图片的主色调，对于图片主色调明显的图片分类更准确。

● 改进方法讨论

从 res_overall.txt 中可以看出，有些图片的分类准确率非常低，例如 elephants/511.jpg。从图形界面的应用程序来看，分类结果大多是 house 类。而且基本都是大片的绿色中间有一些灰棕色：



由此可见，Histogram 方法难以区分主色调相同的图片，因为当大片背

景色相似的时候，该算法并不能准确识别物体的形态、纹理。因此，改进的时候需要加入空间、纹理的相关信息，

五、实验总结

通过这次实验，我更加深入地学习了用 Histogram 来进行 Content Based Image Retrieval 的方法，对于以图搜图的技术有了初步了解。同时，对于不同的距离度量方法的效果也有了更加直观的认识。最后感谢老师和助教的辛勤工作和热心指导！