基于颜色直方图的图像检索与分类

——————实验报告

苗根<miaogen156@outlook.com>

摘要：

本次实验主要目的是实现基于颜色直方图的图像检索算法，以及验证在分割粒度增加的情况下的匹配度变化，主要思路是对图像的 RGB 三个通道分别进行分割，并形成特征向量，在此基础上比较，不同的向量之间的相关度，以此来判断对应的两幅图像的相似程度。

关键字： **颜色直方图　匹配**

目录

一、简介：

从 20 世纪 70 年代开始，有关图像检索的研究就已开始，当时主要是基于文本的图像检索技术（Text-based Image Retrieval，简称 TBIR），利用文本描述的方式描述图像的特征，如绘画作品的作者、年代、流派、尺寸等。到 90 年代以后，出现了对图像的内容语义，如图像的颜色、纹理、布局等进行分析和检索的图像检索技术，即基于内容的图像检索（Content-based Image Retrieval，简称 CBIR）技术。CBIR 属于基于内容检索（Content-based Retrieval，简称 CBR）的一种，CBR 中还包括对动态视频、音频等其它形式多媒体信息的检索技术。

颜色直方图是在许多图像检索系统中被广泛采用的颜色特征。它所描述的是不同色彩在整幅图像中所占的比例，而并不关心每种色彩所处的空间位置，即无法描述图像中的对象或物体。直方图描述了该图像中关于颜色的数量特征，可以反映图像颜色的统计分布和基本色调，直方图只包含了该图像中某一颜色值出现的频数，而丢失了某象素所在的空间位置信息，任一幅图像都能唯一的给出一幅与它对应的直方图，但不同的图像可能有相同的颜色分布，从而就具有相同的直方图，因此直方图与图像是一对多的关系;如将图像划分为若干个子区域，所有子区域的直方图之和等于全图直方图。

颜色直方图特别适于描述那些难以进行自动分割的图像和不需要考虑空间位置的图像[1]。

二、实现原理：

这次实验的主要原理是受《相似图片搜索的原理（二）》[2]启发来实现的。

每一张图都有一张自己的颜色特征，常用的颜色特征包括颜色直方图，颜色聚合向量，颜色矩等，我们这里使用颜色直方图，如果两个图的颜色直方图是非常相似的，那我们可以认为这两张图的图像内容有很大概率也是相似的。基于这样的想法，我们可以提取每一张图的颜色直方图，并计算他们的向量相似度，绝对值越靠近 1 则认为他们越相似。

提取颜色直方图的时候，考虑到 RGB 三个通道各有 256 个颜色，直接生成颜色向量需要 1600 万的向量长，计算过于困难，同时因为要求是相似图片检索，所以需要一定的模糊度，所以在实际操作的时候，我们选择一个 4 分割 (即将每个通道分成等比例分成四个部分) 为基础分割度，并在此基础之上，逐步的增加分割度，并比较不同的分割度下图片匹配的准确性。

三、具体实现：

3．1 提取颜色直方图：

我们可以将每一个通道的颜色值 0 ~ 255 分成四个区：0 ~ 63 为第 0 区，64 ~ 127 为第 1 区，128 ~ 191 为第 2 区，192 ~ 255 为第 3 区。这意味着红绿蓝分别有 4 个区，总共可以构成 64 种组合（4 的 3 次方），然后处理每一个像素，找到他们所在的分区，并将对应的分区的计数值加 1。处理完所有的像素后，我们便得到了一个 64 元素的一维向量。

下表是某张图像进行 4 分割以后的得到的像素数量统计图，将像素数量那一列进行排开，得到一维向量。

红	绿	蓝	像素数量	红	绿	蓝	像素数量	红	绿	蓝	像素数量	红	绿	蓝	像素数量
0	0	0	7414	1	0	0	891	2	0	0	1146	3	0	0	11
0	0	1	230	1	0	1	13	2	0	1	0	3	0	1	0
0	0	2	0	1	0	2	0	2	0	2	0	3	0	2	0
0	0	3	0	1	0	3	0	2	0	3	0	3	0	3	0
0	1	0	8	1	1	0	592	2	1	0	2552	3	1	0	856
0	1	1	372	1	1	1	3462	2	1	1	9040	3	1	1	1376
0	1	2	88	1	1	2	355	2	1	2	47	3	1	2	0
0	1	3	0	1	1	3	0	2	1	3	0	3	1	3	0
0	2	0	0	1	2	0	0	2	2	0	0	3	2	0	0
0	2	1	0	1	2	1	101	2	2	1	8808	3	2	1	3650
0	2	2	10	1	2	2	882	2	2	2	53110	3	2	2	6260
0	2	3	1	1	2	3	16	2	2	3	11053	3	2	3	109
0	3	0	0	1	3	0	0	2	3	0	0	3	3	0	0
0	3	1	0	1	3	1	0	2	3	1	0	3	3	1	0
0	3	2	0	1	3	2	0	2	3	2	170	3	3	2	3415
0	3	3	0	1	3	3	0	2	3	3	17533	3	3	3	53929

某图片 4 分割得到的像素数量统计表

在对样本中所有的图片都进行特征向量提取，并存入数据库以后，接下来要解决的就是如何检索的问题，也就是如何求得两个图片之间的相似度/距离的问题。

距离的度量通常有欧式距离、皮尔逊相关系数及巴氏距离，考虑到欧式距离在处理部分情况比如当[5,5]与[1,1]应该相似的，但是欧式距离发现它们距离会很大，就排除了欧式距离。

3.2 相关性度量

在皮尔逊相关系数和巴氏距离之间进行选择的时候，网上的说法是，巴氏距离要比皮尔逊更加好使，对于这种说法，我针对样本 91.jpg 做 了一个小测试如下表所示：

	巴氏距离和皮尔逊系数在直方图中准确度的简单比较					
	基于巴氏距离得到 匹配结果			**基于皮尔逊系数得到的结果**		
匹配结果 匹配等级	**文件名**	**相关度**	**主观判断 (0-5 由低到高)**	**文件名**	**相关度**	**主观判断 (0-5 由低到高)**
1	494.jpg	0.8988417914900404	0	346.jpg	0.872416904588	0
2	4061.jpg	0.884449126171264	0	1132.jpg	0.870629122065	1
3	42.jpg	0.8759661046818432	0	9.jpg	0.866207932823	4
4	1132.jpg	0.8746897942674687	1	19.jpg	0.864562272581	4
5	957.jpg	0.8631720506528121	0	969.jpg	0.863552133819	0
6	346.jpg	0.8617886389091934	0	204.jpg	0.860292209974	1
7	9605.jpg	0.8607366646438902	0	1.jpg	0.856267420149	2
8	1112.jpg	0.8549438642623272	0	17.jpg	0.855682404155	2
9	5059.jpg	0.8393759216049118	0	4061.jpg	0.850827671298	0
10	9698.jpg	0.8390714133507485	0	957.jpg	0.848671820511	0
准确度: $\frac{\Sigma(\frac{\text{单个准确度}}{5})}{10}$			0.02			0.28

从表中可以看出巴氏距离并没有皮尔逊系数的匹配度好，巴氏距离的匹配度只有 2%，所以我们在这次实验中选择皮尔逊距离作为我们的相关度判断函数。

皮尔逊相关系数的计算公式（下页）：

$$\rho_{X,Y}=\frac{\operatorname{cov}(X,Y)}{\sigma_X\sigma_Y}=\frac{E((X-\mu_X)(Y-\mu_Y))}{\sigma_X\sigma_Y}=\frac{E(XY)-E(X)E(Y)}{\sqrt{E(X^2)-E^2(X)}\sqrt{E(Y^2)-E^2(Y)}}$$

$$\rho_{X,Y}=\frac{N\sum XY-\sum X\sum Y}{\sqrt{N\sum X^2-(\sum X)^2}\sqrt{N\sum Y^2-(\sum Y)^2}}$$

$$\rho_{X,Y}=\frac{\sum XY-\frac{\sum X\sum Y}{N}}{\sqrt{(\sum X^2-\frac{(\sum X)^2}{N})(\sum Y^2-\frac{(\sum Y)^2}{N})}}$$

1 皮尔逊系数计算

皮尔森相关系数的取值总是在-1.0 到 1.0 之间，接近 0 的变量被成为无相关性，接近 1 或者-1 被称为具有强相关性。它是衡量线性关联性的程度，p 的一个几何解释是其代表两个变量的取值根据均值集中后构成的向量之间夹角的余弦。皮尔逊相关系数是余弦相似度在维度值缺失情况下的一种改进。

四、**实验结果与分析：**

对于一个检索系统来讲，召回率和精度是评判检索质量好坏的两个指标，我们这次实验也采用这样的评判标准，但因为我们样本数量较大(9908 张图)，召回率的计算较困难，所以我们仅以精度来判断匹配的准确性。

针对目标匹配图像 91.jpg, 125.jpg, 535.jpg, 774.jpg,812.jpg 得出以下结果

4.1 基 4 分割检测结果：

如下表所示

	基 4 分割检测结果								
	91.jpg			**125.jpg**			**535.jpg**		
匹配结果 / 匹配等级	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**
1	1132.jpg	0.991987	1	5378.jpg	0.865520	0	569.jpg	0.944171	2
2	9604.jpg	0.985670	0	2400.jpg	0.863302	0	1300.jpg	0.942470	2
3	42.jpg	0.978210	1	6694.jpg	0.863062	0	1389.jpg	0.939378	2

4	1.jpg	0.977152	0	2999.jpg	0.862763	0	2547.jpg	0.933466	2
5	981.jpg	0.971187	0	5474.jpg	0.862462	0	423.jpg	0.933466	2
6	4490.jpg	0.966621	0	6698.jpg	0.862287	0	4306.jpg	0.932598	2
7	2219.jpg	0.966619	0	3840.jpg	0.861965	0	1440.jpg	0.932141	2
8	1112.jpg	0.966588	0	9502.jpg	0.861750	0	817.jpg	0.926309	1
9	59.jpg	0.965543	1	131.jpg	0.861414	0	502.jpg	0.924706	1
10	969.jpg	0.965368	0	2071.jpg	0.861308	0	4681.jpg	0.923475	1
准确度: $\frac{\Sigma(\frac{\text{单个准确度}}{5})}{10}$			0.0			0			0.34

接上表:

	基 4 分割检测结果								
	774.jpg			**812.jpg**					
匹配结果 / 匹配等级	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**
1	504.jpg	0.965105	2	896.jpg	0.946570	5			
2	5018.jpg	0.957575	0	883.jpg	0.931533	3			
3	5558.jpg	0.955958	0	1639.jpg	0.925502	2			
4	4500.jpg	0.953096	0	806.jpg	0.923955	3			
5	1098.jpg	0.950950	0	1100.jpg	0.917520	0			
6	3821.jpg	0.948983	0	845.jpg	0.917290	1			
7	9220.jpg	0.946923	1	825.jpg	0.916983	2			
8	5598.jpg	0.942420	0	865.jpg	0.910792	0			
9	5560.jpg	0.941230	1	844.jpg	0.910762	1			
10	1332.jpg	0.941024	0	833.jpg	0.897286	4			

准确度: $\frac{\Sigma(\frac{\text{单个准确度}}{5})}{10}$			0.08			0.66			

分析：基于 4 分割的检索，由于区间分的过大，导致检索准确度很低，大部分都是低于 10%的精度，对于最后一个样本 812.jpg(见下图)，由于该样本的色差较大，对比很明显，即使是大区间划分也没有影响匹配精度，也说明了颜色直方图的检索如果区间划分太大，会导致检索过于模糊。

2.样本 812.jpg 的部分匹配

4.2 **基 11 分割检测结果:**

如下表:

	基 11 分割检测结果								
	91.jpg			**125.jpg**			**535.jpg**		
匹配结果 / 匹配等级	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**
1	346.jpg	0.872416	0	5287.jpg	0.768221	0	4472.jpg	0.786940	3
2	1132.jpg	0.870629	0	138.jpg	0.763516	3	1300.jpg	0.749113	2
3	9.jpg	0.866207	0	6576.jpg	0.753140	0	1565.jpg	0.747077	2
4	19.jpg	0.864562	0	5775.jpg	0.752779	0	328.jpg	0.730812	2

5	969.jpg	0.863552	0	7682.jpg	0.747812	0	569.jpg	0.714223	2
6	204.jpg	0.860292	0	150.jpg	0.747503	3	4672.jpg	0.696107	0
7	1.jpg	0.856267	0	9333.jpg	0.746793	0	4518.jpg	0.689528	0
8	17.jpg	0.855682	1	6085.jpg	0.746524	0	834.jpg	0.683204	0
9	4061.jpg	0.850827	0	2244.jpg	0.744962	0	888.jpg	0.672266	0
10	957.jpg	0.848671	0	6835.jpg	0.744339	0	4461.jpg	0.670987	0
准确度: $\frac{\Sigma(\frac{\text{单个准确度}}{5})}{10}$			0.02			0.12			0.22

接上表

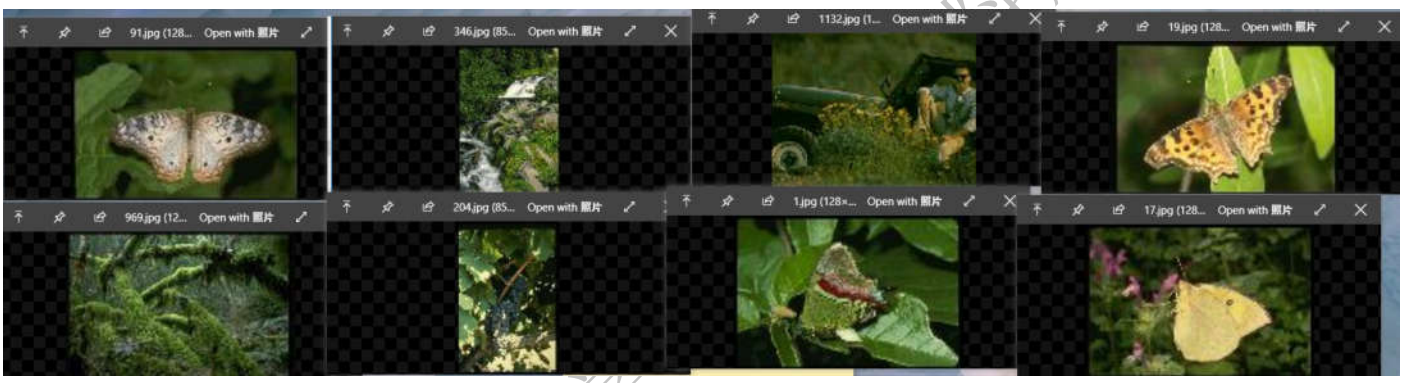
	基 11 分割检测结果								
	774.jpg			**812.jpg**					
匹配结果 / 匹配等级	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**	**文件名**	**相关度**	**主观判断（0-5 由低到高）**
1	1302.jpg	0.851660	2	883.jpg	0.915775	5			
2	349.jpg	0.806900	0	845.jpg	0.903593	3			
3	8632.jpg	0.792550	0	833.jpg	0.899636	2			
4	6066.jpg	0.780857	0	896.jpg	0.892920	3			
5	1866.jpg	0.780154	0	3303.jpg	0.890680	0			
6	1861.jpg	0.777771	0	827.jpg	0.886457	1			
7	367.jpg	0.777286	0	865.jpg	0.881019	2			
8	1855.jpg	0.765106	2	3292.jpg	0.876490	0			
9	6062.jpg	0.763139	0	804.jpg	0.849557	1			
10	6057.jpg	0.762292	0	869.jpg	0.846617	4			
准确度:			0.08			0.42			

$\Sigma(\frac{\text{单个准确度}}{5})$ / 10									

分析：这次我们使用 11 分割，总颜色为 11^3=1331 种，是基 4 分割的 20 倍，虽然效果依然不尽如人意，但是相对基 4 来讲，还是有一些进步的，比如对 91.jpg 的检索，基 4 时检索除了颜色相近以外，没有匹配出相应的轮廓相近的图片，但在基 11 时检索出了 19.jpg(下图右上角)，两幅图有些微的色差以及亮度的差异，但还是很相近的。

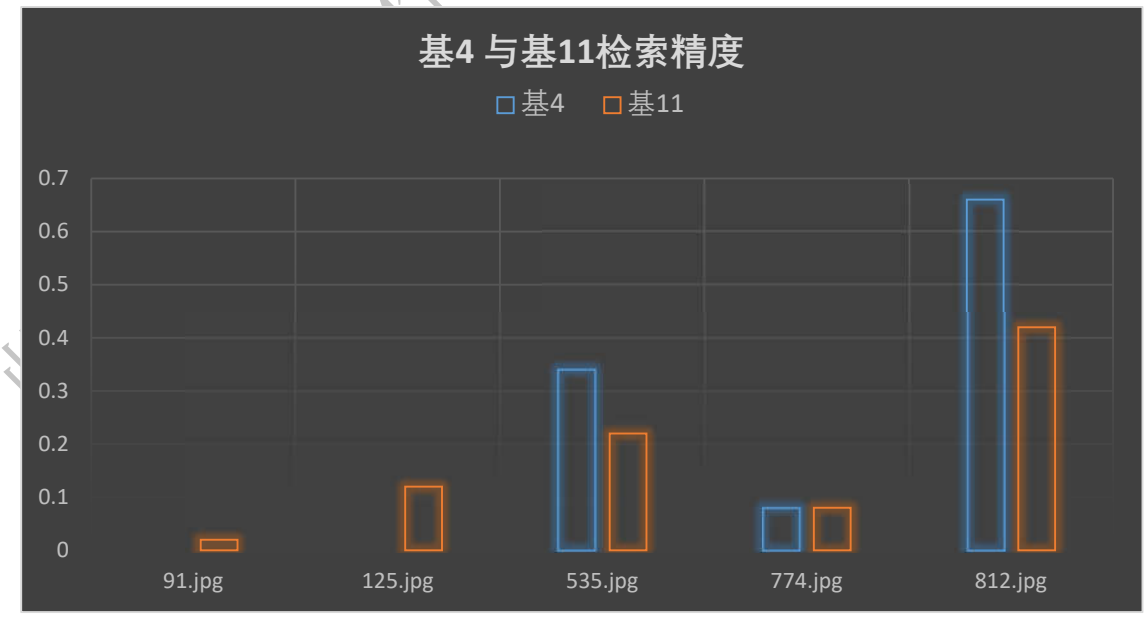
但效果也并非都是在变好，比如对 535.jpg 和 812.jpg 检索 的结果从人的主观感觉还不如基 4 时检索。在讨论基 4 检索时，对于样本 812.jpg 我们曾说过，因为 812.jpg 的色彩对比度比较强所以在大区间分割时，仍能检索，但是在区间变小了以后，容易掺入噪声的影响。

就整体而言，色彩大致都是匹配的到的。如下图，色调都是暗绿色的。



3.对 91.jpg 的基 11 分割的检索

基 4 和基 11 的检索准确度直方图：



从直方图我们可以清楚的看到，对于图 91.jpg 和 774.jpg 的检索精度在两种基本区间下都比较低，而对 535.jpg 和 812.jpg 的检索精度则相对较高。基 11 分割没有像我们想象的那样造成匹配精度的提升。

五、结论

颜色直方图作为一种常用的图像检索的方法，在检索图像时受到区间划分、相关度算法以及图像本身的限制。在我们的实验结果中，不管是哪一种分割方式，虽然不能匹配到很准确的图像，但都可以匹配得到图像色彩相近的图像。

单纯的依靠颜色直方图的检索得到的结果并不如人意，如果加上针对图像本身属性的附加处理对其进行改进，效果应该会好很多。

还有一个结论在上面实验中，并没有体现出来：效率，在进行匹配的时候，随着维数的增加，依次增加到 4，5，7, 11, 20，即便是进行 4 分割，在样本库数量为 9908 份的时候，遍历所有值仍然需要 4s 左右的时间，20 分割的时候，居然长达 5 分钟，所以仅基于颜色直方图的检索算法，并不能用来构建图像搜索引擎。

六、改进

这次实验有三点需要改进的地方:

1. 匹配的算法，颜色直方图的检索效果并不好，后续的实验中可以尝试一些 精度更高的算法。
2. 数据的数量，这次在比较不同分割区间的检索精度时，使用的数据集较小，不够充分。
3. 检索的效率，由于是依次遍历进行检索，检索效率很差，后续的实验中会尝试使用机器学习的聚类算法，进行优化。

针对颜色直方图匹配效果不好的问题，我尝试了 pHash 的检索算法，因为本次实验主要要求是基于颜色直方图的检索，所以不再给出 pHash 的介绍以及算法实现。以下是基于 pHash 检索算法的针对新建样本 pikaqiu1.png 的检索结果：

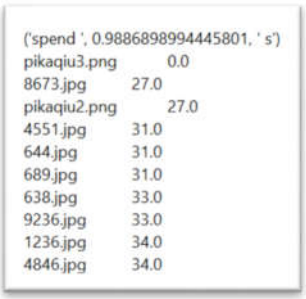


Figure 4 对 pikaqiu1.png 的 phash 检索结果

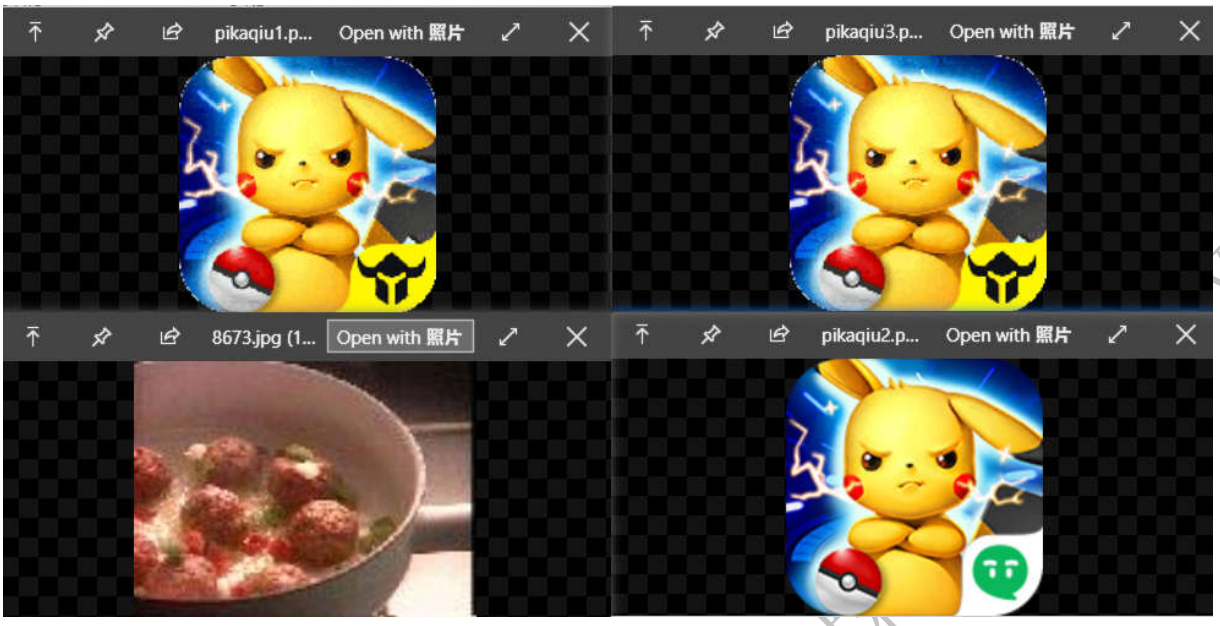


Figure 5 对 pikaqiu1.png 的 phash 检索原图与前三个匹配结果

第一张图是原图，第二章是原图的 copy，第三，四张是匹配的前两个，从结果中我们也可以看出，pHash 快是快，但是准确度上的确实有所不足，Figure4 中也可以看出第三张图和第四张图的 256 位 hash 跟原图的汉明距都是 27，但主观上二者完全不同。

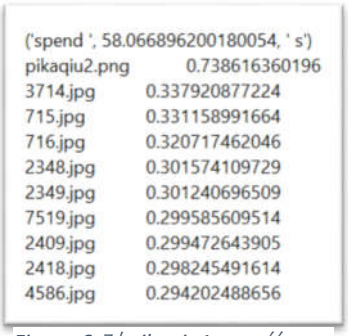


Figure 6 对 pikaqiu1.png 的 histogram 检索结果

左图是颜色直方图的检索结果，不像 pHash，这里第一匹配和第二匹配的差距很明显，由此可得不同的检索算法有自己不同的适用场合，需要根据实际场景需要来使用最佳检索算法。

针对查询效率太低的问题，在原来的基础上我修改了匹配算法，改成 annoy 聚类优化算法，改进之后查询速度相当的快，完全没有延迟的感觉，但稍有不足的是，annoy 在 github 上的库并不支持皮尔逊距离。

七、参考文献

[1] 百度百科，颜色直方图

[2] 阮一峰的网络日志，相似图片搜索的原理（二）

[3] 维基百科，皮尔逊积矩相关系数

[4] 伯乐在线，用 Python 和 OpenCV 创建一个图片搜索引擎的完整指南

[5] CSDN 论坛，基于颜色直方图的搜索

[6] 简书，图像特征提取

[7] CSDN 论坛，皮尔森相关系数及 python 计算代码

[8] 程序园，图像相似性搜索的原理

[9] 掘金，老司机带你检测相似图片

附：本次实验的代码已经上传至 https://github.com/miaogen123/imageMatch