编程作业二

王书琦 520030910357

部分关键过程参数与代码在报告中呈现。源代码参见 `homework.py` ，代码与其他过程参数输出见 `homework.ipynb` .

实现思路

SIFT: 从训练图像中提取特征

1. 提取特征点并利用 `.extend()` 整合
2. 统计特征点最少的类别的特征点个数 此处经提醒，原代码中

```
kp_list = vec_dict[i]['kp'] = sorted((vec_dict[i]['kp']),
                                     key=lambda x: x.response,
                                     reverse=True)
```

应改为

```
    temp = [(vec_dict[i]['kp'][j], vec_dict[i]['des'][j]) for j in
range(len(vec_dict[i]['kp'])) ]
    temp = sorted(temp, key=lambda x: x[0].response, reverse=True)
    for j in range(len(temp)):
        vec_dict[i]['kp'][j] = temp[j][0]
        vec_dict[i]['des'][j] = temp[j][1]
```

以在排序中同步改变 `kp` 与 `des`。(代码参考：陈琦 520030910151)

3. 向最少的个数看齐，整齐化特征字典

```
for i in range(1, 102):
    vec_list.extend(vec_dict[i]['des'][0:bneck_value])
```

Kmeans: 对特征点聚类

1. 选取合适的k值 (`N_clusters` = k)

在 `Kmeans` 类中，`sklearn` 提供了参数`inertia` 用来评估簇的个数是否合适，距离越小说明簇分的越好，官方文件注释说明如下：

> _kmeans.py
>
> #inertia_ : float, Sum of squared distances of samples to their closest cluster center

为选取合适的k值，进行测试：

k	inertia	runtime
1	142997.03860247726	1.1s
10	106983.13549884284	17.4s
20	98005.53806102792	42.9s
50	88128.89138771851	2m 1.3s
100	81447.3748741387	4m 41.3s
150	77858.47644890724	7m 4.3s

从上表中可以看出，当 k 下降时，inertia 随之下降，k=150 依然未取到极值，但考虑到训练时间过长，最终选取k=150 完成本实验，为提升效果，可以考虑继续提升k值。实现如下：

```
kmeans = KMeans(n_clusters=N_clusters, random_state=10).fit(vec_list)
```

2. 直方图统计图像中特征点所属聚类中心个数

3. 直方图归一化得到特征向量

此处迭代单一元素`des_vector[j]`进行 `predict()` 会导致运行时间过长，为提升效率，`kmeans.predict.(des_vector)`得到`center`，再对`center`进行统计。

```
hist_vector = np.zeros((num_images, N_clusters))
for i in range(num_images):
    tep = cv2.normalize(data.train_images[i], None, 0, 255,
cv2.NORM_MINMAX).astype('uint8')

    kp_vector, des_vector = sift.detectAndCompute(tep, None)
    des_vector = np.float64(des_vector)
    centers = kmeans.predict(des_vector)
    for j in range(N_clusters):
        hist_vector[i][j] = (centers == j).sum()/len(centers)
```

SVM: 对特征点进行分类

对直方图向量训练出SVM分类器。

```
classifier = svm.SVC(probability=True)
classifier.fit(hist_vector, data.train_lb)
```

处理测试集

与训练集的处理方法相同：

1. 进行数据归一化
2. SIFT寻找特征向量
3. 使用训练好的K-means聚类模型预测
4. 使用SVM分类
5. 输出预测准确率。

最终结果

```
accuracy 0.3713556851311953
```

总用时约19分钟。

可进一步提高准确率的方式：

1. 继续提升k值直至 `inertia` 收敛
2. 提升K-means训练迭代轮数（目前使用默认值300轮，但未至收敛）
3. 使用高性能分类器，替换线性的SVM分类器。