CV_HW2

1. TODO1

利用 SIFT 计算图片的特征，其中 kp、des 分别表示特征点列表与特征点信息描述向量。

根据 SIFT 计算出的数据，填写 vec_dict 的对应项即可。

```
vec_dict[data.train_lb[i]]['kp'].extend(kp_vector)
vec_dict[data.train_lb[i]]['des'].extend(
    des_vector)
```

2. TODO2

这里我们要为每个类别选择同样多的特征点用于聚类，特征点个数 bneck_value，因此对于 des 进行截取，并把第 i 类的特征点信息加入到 vec_list 中即可。

这里我其实有一点疑惑，因为 kp 和 des 一一对应，但是在助教给出的代码中，只对 kp 进行了排序，但是没有对其对应的 des 进行排序，这里我其实有一点疑惑为什么 des 不跟着一起排序，

```
tmp = vec_dict[i]['des'][0:bneck_value]
vec_list = np.append(vec_list, tmp, axis=0)
```

3. TODO3

使用 kmeans 进行聚类，类的数量我选取了不同的值进行实验。个人的电脑感觉因为内存的问题跑起来很慢，因此我选取了 max_iter=100 进行验证。（默认值为 300 次 kmeans 迭代）

```
kmeans = KMeans(n_clusters=N_clusters,max_iter =
    100, random_state=0).fit(vec_list)
```

4. TODO4 and TODO5

这里两个 TODO 的实现逻辑一致，因为是分别对训练集以及测试集进行操作，操作的步骤都是一样的。

根据聚类结果求出对应图像的特征向量。特征向量的求解是统计每张图像中的特征点所属聚类中心的个数再归一化即可。

```
# 计算SIFT
kp_vector, des_vector = sift.detectAndCompute(tep,
     None)
# 统计
for j in range(len(kp_vector)):
    hist_test_vector[i][kmeans.predict(np.float64(
     des_vector[j]).reshape(1, -1))] += 1
# 归一化
hist_test_vector[i] =hist_test_vector[i] / len(
    kp_vector)
```

5. 效果和分析

在写代码的过程中我尝试的很多种的写法，从许多同学那里学到了一些思考的方式和想法，最终的代码虽然精简但确实中间走了一些弯路，虽然代码工作量不大但确实让我学习到了不少。

结果如下：

使用 1000 个 clusters 和 100 次的 kmeans 迭代，最终效果为 0.36042274052478135。

个人感觉效果偏低，原因我个人认为有如下几点：首先是 kmeans 迭代次数 100 也许偏低，适当增大让算法多运行几轮可能会更加收敛。第二点是 SVM 分类器本身的局限性，支持向量机是最基本最基本的模型，本任务种的特征带着一部分的非线性信息，SVM 在这个上面做不到很好的效果。第三就是聚类的数量偏多，导致聚出了一些非必要的聚类中心，因为数据本身的影响，聚类中心的格数需要更精细的调整。

对于为什么只排了 kp 没有排 des 我其实还是不太想的明白。