图像分类系统设计文档

汤锦禾 109班 2021213356

1. 算法整体流程

数据预处理->SIFT提取特征->对特征进行K-means词袋表示+IDF加权->SVM分类训练->评估

1.1 数据预处理

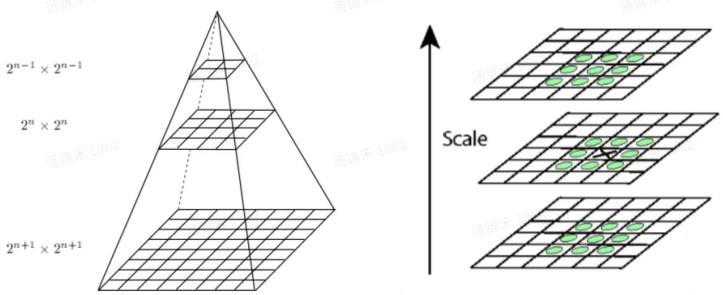
15-Scene数据由15个类别构成，每个类别前150张构成训练集，其余构成测试集。测试集每个类别的图片数量不同。处理时先将数据集划分为了训练集和测试集两个文件夹。

1.2 SIFT特征提取

提取图像中的尺度、旋转不变的极值点作为关键特征。

1.2.1 尺度空间极值检测（Scale-space Extrema Detection）：

- 构建尺度空间：通过逐渐增大图像的模糊程度（使用高斯模糊）来创建一个尺度空间，以模拟图像在不同尺度下的观察结果。
- 在尺度空间中寻找极值点：在每个尺度空间的图像中查找对比度高的关键点（潜在的特征点），这些点在尺度和空间位置上都是局部极值。



1.2.2 关键点定位（Keypoint Localization）：

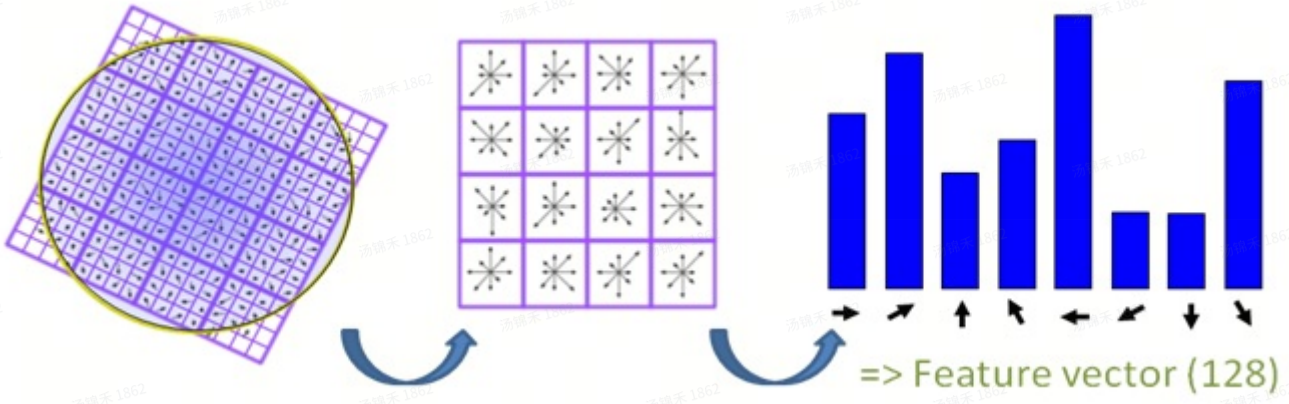
根据Taylor展开的DoG响应去除边缘响应较强的关键点，只保留角点作为稳定关键点。

1.2.3 方向赋值（Orientation Assignment）：

为每个关键点赋予方向，来增强匹配的鲁棒性。这样做可以确保即使图像旋转，相同的关键点也能在图像间被匹配到，因为特征描述符将通过相对于主方向的相对方向来计算，并不是基于图像的绝对方向。这一步通过计算关键点周围邻域的梯度直方图来完成。首先计算其周围区域内像素的梯度大小和方向。接着，在关键点位置构建局部梯度方向直方图，统计不同方向梯度的累积量。直方图中的峰值对应于关键点的主方向。

1.2.4 关键点描述（Keypoint Descriptor）：

在关键点的周围选取一个邻域，并基于该邻域内的梯度方向和大小，生成描述符。描述符是一个高维向量，通常为128维（16个邻域*8个方向），向量中的每个元素是局部梯度直方图的统计结果。



1.3 词袋表示

1.3.1 K-means聚类

根据SIFT得到的descriptors特征向量寻找k个聚类中心，形成视觉词汇的词汇表。

1.3.2 选择K个初始中心：

- 随机选择K个数据点作为初始的聚类中心，来选择初始中心。

1. 分配步骤（Assignment Step）：
   - 对于数据集中的每一个点，计算它与所有K个中心的距离，并将它分配到最近的聚类中心所形成的聚类中。
2. 更新步骤（Update Step）：
   - 对于每个聚类，重新计算聚类中心，通常是取聚类内所有点的算术平均值。
3. 重复迭代：
   - 重复分配和更新步骤，直至满足停止条件。停止条件可以是聚类中心的变化非常小（小于某个阈值），或者是达到了预设的迭代次数。
4. 收敛：
   - 当聚类中心不再发生变化或变化很小，说明算法已收敛，此时可以停止迭代。

1.3.3 IDF

对于每一张图片，用直方图（histogram）统计descriptor所属视觉词汇聚类，记录每个视觉词汇在全体图片当中的出现次数，计算每个视觉词汇的IDF（Inverse Document Frequency）来衡量词汇的重要性。经常出现在各个图片中的词要相对来说更难提供有用信息。其中 n_t 代表含有词t的文档（图片）数量， N 代表文档总数

$$\mathrm{IDF}(t) = \log\left(\frac{N}{n_t + 1}\right) \tag{1}$$

1.4 SVM分类

通过SVM对每张图得到的视觉词汇向量进行有监督分类。

2. 函数功能和参数说明

2.1 sift_feature

功能：提取图像的sift特征，返回 `descriptors` 矩阵。

入参： `img_paths` 列表

- 创建一个SIFT特征提取器，遍历每个图像路径，对每张图像执行以下操作：
  - 读取和调整图像大小。
  - 使用SIFT检测关键点。
  - 计算关键点的描述符。
- 将每张图像的路径和关键点描述符存储在 `des_list` 中。
- 从 `des_list` 的第一个元素开始，将所有图像的描述符垂直堆叠成一个大的描述符矩阵 `descriptors`。
- 返回 `des_list` 和 `descriptors`。

2.2 bof_feature

功能：提取BoF特征，返回直方图、IDF和聚类中心矩阵。

入参：des_list, descriptors, k

- `des_list`：列表，每个元素是一个元组 `（image_path，descriptors）`
- `descriptors`：每一行为一个 `descriptor` 向量的矩阵
- `voc`：聚类中心矩阵，每一行代表一个聚类中心，共k个中心，列的数量为 `descriptor` 向量的维度（128）
- `im_features`：特征直方图矩阵，一行代表一张图的直方图向量，列数为k，大小 `len(image_paths)*k`
- `vq`（vector quantization）：scipy的矢量量化函数，
  - 接受两个参数：观察点集合（SIFT描述符）、码本（聚类中心）
  - 返回两个数组：每个观察点的码本索引、每个观察点到最近聚类中心的距离

```
words, distance = vq(des_list[i][1], voc)
```

- nbr_occcurences（number of occurence）：对im_features中每个feature转换为布尔，代表一个词汇在图片中是否出现。沿列求和得到含有每个词t的图片（文档）数量，大小1*k。
- idf：计算得到大小1*k的idf数组，用于对im_feature进行加权

```
nbr_occurences = np.sum((im_features > 0) * 1, axis=0)
idf = np.array(np.log((1.0 * len(image_paths) + 1) / (1.0 * nbr_occurences + 1)))
```

2.3 svm.SVC

功能：创建一个线性核支持向量机

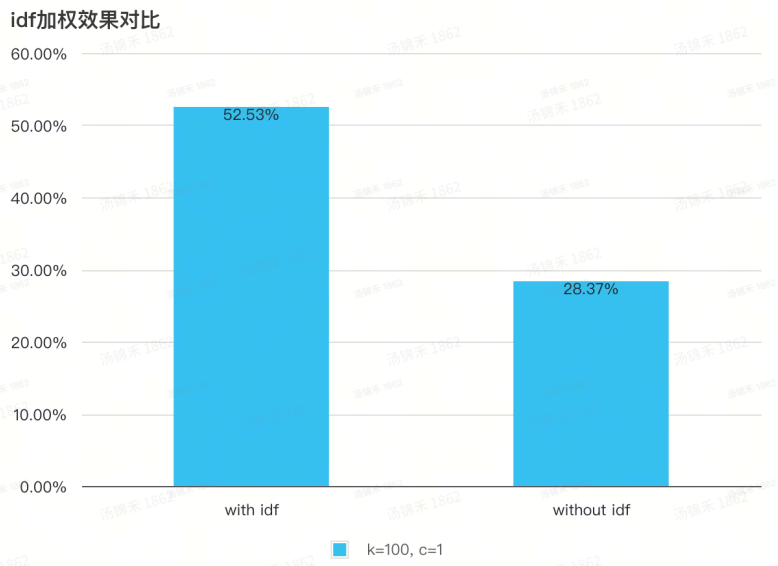
入参：C（正则化参数），C越大模型对分类误差的容忍度越低（更弱的正则化）

```
clf = svm.SVC(C=50, kernel='linear')
clf.fit(im_features_weighted, np.array(classes_names))
```

3. 参数对结果影响分析

3.1 IDF加权

对比了是否对词袋直方图向量进行IDF加权对最终分类结果的影响。针对k=100，C=1，发现加IDF会对最终效果有**显著提升**。

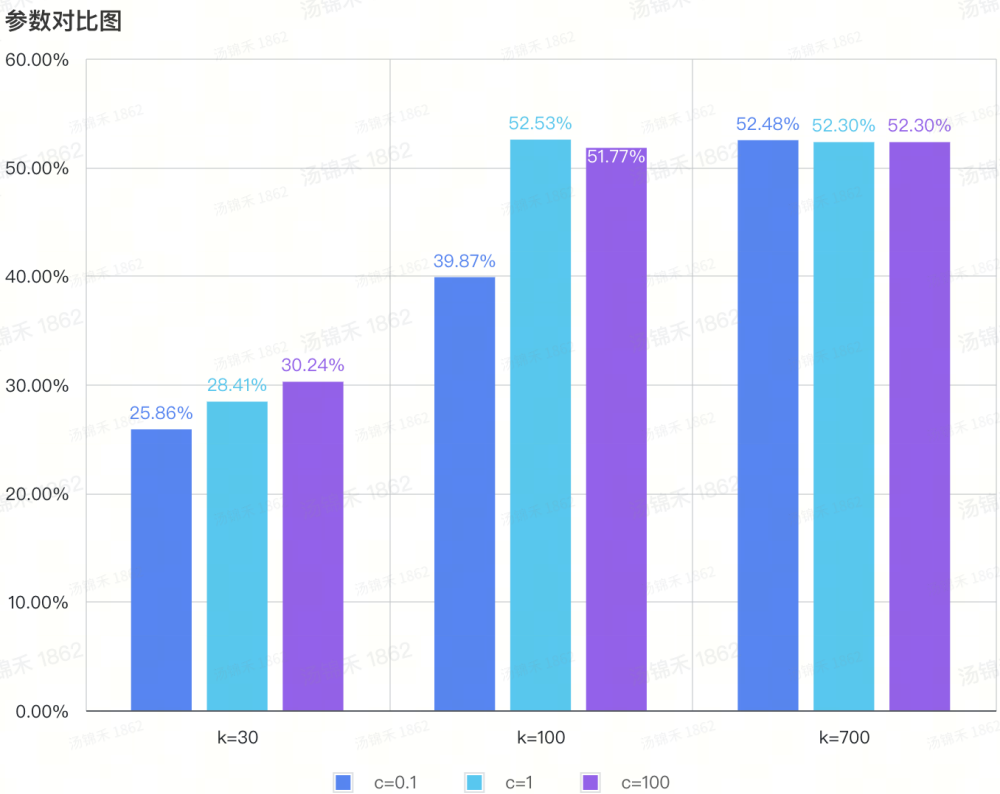


3.2 词袋表示聚类中心数量k

对比了k=30、100、700，发现k=100和700在效果上近似，k=30由于特征不足，效果相对较差（见如下参数对比图）。由于k=100和k=700在效果上**几乎接近**，而k=700聚类**耗时**（1347s）要**显著高于**k=100（143s），因此采用k=100即可。

3.3 SVM正则化参数C

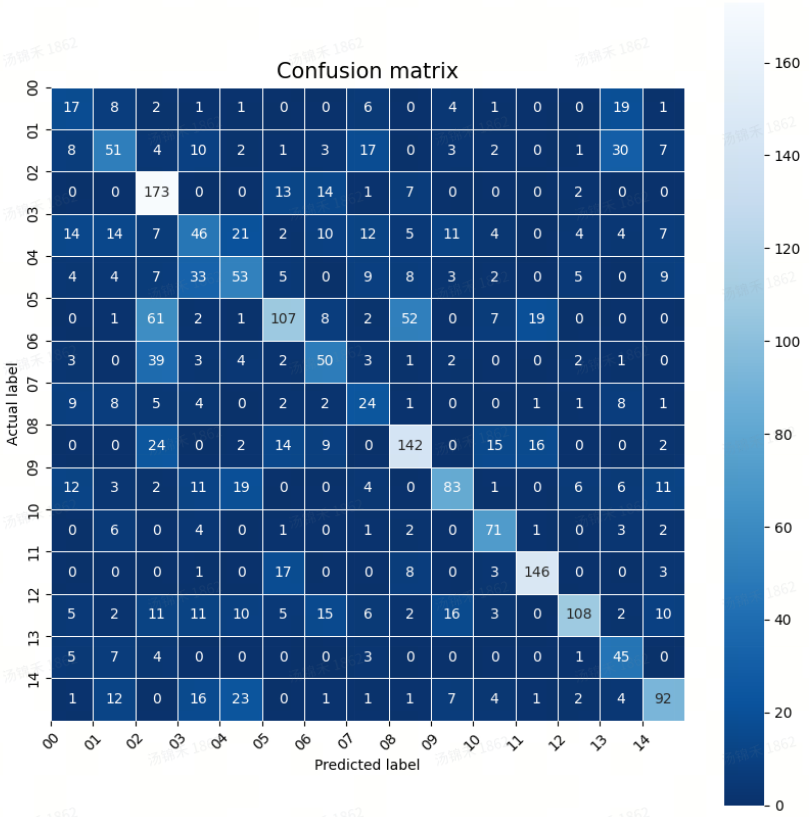
对比了C=0.1，1，100，发现在k值较小时（30、100），C对于SVM分类效果的影响更大。当k上升为700，C的取值对分类效果几乎没有影响。（加IDF）



4. 最终结果

k=100，C=3，使用idf对词袋向量进行加权后得到的混淆矩阵：

准确率52.53%


Confusion matrix
Actual label
Predicted label
00 01 02 03 04 05 06 07 08 09 10 11 12 13 14
17 8 2 1 1 0 0 6 0 4 1 0 0 19 1
8 51 4 10 2 1 3 17 0 3 2 0 1 30 7
0 0 173 0 0 13 14 1 7 0 0 0 2 0 0
14 14 7 46 21 2 10 12 5 11 4 0 4 4 7
4 4 7 33 53 5 0 9 8 3 2 0 5 0 9
0 1 61 2 1 107 8 2 52 0 7 19 0 0 0
3 0 39 3 4 2 50 3 1 2 0 0 2 1 0
9 8 5 4 0 2 2 24 1 0 0 1 1 8 1
0 0 24 0 2 14 9 0 142 0 15 16 0 0 2
12 3 2 11 19 0 0 4 0 83 1 0 6 6 11
0 6 0 4 0 1 0 1 2 0 71 1 0 3 2
0 0 0 1 0 17 0 0 8 0 3 146 0 0 3
5 2 11 11 10 5 15 6 2 16 3 0 108 2 10
5 7 4 0 0 0 0 3 0 0 0 0 1 45 0
1 12 0 16 23 0 1 1 1 7 4 1 2 4 92
160 140 120 100 80 60 40 20 0