实验设计文档

一. 设计概述

本程序是一个图像分类系统，能够对输入图像进行类别预测。具体而言使用了SVM学习SPM算法生成的特征向量，并对学习结果进行了评估。

SPM即Spatial Pyramid Matching，是一种利用空间金字塔进行图像匹配、识别、分类的算法。SPM是BOF(Bag Of Features)的改进，因为BOF是在整张图像中计算特征点的分布特征，进而生成全局直方图，所以会丢失图像的局部细节信息，无法对图像进行精确地识别。SPM算法克服了BOF的这些缺陷。

1.1 算法流程

1. 提取数据集中的样本，并划分训练集和测试集
2. 提取训练集图片的 SIFT 特征点，并将 SIFT 特征点向量归一化
3. 对所有 SIFT 特征点使用聚类算法分为 n 类
4. 将 n 类特征点的中心点作为视觉词汇，生成词袋（字典）
5. 使用论文中的 SPM 算法生成图片的特征向量
6. 将图片的特征向量作为数据集，使用支持向量机进行训练
7. 对测试集图片做和训练集相同的数据处理操作
8. 使用支持向量机模型对测试集图片的类别进行预测并评估

1.2 最终结果

分类报告如下：

	precision	recall	f1-score	support
CALsuburb	0.78	0.86	0.82	91
MITcoast	0.62	0.71	0.66	210
MITforest	0.76	0.91	0.83	178
MIThighway	0.64	0.55	0.60	110
MITinsidecity	0.63	0.50	0.56	158
MITmountain	0.56	0.58	0.57	224
MITopencountry	0.66	0.52	0.58	260
MITstreet	0.60	0.79	0.68	142
MITtallbuilding	0.75	0.63	0.68	206
PARoffice	0.50	0.48	0.49	65
bedroom	0.30	0.36	0.33	66
industrial	0.43	0.32	0.37	161
kitchen	0.24	0.43	0.31	60
livingroom	0.55	0.45	0.50	139
store	0.58	0.62	0.60	165
accuracy			0.60	2235
macro avg	0.57	0.58	0.57	2235
weighted avg	0.60	0.60	0.59	2235

混淆矩阵如下：

```
[[ 78   0   6   0   0   1   0   0   0   1   0   2   1   1   1]
 [  1 149   0   6   0  12  33   0   7   0   2   0   0   0   0]
 [  1   0 162   0   0  10   2   0   0   0   0   0   0   0   3]
 [  0  29   0  61   1   6   3   4   1   0   1   2   2   0   0]
 [  0   2   0   2  79   0   0  28   5   1   1  10  15   3  12]
 [  8  18  19   5   0 131  21   3   1   0   1   2   1   1  13]
 [  1  29  11  11   0  59 134   3   2   0   2   5   0   0   3]
 [  1   0   0   1   5   2   1 112   1   0   3   3   2   2   9]
 [  1   4   1   0  14   5   6   5 130   2   4  21   1   5   7]
 [  0   2   0   0   1   0   0   0   0  31   9   2  15   5   0]
 [  0   2   1   1   2   1   0   0   5   5  24   6   8  10   1]
 [  2   5   0   6  12   7   4  12  14   5  10  52   9   8  15]
 [  0   2   0   1   3   0   0   1   1   4   8   1  26  12   1]
 [  1   0   0   1   2   0   0   4   1  13  14   8  24  63   8]
 [  6   0  12   0   7   0   0  15   6   0   0   7   6   4 102]]
```

二. 核心函数说明

2.1 数据预处理

- 功能：提取数据集中的样本，并划分训练集和测试集
- 输入：数据库路径
- 输出：训练路径，训练标签，测试路径，测试标签

```python
def load_data(path):
    categories = os.listdir(path)
    train_label = []
    test_label = []
    train_paths = []
    test_paths = []

    for category in categories:
        img_paths = glob.glob(path + category + '/*.jpg')
        train_paths.extend(img_paths[:150])
        test_paths.extend(img_paths[150:])

        train_label.extend([category] * 150)
        test_label.extend([category] * (len(img_paths) - 150))

    return train_paths, train_label, test_paths, test_label
```

- 功能：提取训练集图片的 SIFT 特征点，并将 SIFT 特征点向量归一化
- 输入：输入路径
- 输出：归一化的SIFT描述符

```python
def SIFT(paths):
    sift = cv.SIFT_create()
    descriptors = []
    features = []
    shapes = []
    for img_path in tqdm(paths):
        img = cv.imread(img_path)
        shapes.append(img.shape[:2])

        feature, descriptor = sift.detectAndCompute(img, None)
```

```
            descriptors.append(descriptor)
            features.append(feature)
            descriptors = normalize_desc(descriptors)
    return descriptors, features, shapes
```

- 功能：通过聚类算法，生成词袋模型。也就是对所有 SIFT 特征点使用聚类算法分为 n 类，并将 n 类特征点的中心点作为视觉词汇，生成词袋
- 输入：训练集的特征描述符
- 输出：聚类模型

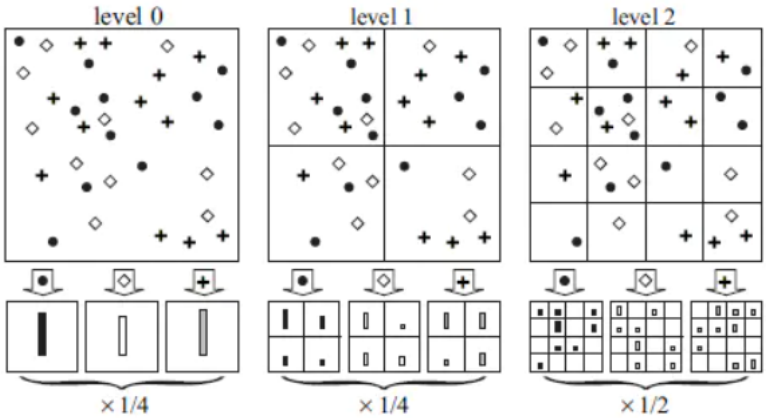
```
def get_BoW(train_desc):
    data = []
    for i in range(len(train_desc)):
        for j in range(len(train_desc[i])):
            data.append(train_desc[i][j])
    data = np.array(data)
    kmeans = MiniBatchKMeans(n_clusters=num_of_bag, random_state=42)
    kmeans.fit(data)
    return kmeans
```

2.2 SPM算法

首先将图像划分为固定大小的块，1*1，2*2，4*4；

然后统计不同level中各个块内的特征对视觉词汇的投票，也就是计算落入每个bins中属于不同类别的word的个数；

最后个将每个level中获得的投票数据合并，并且给每个level赋给相应的权重，连接生成(1, 21*n)的向量，作为一个新的特征。



- 输入：图片的SIFT特征向量，图片的SIFT特征点，图片的形状
- 输出：SPM算法生成的特征向量

```
def SPM(descs, kps, shape):
    """
    使用SPM算法，统计不同尺度下的直方图
    :param descs:图片的SIFT特征向量
    :param kps: 图片的SIFT特征点
    :param shape: 图片的形状
    :return: SPM算法生成的特征向量
```

```
    """
    center = kmeans.cluster_centers_

    # 划分3个尺度
    level2 = np.zeros((16, num_of_bag))
    level1 = np.zeros((4, num_of_bag))
    level0 = np.zeros((1, num_of_bag))

    # 图片分格
    step_x = math.ceil(shape[1] / 4)
    step_y = math.ceil(shape[0] / 4)

    for i in range(len(descs)):
        desc = descs[i]
        kp = kps[i]
        x, y = kp.pt  # 特征点的坐标

        diff = np.tile(desc, (num_of_bag, 1)) - center
        distance = np.sum(np.square(diff), axis=1)
        index = np.argmin(distance)  # 找到离这个特征点距离最近的词汇中心（即划分此特征点的类别）

        cell = math.floor(x / step_x) + math.floor(y / step_y) * 4  # 特征点所在区域
        level2[cell][index] += 1  # 投票

    for i, idx in enumerate([0, 2, 8, 10]):
        level1[i] = level2[idx] + level2[idx + 1] + level2[idx + 4] + level2[idx + 5]

    for i in range(4):
        level0[0] += level1[i]

    result = np.array([]).reshape(0, num_of_bag)
    # 加权求和
    result = np.append(result, level0 * 0.25, axis=0)
    result = np.append(result, level1 * 0.25, axis=0)
    result = np.append(result, level2 * 0.5, axis=0)
    return result
```

2.3 训练及评估

- 功能：评估实验结果
- 输入：训练得到的模型

```
def evaluate(model):
    pred_labels = model.predict(test_bin)

    report = classification_report(test_label, pred_labels)
    conf_mx = confusion_matrix(test_label, pred_labels)
    print(report)
    print(conf_mx)
```

函数的全流程如下：

```
# 提取数据集中的样本，并划分训练集和测试集
train_paths, train_label, test_paths, test_label = load_data(path)

# 提取训练集图片的 SIFT 特征点，并将 SIFT 特征点向量归一化
print("Extract SIFT features...")
train_desc, train_keypoint, train_shape = SIFT(train_paths)
test_desc, test_keypoint, test_shape = SIFT(test_paths)

# 将所有SIFT特征点使用聚类算法分为n类
# 并将n类特征点的中心点作为视觉词汇，生成词袋
kmeans = get_BoW(train_desc)

# 使用 SPM 算法生成图片的特征向量
print("Executing SPM algorithm...")
train_bin = get_SPM_feature(train_desc, train_keypoint, train_shape)
test_bin = get_SPM_feature(test_desc, test_keypoint, test_shape)

# 将图片的特征向量作为数据集，使用支持向量机算法完成分类任务
svm = SVC(kernel='rbf', C=10, gamma=300)
svm.fit(train_bin, train_label)

# 评估结果
evaluate(svm)
```