**智能数据挖掘大作业**

1. **实验概述**

本次实验使用的是VOC2012数据集，首先从图像中随机采样图像块，然后利用Hog方法提取图像块特征，最后采用余弦相似度和k-means聚类两种方法来挖掘视觉模式。

1. **数据集说明**

本次实验使用VOC2012数据集。VOC2012数据集常用于目标检测、图像分割、网络对比实验和模型效果评价。对于图像分割任务，VOC2012的训练验证集包含了2007-2011年的所有对应图像，包含有2913张图片和6929个目标，测试集只包含了2008-2011年。

由于该数据集多用于目标检测等任务中，因此在本次实验中，仅使用到该数据集中的8类数据。

数据集下载链接：

<http://host.robots.ox.ac.uk/pascal/VOC/voc2012/index.html>

1. **算法模型介绍**
   1. **Hog特征提取**

方向梯度直方图（Histogram of Oriented Gradient, HOG）特征是一种在计算机视觉和图像处理中用来进行物体检测的特征描述子。它通过计算和统计图像局部区域的梯度方向直方图来构成特征，它的主要步骤如下：

1. 灰度化

将图片进行灰度化，滤除无关颜色信息。

1. 颜色空间的标准化

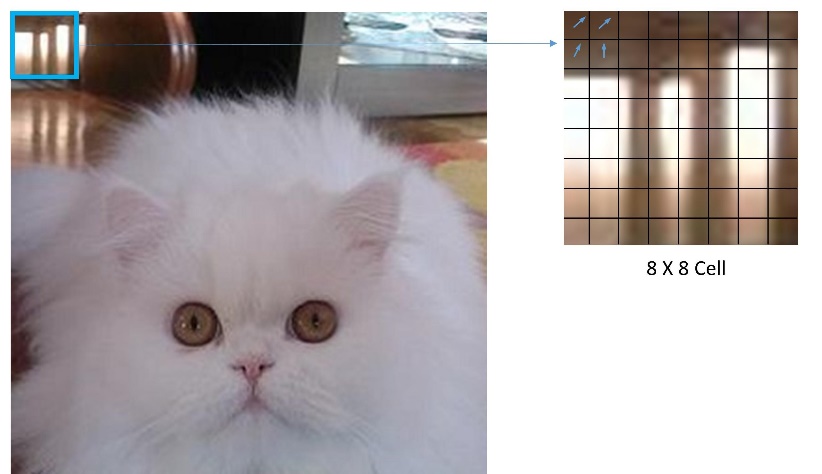
采用Gamma校正法对输入图像进行颜色空间的标准化，可以调节图像的对比度，降低图像局部的阴影和光照变化所造成的影响，同时可以抑制噪音的干扰。

1. 梯度计算

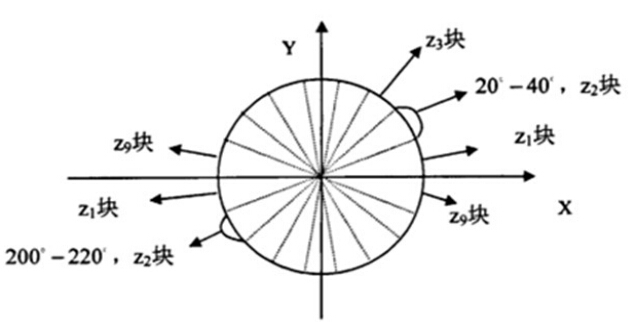
计算图像每个像素的梯度（包括大小和方向），可以捕获轮廓信息，同时进一步弱化光照的干扰。

1. Cell划分

将图像划分成小cells，以8x8大小的Cell为例，如图所示：



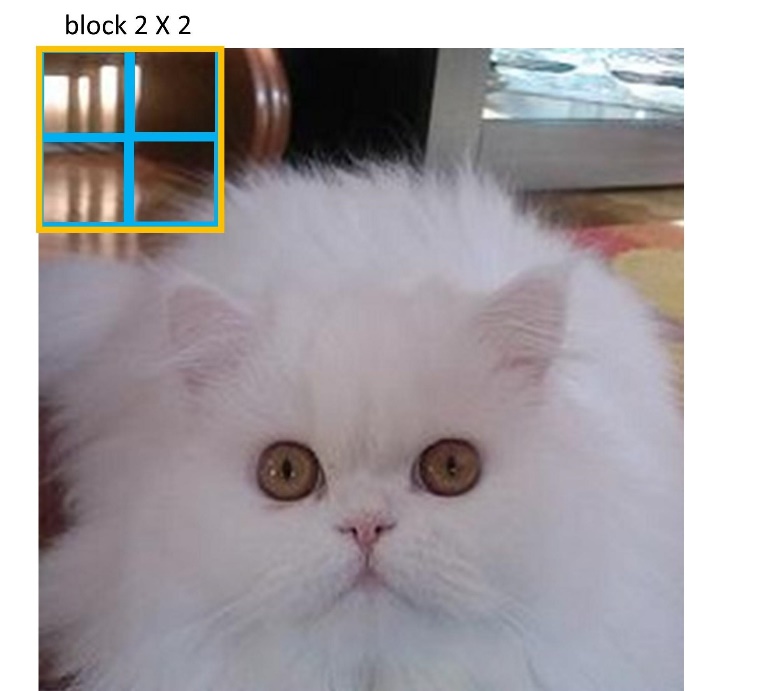
1. 统计每个cell的梯度直方图



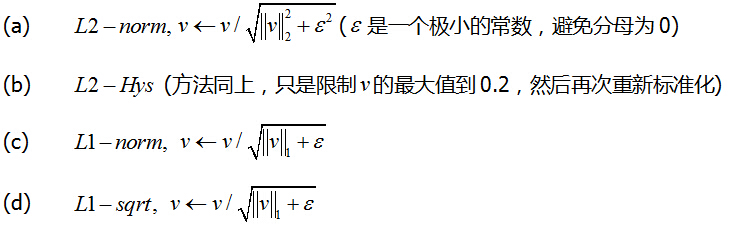
如图所示，将360°划分成18份，每一份角度为20°。同时，不考虑方向大小，即总共为9份角度类别，统计每个类别的频度。

1. 将每几个cell组成一个block

以2x2大小的block为例，如图所示：



图中的蓝色框表示一个cell，黄色框表示一个block，将2x2个cell组成为一个block，同时在每个block内将梯度直方图进行归一化。归一化的方法主要有四种：



原论文中说明L2-Hys的方法效果最好[1]，因此在本实验中，归一化方法也使用L2-Hys。

1. 移动block

将每个block以cell为间隔在水平和垂直两个方向上进行移动，最终串联所有特征向量得到图像块的Hog特征。

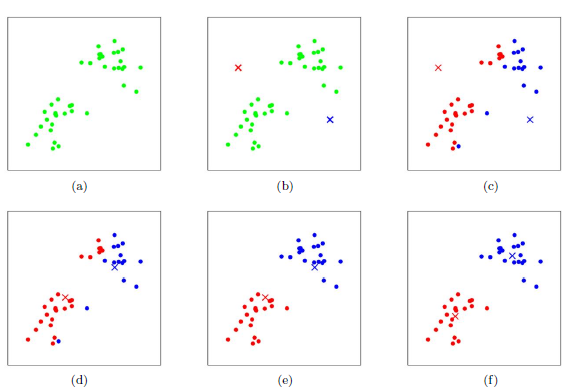
* 1. **余弦相似度**

得到每个图像块的Hog特征之后，通过计算每个图像块特征向量的余弦相似性来进行类别的划分，余弦相似度的计算公式如下：



* 1. **K-means聚类**

得到每个图像块的Hog特征后，还可使用K-means聚类的方式来进行视觉模式的挖掘。K-means聚类的过程如图所示：



首先随机初始化两个点作为聚类中心，计算每个点到聚类中心的距离，并聚类到离该点最近的聚类中去。之后，计算每个聚类中所有点的坐标平均值，并将这个平均值作为新的聚类中心；重复上述两个步骤，直到每个类的聚类中心不再变化，完成聚类。

1. **频繁性和判别性评价指标**
   1. **频繁性评价指标**

若一个图案多次出现在正类图像中，则称其具有频繁性。在本实验中，频繁性的评价指标参考了王倩楠等人[2]的评价标准，定义频繁性公式如下：



式中，N表示某一类样本的总数，表示该类中样本u和样本v的余弦相似度，表示阈值。

* 1. **判别性评价指标**

如果一个模式值出现在正类图像中，而不是在负类图像中，则称其为具有判别性。在本次实验中，使用视觉模式的平均分类精度来定义判别性，公式如下：



式中，M表示样本类别总数，表示类中样本视觉模式平均值m和样本v的余弦相似度，表示阈值。

1. **实验步骤**

**5.1数据集类别划分**

本次实验采用的VOC2012数据集，并没有按类别将图片进行归类。因此首先需要根据类别，将包含该类别的图片进行划分。这里总共使用7个类别：“车、马、猫、狗、鸟、羊、牛”。

执行划分的核心代码如下：

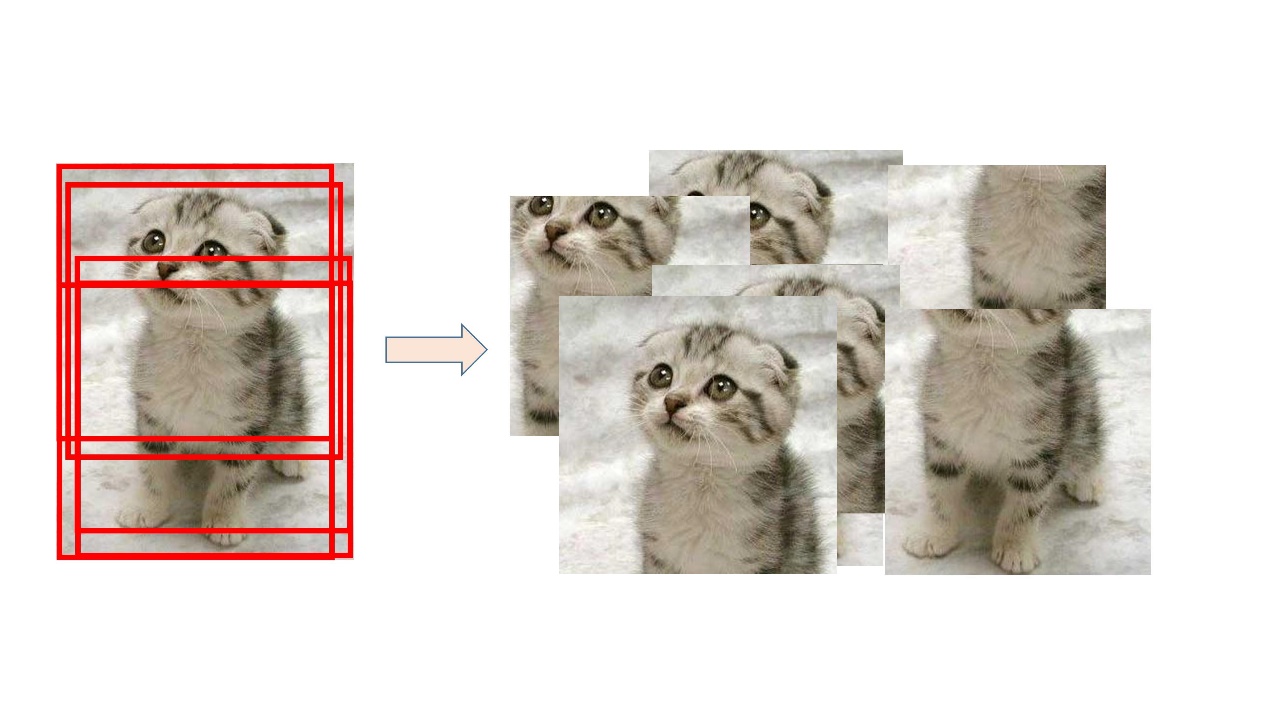
def get\_my\_classes(Annotations\_path, image\_path, save\_img\_path, classes):  
 xml\_path = os.listdir(Annotations\_path)  
 for i in classes:  
 if not os.path.exists(save\_img\_path+"/"+i):  
 os.mkdir(save\_img\_path+i)  
 for xmls in xml\_path:  
 print(Annotations\_path+"/"+xmls)  
 in\_file = open(os.path.join(Annotations\_path, xmls))  
 print(in\_file)  
 tree = ET.parse(in\_file)  
 root = tree.getroot()  
 if len(set(root.iter('object'))) != 1:  
 continue  
 for obj in root.iter('object'):  
 cls\_name = obj.find('name').text  
 print(cls\_name)  
 try:  
 shutil.copy(image\_path+"/"+xmls[:-3]+"jpg", save\_img\_path+"/"+cls\_name+"/"+xmls[:-3]+"jpg")  
 except:  
 continue

代码执行之后的结果如下：



**5.2图像块采样**

为了采样图像块，本实验中选用了随机裁剪的方式。以每张图像的中心点为基准，在[-图片长宽/6，图片长宽/6]的限定范围内进行中心点偏移，从而获得采样图像块，采样过程如图所示：



对于每张图像，共随机采样得到10个采样块，核心代码如下：

for each\_image in os.listdir(IMAGE\_INPUT\_PATH):  
 # 每个图像全路径  
 image\_input\_fullname = IMAGE\_INPUT\_PATH + "/" + each\_image  
 # 利用PIL库打开每一张图像  
 img = Image.open(image\_input\_fullname)  
 # 定义裁剪图片左、上、右、下的像素坐标  
 x\_max = img.size[0]  
 y\_max = img.size[1]  
 mid\_point\_x = int(x\_max/2)  
 mid\_point\_y = int(y\_max/2)  
 for i in range(0, 10):  
 # 中心点随机偏移  
 crop\_x = mid\_point\_x + \  
 random.randint(int(-mid\_point\_x/3), int(mid\_point\_x/3))  
 crop\_y = mid\_point\_y + \  
 random.randint(int(-mid\_point\_y/3), int(mid\_point\_y/3))  
 dis\_x = x\_max-crop\_x  
 dis\_y = y\_max-crop\_y  
 dis\_min = min(dis\_x, dis\_y, crop\_x, crop\_y) # 获取变动范围  
 down = crop\_y + dis\_min  
 up = crop\_y - dis\_min  
 right = crop\_x + dis\_min  
 left = crop\_x - dis\_min  
 BOX\_LEFT, BOX\_UP, BOX\_RIGHT, BOX\_DOWN = left, up, right, down  
 # 从原始图像返回一个矩形区域，区域是一个4元组定义左上右下像素坐标  
 box = (BOX\_LEFT, BOX\_UP, BOX\_RIGHT, BOX\_DOWN)  
 # 进行roi裁剪  
 roi\_area = img.crop(box)  
 # 裁剪后每个图像的路径+名称  
 image\_output\_fullname = IMAGE\_OUTPUT\_PATH + \  
 "/" + str(i) + "\_" + each\_image  
 # 存储裁剪得到的图像  
 roi\_area.save(image\_output\_fullname)

**5.3 Hog特征提取**

对每个图像块，调整其尺寸为256x256，并进行归一化，提取其Hog特征，核心代码如下：

# 图片预处理  
def preprocessing(src):  
 gray = cv2.cvtColor(src, cv2.COLOR\_BGR2GRAY) # 将图像转换成灰度图  
 img = cv2.resize(gray, (256, 256)) # 尺寸调整g  
 img = img/255.0 # 数据归一化  
 return img

# 提取hog特征  
def extract\_hog\_features(X):  
 image\_descriptors = []  
 for i in range(len(X)):  
 '''  
 参数解释：  
 orientations：方向数  
 pixels\_per\_cell：胞元大小  
 cells\_per\_block：块大小  
 block\_norm：可选块归一化方法L2-Hys(L2范数)  
 '''  
 fd, \_ = hog(X[i], orientations=9, pixels\_per\_cell=(  
 16, 16), cells\_per\_block=(16, 16), block\_norm='L2-Hys')  
 image\_descriptors.append(fd) # 拼接得到所有图像的hog特征  
 return image\_descriptors # 返回的是训练部分所有图像的hog特征

这里的cell大小选择为(16,16)，block大小为(16,16)。

**5.4 通过余弦相似度进行挖掘**

根据所有图像块的Hog特征，利用频繁性和判别性的标准来进行计算，核心代码如下：

threshold = 0.6  
group1 = []  
group2 = []  
group1.append(X\_features[0])  
for i in range(1, len(X\_features)):  
 res = cosine\_similarity(X\_features[0].reshape(1, -1), X\_features[i].reshape(1, -1))  
 if res > threshold:  
 group1.append(X\_features[i])  
 else:  
 group2.append(X\_features[i])

根据文献[2]的经验，这里的阈值分别选用了0.6,0.7,0.8，实验结果见下一节。

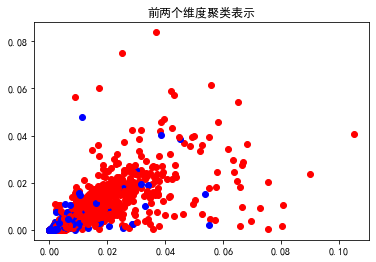
**5.5 通过k-means聚类方法进行挖掘**

在本实验中，也采用另一种方式即k-means聚类的方式来挖掘的视觉模式，核心代码如下：

cluster = KMeans(n\_clusters=2, random\_state=0)  
y = cluster.fit\_predict(X\_features)

colors = ['blue', 'red']  
plt.figure()  
for i in range(len(y)):  
 plt.scatter(X\_features[i][0], X\_features[i][1], color=colors[y[i]])  
plt.title("前两个维度聚类表示")  
plt.savefig("cluster.png")  
plt.show()

以“羊”作为频繁性的挖掘类别，将挖掘出的正类和负类样本前两个维度可视化表示如下：



提取出具有频繁性的视觉模式后，对判别性的挖掘，仍采用余弦相似度的方式，结果见下一节。

1. **实验结果**

本次实验选择的类别是“羊”(sheep)，并采用了两种方法以及不同的余弦相似度阈值，其数值结果如下表所示：

|  |  |
| --- | --- |
| 余弦相似度方法视觉模式挖掘频繁性 | |
| 阈值 | **频繁性** |
| 0.6 | **0.623** |
| 0.7 | **0.863** |
| 0.8 | **0.999** |

可以发现，随着阈值的增大，挖掘出的视觉模式频繁性越大。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 余弦相似度方法视觉模式挖掘判别性 | | | | | | | |
| 阈值 | **car** | **horse** | **cat** | **dog** | **bird** | **cow** | **平均判别性** |
| 0.6 | **0.887** | **0.291** | **0.315** | **0.258** | **0.489** | **0.144** | **0.397** |
| 0.7 | **0.957** | **0.513** | **0.64** | **0.567** | **0.708** | **0.382** | **0.628** |
| 0.8 | **0.988** | **0.791** | **0.914** | **0.841** | **0.873** | **0.716** | **0.854** |

相类似，随着阈值的增大，挖掘出的视觉模式判别性也越大，当阈值取值为0.8时，最高的平均判别性为0.854。

将该方法挖掘出的视觉模式，可视化部分如下：



在本实验中还采用了K-means聚类的方式来挖掘视觉模式，得到的频繁性数值为**0.707**，对于判别性的计算，仍采用余弦相似度的方式，其结果如下表所示：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| K-means聚类方法挖掘出的视觉模式判别性 | | | | | | | |
| 阈值 | **car** | **horse** | **cat** | **dog** | **bird** | **cow** | **平均判别性** |
| 0.6 | **0.883** | **0.26** | **0.299** | **0.242** | **0.493** | **0.146** | **0.387** |
| 0.7 | **0.956** | **0.503** | **0.632** | **0.569** | **0.713** | **0.391** | **0.627** |
| 0.8 | **0.987** | **0.793** | **0.91** | **0.843** | **0.879** | **0.724** | **0.856** |

和前一种方法相类似，随着阈值的增大，挖掘出的视觉模式判别性也越大，当阈值取值为0.8时，最高的平均判别性为0.856。

将该方法挖掘出的视觉模式，可视化部分如下：



由图可见，虽然两个方法挖掘出的视觉模式数值上差异性不大，但可视化结果却有差异。余弦相似度方法挖掘出的视觉模式更多在于羊的面部特征，而K-means聚类挖掘出的视觉模式更多在于羊的身体特征。

1. **实验总结**

本次实验，使用了传统的Hog特征提取方式，并使用余弦相似度和K-means聚类的方式来挖掘视觉模式。通过本实验，可以发现某一类图片的视觉模式可能不只一种，在本实验中，未考虑多种视觉模式的情况。针对此类情况，采用基于密度的聚类方式[2]可能会更加适合。

此外，对于视觉模式的挖掘，诸如Hog、Sift等传统特征提取方式，表征能力不强，不具有抽象能力。使用类似CNN的神经网络是目前更加主流且有效的方式。

**参考文献**

[1] N Dalal, B Triggs. Histograms of Oriented Gradients for Human Detection [J]. IEEE Computer Society Conference on Computer Vision & Pattern Recognition, 2005, 1(12): 886-893.

[2] 王倩楠. 基于深度学习的视觉模式挖掘算法研究[D].西安电子科技大学,2021.DOI:10.27389/d.cnki.gxadu.2021.002631.