**Literature Survey文献综述**

**回顾文献中的相关技术，以及理解你所选择的技术的任何必要背景。**

Task1是一个目标检测任务。

**Method 方法**

**证明并解释你所实施的技术的选择，必要时使用相关参考资料和理论。**

**》〉Task1**

**\*\*\*\*hog+SVM**

目标检测任务中，需要确定图像中目标是否存在以及其位置。HOG是一个基于梯度来计算直方图的特征描述符。HOG常用于人脸识别中，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。Hog所得到的特征描述符能够为分类以及目标识别提供重要的判断信息。

在计算HOG描述符时，会有以下参数.windowSize，blockSize，blockStep，cellSize，binNum.每个窗口会被划分成块，每个块会被划分成单元，nbins代表单元中统计梯度的方向数。采集单元中每个像素的梯度直方图。最后，将这些直方图组合起来，形成一个特征描述符。

一个块中的单元数为

A=(blockSize. width)/(cellSize. width)\*(blockSize. height)/(cellSize. height)

每个块包含binNum\*A梯度直方图。

一个窗口中的块数为

B=(((windowSize.width-blockSize.width)/(blockStep.width)+1)\*(((windowSize.height-blockSize.height)/(blockStep.height)+1)

一个窗口包含9AB梯度直方图。

The target detection task involves determining the presence of a target in an image and its location. HOG is a histogram of gradient-based feature descriptors, which is often used in face recognition to compute and count histograms of gradient directions in localized areas of an image to form features.

When computing the HOG Descriptor, there will be the following parameters. windowSize, blockSize, blockStep, cellSize, binNum. Each window will be divided into blocks, and each block will be divided into cells. nbins represent the number of directions of a statistical gradient in a cell. A histogram of the gradient of each pixel in the cell unit is captured. Finally, these histograms are combined to form a feature descriptor.

The number of cells in a block is:

A=(blockSize.width)/(cellSize.width)\*(blockSize.height)/(cellSize.height)

Each block contains binNum\*A gradient histogram.

The number of blocks in a window is:

B=((windowSize.width-blockSize.width)/(blockStep.width)+1)\*((windowSize.height-blockSize.height)/(blockStep.height)+1)

One window contains 9AB gradient histograms.

在使用hog的时候还有两个问题需要解决：尺度和位置问题。尺度问题可以通过resize图像来解决。为了解决位置问题，采用图像金字塔和滑动窗口。

There are two other issues that need to be solved when using hog: scale and position. The scale problem can be solved by resizing the image. To solve the position problem, image pyramids and sliding windows are used.

\*\*图像金字塔

图像金字塔是图像的多尺度表示，这有助于解决不同尺度下的目标检测问题。金字塔的层级越多，计算量越大，计算会更慢，但是结果在某种程度上越准确。

\*\* Image Pyramid

An image pyramid is a multi-scale representation of an image, which helps solve the problem of target detection at different scales. The more layers of the pyramid, the slower the computation will be, but the more accurate the results will be to some extent.

\*\*滑动窗口

设置窗口大小在图像中滑动，滑动窗口的检测和使用图像金字塔对各部分进行检测，这是为了在多尺度下检测对象。滑动窗口通过扫描较大图像的较小区域来解决定位问题，进而在同一图像的不同尺度下重复扫描。但是同时也会产生区域重叠的问题，就需要采取非极大值抑制来解决。

\*\*Sliding window

Setting the window size to slide in the image, sliding window detection and using the image pyramid to detect each part, this is to detect objects at multiple scales. The sliding window solves the localization problem by scanning smaller areas of a larger image and thus repeating the scan at different scales of the same image. However, there is also the problem of region overlap, which requires non-maximal suppression to solve.

**\*\*非极大值抑制**non\_max\_suppression

非极大值抑制在计算器视觉任务中使用广泛。由于在目标检测中，对于同一个目标常常会产生多个候选框，并且产生重叠。那么需要利用非极大值抑制来找到效果最佳的目标候选框，并且删除冗余的候选框。

Non-maximal suppression is widely used in calculator vision tasks. Since in target detection, multiple candidate boxes are often generated for the same target, and they overlap. It is necessary to use non-maximal suppression to find the best target candidate frame and remove the redundant candidate frames.

特征计算结束后，需要分类器来对特征进行分类。一般，采取SVM来分类。

在机器学习中，支持向量机（ support vector machine， SVM）是在分类与回归分析中分析数据的监督式学习模型与相关的学习算法。给定一组训练实例，每个训练实例被标记为属于两个类别中的一个或另一个，SVM训练算法创建一个将新的实例分配给两个类别之一的模型，使其成为非概率二元线性分类器。

After HOG, a classifier is needed to classify the features. Generally, support vector machine (SVM) will be chosen.

In machine learning, a support vector machine (SVM) is a supervised learning model and associated learning algorithm for analyzing data in classification and regression analysis. Given a set of training instances, each of which is labeled as belonging to one or the other of two categories, the SVM training algorithm creates a model that assigns new instances to one of the two categories, making it a non-probabilistic binary linear classifier.

**\*\*\*\*根据颜色阈值来区分（原理eddie）**

**Experiment Setup**

**解释实验设置和*评价方法*。**

**\*\*HOG+SVM**

<https://blog.csdn.net/nsh119/article/details/89738996>

1. 读取数据，获得正负样本，正样本根据csv裁剪获得，负样本是对图片除植物以外的区域进行随机裁剪获得。后期为了负样本的数据量更多并且准确，随机裁剪以及手动裁剪了负样本进行保存。每次运行的时候直接读取负样本数据。
2. 对所有的数据进行resize。这一步是为了加快运行速率。图片进行resize的同时，要对csv中的数据也进行一定的resize处理，使得图片与csv中的数据能够对应。Resize的大小通过比较现有图片的大小，设定为resize = (1647, 1158)。
3. 计算正负样本的hog特征，并为所有样本加上对应的label。正样本的label为1， 负样本的label为-1.
   1. Hog的参数设置

https://blog.csdn.net/yangpan011/article/details/78353873

hog = cv2.HOGDescriptor(winSize, blockSize, blockStep, cellSize, binNum)

winSize = (128, 128)

blockSize = (16, 16)

blockStep = (8, 8)

cellSize = (8, 8)

binNum = 9

nbins代表在一个胞元中统计梯度的方向数目。nbins=9表示一个胞元内统计9个方向的梯度直方图。

一个block 包含的cell个数是

一个block含有binNum\*A个梯度直方图

一个window 包含的block个数为

一个窗口包含9AB个梯度直方图。

所以根据参数设定，一个窗口会产生8100个梯度直方图。

* 1. 计算特征向量hog.compute()

1. 构建svm模型并进行第一次训练。
2. 根据第一次的训练结果，使用hardexample来优化模型。hard example是指利用第一次训练的分类器在负样本原图上进行植物检测时所有检测到的矩形框。这些矩形框区域很明显都是误报，把这些误报的矩形框保存为图片，加入到初始的负样本集合中，重新进行SVM的训练，可显著减少误报。这种方法叫做自举法(Bootstrap)，自举法首先使用初始负样本集来训练一个模型，然后收集被这个初始模型错误分类的负样本来形成一个负样本难例集。用此负样本难例集训练新的模型，此过程可以重复多次。
3. 提取hardexample的HOG特征并加入到先前的样本特征中，并且在label\_list加上hardexample对应的label（-1）。将所有hog特征重新进行训练，并保存模型。
4. 检测与识别。需要定位植物，因此需要获得区域信息。所以使用Hog类的setSVMDetector和detect（detectMultiScale）。
   1. rects, scores = hog.detectMultiScale(image, winStride=(8, 8), padding=(8, 8), scale=1.05)

img：待检测的图像

found\_locations：检测到的目标的包围框数组

hit\_threshold：检测到的特征到SVM分类超平面的距离，一般是设为0，在分类器参数中指明。

win\_stride：检测窗口每次移动的距离，必须是块移动的整数倍

padding：保持CPU接口兼容性的虚参数，必须是(0,0)。但网上下载的例子里是(32,32)

scale0：滑动窗口每次增加的比例

group\_threshold：组阈值，即校正系数，当一个目标被多个窗口检测出来时，该参数此时就起了调节作用， 为0时表示不起调节作用。

1. 非极大抑制优化重叠的检测区域，得到最终预测的bbox。
   1. 前提：目标边界框列表及其对应的置信度得分列表，设定阈值，阈值用来删除重叠较大的边界框。  
      IoU：intersection-over-union，即两个边界框的交集部分除以它们的并集。
   2. 根据置信度得分进行排序
   3. 选择置信度最高的比边界框添加到最终输出列表中，将其从边界框列表中删除
   4. 计算所有边界框的面积
   5. 计算置信度最高的边界框与其它候选框的IoU。
   6. 删除IoU大于阈值的边界框
   7. 重复上述过程，直至边界框列表为空。
2. 根据得到的bbox以及csv文件中的ground truth进行对比计算得到ap。

**\*\*颜色阈值**

1. 将绿色提取出来，进行预处理**（eddie）**
2. 此时得到的图像背景为黑，前景为白。反转颜色，使得背景为白255， 前景为黑0。得到binary image。
3. 使用两次联通运算（two-pass connected components algorithm）。在第一道中为每个数据点分配临时标签并存储标签与其邻居标签为等价物。第二道中选取等价物中的最小标签来替换临时标签。此时得到粗略的预测bbox。
4. 由于得到的二值图的效果有限，部分植物的茎叶缺失，导致同一株植物的被分成了好几个部分。因此要将这些部分进行合并。通过粗略的预测bbox，来重新分配label。
   1. 创建一个全为0的新的label\_image.找到每一个label的xmin，ymin，xmax，ymax，将整个矩形区域分配同一个label。将bbox根据area区域面积进行降序排列。
   2. 如果该区域的label全为0，那么就分配新的label。如果该区域的label不全为0，但是区域面积大于阈值，也分配新的label给这个区域。当区域面积小于阈值且label不全为0，那么找到label中不为0的值，将该区域赋值为这个value。
      1. 阈值是根据area面积排序得到的。选取粗略bbox中area降序排列前5%作为阈值。
      2. 如果区域面积大于阈值，推断有可能是一株植物。要与其他的植物区分开来。因此设定阈值。
5. 得到了全新的bbox。根据得到的bbox以及ground truth进行对比计算得到ap。

**\*\*AP评价方法的计算**

<https://www.zybuluo.com/Team/note/1175689>

reference：<https://tarangshah.com/blog/2018-01-27/what-is-map-understanding-the-statistic-of-choice-for-comparing-object-detection-models/>

目标检测问题中常用的评估指标之一就是Average Precision。计算AP，实际上就计算ground truth（真实标签）以及预测框之间的关系。精确度和召回率是评估模型的一般方法。AP的计算也需要精确度和召回率。

通常来说，精确度和召回率的计算需要鉴别True Positives、False Positives、True Negatives和 False Negatives。目标检测问题不同于普通的二元分类问题。在二元分类中，精确度和召回率都是直观的统计问题。在目标检测问题中涉及的不仅仅是分类问题，还有图像中物体的定位。

为了获得True Positives和False Positives，需要计算IoU（Intersection over Union）。

\*\*\*IoU（Intersection over Union）

IoU是评价边界框正确性的度量指标之一。 IoU计算预测框与ground truth的交集和并集的比值。对于每个类，预测框和ground truth重叠的区域是交集，而横跨的总区域就是并集。

在计算IoU的时候，会使用一个阈值，最常用的阈值为0.5。如果IoU大于阈值，那么就认为这个预测值是True Positive。如果IoU小于阈值，那么判断这个预测结果是False Positive。

True Negatives在目标检测中难以计算，因为只要是图像中未预测到的物体的部分都会被认为是Negative。因此只计算模型检测中未检测到的物体作为False Negatives。

步骤：

计算每个Positive预测框与ground truth的IoU值，并取最大的IoU值，认为该预测框检测到了那个IoU最大的ground truth。然后根据IoU阈值，计算出一张图片中各个类别的正确检测值（True Positives, TP）数量以及错误检测值数量（False Positives, FP）。据此，可以计算出各个类别的precision：

1. 计算预测的bbox中的每一个预测框与ground truth的IoU，并取IoU的最大值与IoU阈值进行比较判断TP与FP的数量。
2. 根据公式计算精确度和召回率。
3. 根据参考文献，选择11个不同的recall值[0, 0.1, ..., 0.9, 1.0]），并计算11个recall的精确度的平均值作为AP的值。

**Results**

**提供统计和视觉结果，同时讨论方法性能和实验结果。**

\*\*hog+SVM

给图，数据，表格

**Discussion and conclusions**

**\*\*HOG+SVM**

1. 非极大值抑制的阈值选择
2. 样本的多少，负样本的选取，样本很重要，要贴合项目实际环境去获取图片数据集。之前有尝试在读取图片数据后，随机提取与正样本数量相同的负样本。问题在于：选取的不够全面，仍会导致最后的检测不准确。所以最后是直接随机截取样本保存后，再进行人工筛选。
3. 不同的图片的效果不同。尤其是当植物较小的时候，检测效果不好。考虑是否在正样本的数据中多加入一些不同的较小的植物图片。
4. 运行时间较短，如果多次运行，可以直接调用保存的模型。

**\*\*颜色阈值**

1. 在预处理部分，如果过多的去噪音，那么就会导致植物的茎叶不明显。同一株植物叶子与叶子之间出现断连。
2. 在得到第一次粗略的预测bbox时，要剔除一些label的个数很少的点。实际上可以看作是手动去噪音。
3. 如果两株植物靠的太近或者两株植物的叶片发生了重叠，那么就会被认为是同一株植物。虽然可以定位植物的位置，但是会影响植物个数的判断。
4. 使用传统方法，因此在运行过程中，对图片的像素点进行了多次遍历。时间复杂度大，运行时间较长。

**\*\*两个方法进行总结比较**

--》不同点

1. Hog+svm需要大量的正负样本，并且样本的选取尽量需要贴合项目实际环境去获取。
2. 两者本质的不同，hog+svm是机器学习，颜色阈值就是传统方法。
3. Hog+svm适用于同类型的目标进行检测，例如行人检测。颜色阈值适用于单色物体的目标检测与定位。
4. Hog+svm 用时短，颜色用时长。但是前者对于检测较小的植物效果很差，检测较大的植物效果好。而颜色阈值的方法受到植物大小的影响较小。

--》相同点

1.

2.