**关于batchsize**

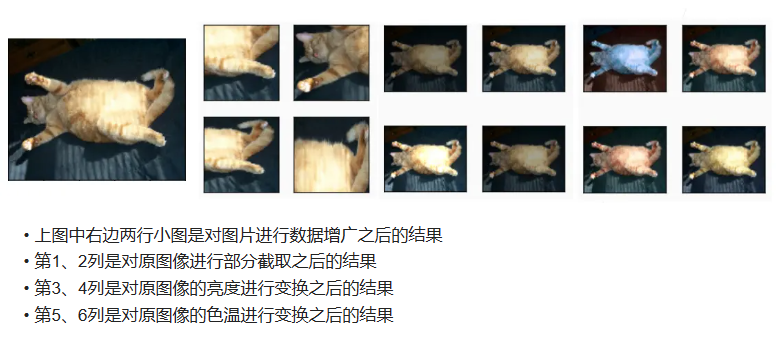
batchsize达到一定程度，每个batch内的样本的多样性不会比之前有多大增长，对梯度的贡献也不会比之前的batch大多少，但是大的batchsize会带来更多的训练时间，就造成了训练有效性下降。

**关于数据增广技术**

数据增强就是在已有的数据集上通过对数据的变换，使得数据具有多样性

对于语音数据来说，可以加入不同的背景噪音

对于图片数据来说，可以改变图片的形状，颜色等



1.数据增强通过变形数据来获取多样性从而使得模型泛化性能更好（基于现有的训练数据生成随机图像，以**提高模型的泛化能力**，本质上是希望通过对原始数据的变化来模拟现场部署或者是测试集中可能出现的情况）

2.常见的图片增强包括翻转、切割、变色

为了在预测过程中得到确切的结果，通常会只对训练样本进行图像增广，而在预测过程中不使用带随机操作的图像增广

3.在实践中，通常会组合使用多种图像增广方法

代码：



结果：



**关于微调：**

在源数据集上训练的模型对目标数据集也能起一定的作用

将前面层权重copy过来，在最后一层调整，重新训练。之前的叫pre-train

固定一些层：因为底层的特征更加通用，高层次的特征和数据集相关，可以固定底层的数据，不参与更新

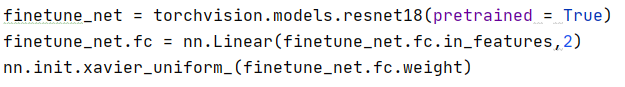
1.迁移学习将从源数据集中学到的知识迁移到目标数据集，微调是迁移学习的常见技巧。

2.除输出层外，目标模型从源模型中复制所有模型设计及其参数，并根据目标数据集对这些参数进行微调。但是，目标模型的输出层需要从头开始训练。

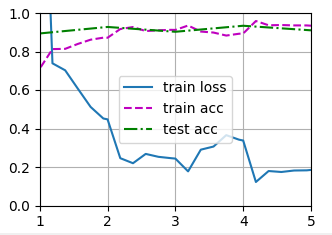
3.通常，微调参数使用较小的学习率，而从头开始训练输出层可以使用更大的学习率。

代码：

使用热狗数据集，输出的分类为两类，使用resnet18模型，将最后一层初始化

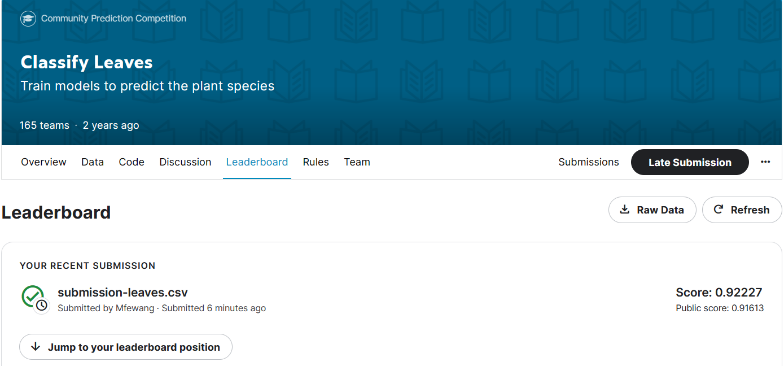




结果：



使用预训练过的模型更好

利用ResNet34选择pre-train，参加kaggle树叶分类比赛

acc为0.92227

**物体检测：**

在目标检测任务中，一张图像里往往不只一个感兴趣的物体对象，目标不仅仅是识别图像中所有感兴趣的物体（找出所有感兴趣的物体），还要找出它们在图像中所在的具体位置（通过方框来表示）

边缘框（boundingbox）：

边缘框可以用四个数字来定义（两种常用的表示方法）

（左上x，左上y，右下x，右下y）

（中心x，中心y，宽，高）

1.物体检测不仅能够识别图片所有感兴趣的物体的类别，还能够识别他们的位置，该位置通常由矩形边框表示

2.位置通常用边缘框表示（通常有四个数字）

3.用于目标检测的数据加载与图像分类的数据加载类似，但是，在目标检测中，标签还包含真实边界框的信息，它不出现在图像分类中

锚框（anchor）：

https://zhuanlan.zhihu.com/p/390824327

基于锚框来预测的检测算法

1. 生成大量锚框，每个锚框作为样本来训练
2. 计算IoU，给锚框上标号
3. NMS非极大值抑制

**目标检测的常用算法**

https://www.bilibili.com/h5/note-app/view?cvid=15651584&pagefrom=comment

一、区域卷积神经网络

R-CNN 模型的四个步骤：

1.对输入图像使用选择性搜索来选取多个高质量的提议区域。这些提议区域通常是在多个尺度下选取的，并具有不同的形状和大小；每个提议区域都将被标注类别和真实边框

2.选择一个预训练的卷积神经网络，并将其在输出层之前截断。将每个提议区域变形为网络需要的输入尺寸，并通过前向传播输出抽取的提议区域特征

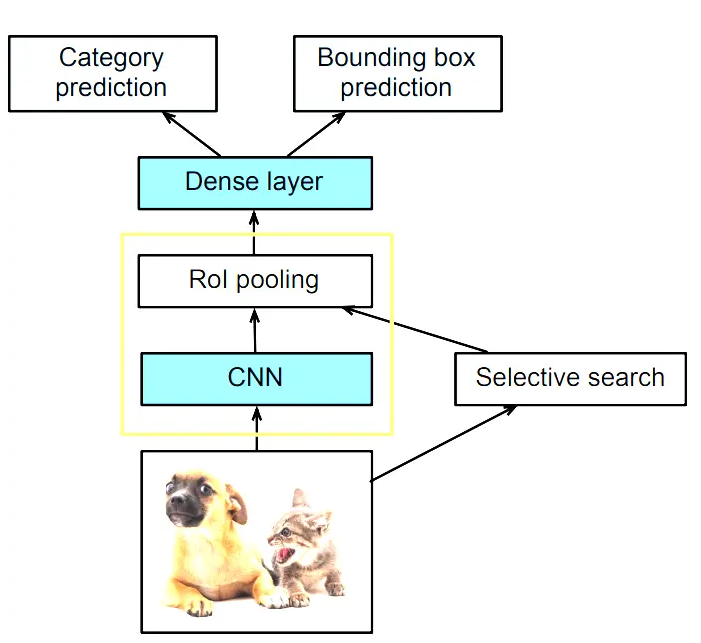
3.将每个提议区域的特征连同其标注的类别作为一个样本。训练多个支持向量机对目标分类，其中每个支持向量机用来判断样本是否属于某一个类别

4.将每个提议区域的特征连同其标注的边界框作为一个样本，训练线性回归模型来预测真实边界框

RoI pooling（兴趣区域池化层）：R-CNN 中比较关键的层，作用是将大小不一的锚框变成统一的形状

给定一个锚框，先将其均匀地分割成 n \* m 块，然后输出每块里的最大值，这样的话，不管锚框有多大，只要给定了 n 和 m 的值，总是输出 nm 个值

二、Fast R-CNN﻿

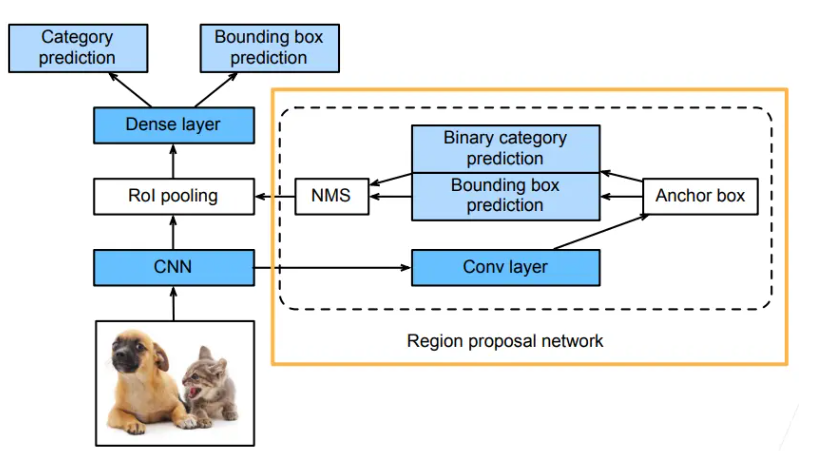
Fast R-CNN 的改进是：

1. CNN：对对整张图片进行特征提取（仅在整张图像上执行卷积神经网络的前向传播），得到一个 7 \* 7 或者 14 \* 14 的 feature map

2.selective search：用搜索到的原始图片上的锚框映射到 CNN 的输出上，再使用 RoI pooling 对 CNN 输出的 feature map 上的锚框进行特征抽取，生成固定长度的特征（nm 维的向量），之后通过一个全连接层对每个锚框进行预测：物体的类别和真实的边缘框的偏移

Fast R-CNN 相对于 R-CNN 更快的原因是：Fast R-CNN 中的 CNN 不再对每个锚框抽取特征，而是对整个图片进行特征的提取（这样做的好处是：不同的锚框之间可能会有重叠的部分，如果对每个锚框都进行特征提取的话，可能会对重叠的区域进行多次重复的特征提取操作），然后再在整张图片的feature中找出原图中锚框对应的特征，最后一起做预测

三、Faster R-CNN﻿

1.Faster R-CNN 的改进：使用 RPN 神经网络来替代 selective search

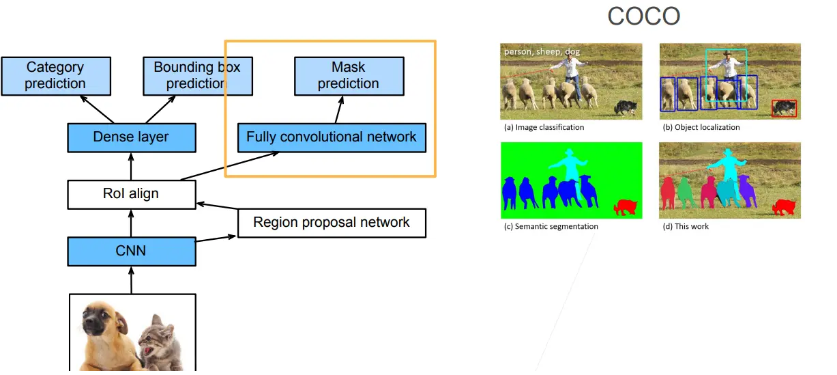
2.RoI 的输入是CNN 输出的 feature map 和生成的锚框

3.RPN 的输入是 CNN 输出的 feature map，输出是一些比较高质量的锚框（可以理解为一个比较小而且比较粗糙的目标检测算法： CNN 的输出进入到 RPN 之后再做一次卷积，然后生成一些锚框（可以是 selective search 或者其他方法来生成初始的锚框），再训练一个二分类问题：预测锚框是否框住了真实的物体以及锚框到真实的边缘框的偏移，最后使用 NMS 进行去重，使得锚框的数量变少）

3.RPN 的作用是生成大量结果很差的锚框，然后进行预测，最终输出比较好的锚框供后面的网络使用（预测出来的比较好的锚框会进入 RoI pooling，后面的操作与 Fast R-CNN 类似）

4.通常被称为两阶段的目标检测算法：RPN 做小的目标检测（粗糙），整个网络再做一次大的目标检测（精准）

5.Faster R-CNN 目前来说是用的比较多的算法，准确率比较高，但是速度比较慢

四、Mask R-CNN﻿

Mask R-CNN 是基于 Faster R-CNN 修改而来的，改进在于

1.假设有每个像素的标号的话，就可以对每个像素做预测（FCN）

2.将兴趣区域汇聚层替换成了兴趣区域对齐层（RoI pooling -> RoI align），使用双线性插值（bilinear interpolation）保留特征图上的空间信息，进而更适于像素级预测：对于pooling来说，假如有一个3 \* 3的区域，需要对它进行2 \* 2的RoI pooling操作，那么会进行取整从而切割成为不均匀的四个部分，然后进行 pooling 操作，这样切割成为不均匀的四部分的做法对于目标检测来说没有太大的问题，因为目标检测不是像素级别的，偏移几个像素对结果没有太大的影响。但是对于像素级别的标号来说，会产生极大的误差；RoI align 不管能不能整除，如果不能整除的话，会直接将像素切开，切开后的每一部分是原像素的加权（它的值是原像素的一部分）

3.兴趣区域对齐层的输出包含了所有与兴趣区域的形状相同的特征图，它们不仅被用于预测每个兴趣区域的类别和边界框，还通过额外的全卷积网络预测目标的像素级位置

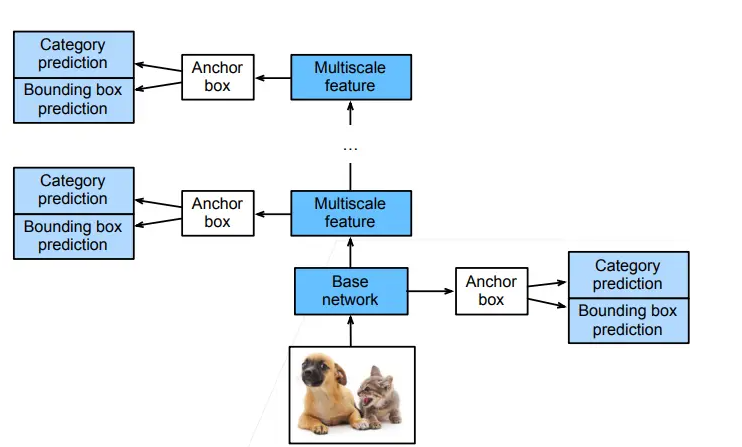
总结

1.R-CNN 是最早、也是最有名的一类基于锚框和 CNN 的目标检测算法（R-CNN 可以认为是使用神经网络来做目标检测工作的奠基工作之一），它对图像选取若干提议区域，使用卷积神经网络对每个提议区域执行前向传播以抽取其特征，然后再用这些特征来预测提议区域的类别和边框

2.Fast/Faster R-CNN持续提升性能：Fast R-CNN 只对整个图像做卷积神经网络的前向传播，还引入了兴趣区域汇聚层（RoI pooling），从而为具有不同形状的兴趣区域抽取相同形状的特征；Faster R-CNN 将 Fast R-CNN 中使用的选择性搜索替换为参与训练的区域提议网络，这样可以在减少提议区域数量的情况下仍然保持目标检测的精度；Mask R-CNN 在 Faster R-CNN 的基础上引入了一个全卷积网络，从而借助目标的像素级位置进一步提升目标检测的精度

3.Faster R-CNN 和 Mask R-CNN 是在追求高精度场景下的常用算法（Mask R-CNN 需要有像素级别的标号，所以相对来讲局限性会大一点，在无人车领域使用的比较多）

单发多框检测（SSD）：

1.输入图像之后，首先进入一个基础网络来抽取特征，抽取完特征之后对每个像素生成大量的锚框（每个锚框就是一个样本，然后预测锚框的类别以及到真实边界框的偏移）

2．SSD 在给定锚框之后直接对锚框进行预测，而不需要做两阶段（为什么 Faster RCNN 需要做两次，而 SSD 只需要做一次？SSD 通过做不同分辨率下的预测来提升最终的效果，越到底层的 feature map，就越大，越往上，feature map 越少，因此底层更加有利于小物体的检测，而上层更有利于大物体的检测）

3.SSD 不再使用 RPN 网络，而是直接在生成的大量样本（锚框）上做预测，看是否包含目标物体；如果包含目标物体，再预测该样本到真实边缘框的偏移

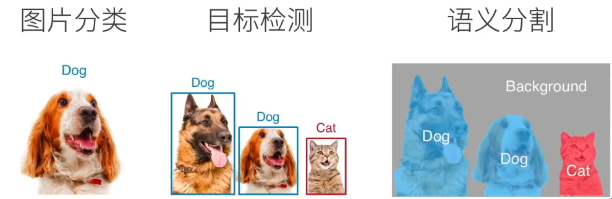
YOLO：

1.yolo 也是一个 single-stage 的算法，只有一个单神经网络来做预测

2.yolo 也需要锚框，这点和 SSD 相同，但是 SSD 是对每个像素点生成多个锚框，所以在绝大部分情况下两个相邻像素的所生成的锚框的重叠率是相当高的，这样就会导致很大的重复计算量。

3.yolo 的想法是尽量让锚框不重叠：首先将图片均匀地分成 S \* S 块，每一块就是一个锚框，每一个锚框预测 B 个边缘框（考虑到一个锚框中可能包含多个物体），所以最终就会产生 S ^ 2 \* B 个样本，因此速度会远远快于 SSD

4.yolo 在后续的版本（V2,V3,V4...）中有持续的改进，但是核心思想没有变，真实的边缘框不会随机的出现，真实的边缘框的比例、大小在每个数据集上的出现是有一定的规律的，在知道有一定的规律的时候就可以使用聚类算法将这个规律找出来（给定一个数据集，先分析数据集中的统计信息，然后找出边缘框出现的规律，这样之后在生成锚框的时候就会有先验知识，从而进一步做出优化）

**语义分割：**

语义分割将图片中每个像素分类到对应类别

（有监督学习）

语义分割通过将图像划分为属于不同语义类别的区域，来识别并理解图像中像素级别的内容

由于语义分割的输入图像和标签在像素上一一对应，输入图像会被随机裁剪为固定尺寸而不是缩放