1.2.3模型设计

1.2.3.1 Faster RCNN模型步骤

(1)首先在网络中输入自定义大小图片;

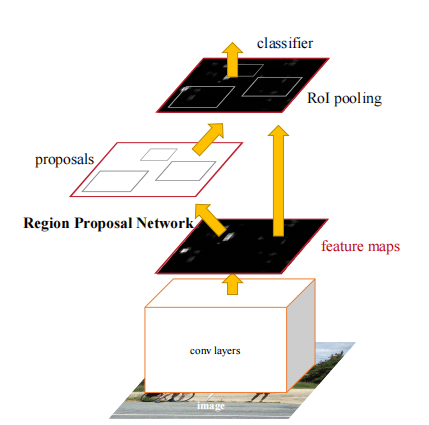
(2)在主干网络中时通过使用一组卷积、Relu层、池化层来对图像特征进行提取，从而得到图像特征信息，获取特征图(Feature map)。之后将得到的Feature map传输到RPN网络层，以便于生成候选区域和全连接层进行目标类别的分类。

(3)使用RPN网络主要用于生成区域推荐(ROIS)，在RPN网络层中通过使用Softmax函数来将Anchors进行二分类，判断是目标物还是不是目标物。接下来运用Bounding box regression对Anchors进行校正，这样可以得到更准确的Proposals。

(4)实现目标物的分类，把在第2个步骤得到的高维特征图和第3个步骤得到的输出区域--起运送到RoI池化层中，在神经网络模型的末尾运用全连接层，把目标物进行类别区分，实现目标物体的多分类。

(5)实现边框的回归利用Proposal feature maps计算每个Proposal所属地的不同类别概率信息，利用Boundingboxregression来得到物体检测框的准确位置，从而得到目标对象的检测结果。

Fastern RCNN模型的整体结构如图所示：

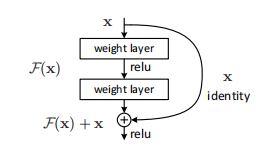


图Fastern RCNN的模型结构

1.2.3.2改进后的Faster RCNN模型介绍

(1)主干网络修改

首先针对于本次垃圾分类数据集的特点选用ResNet101网络作为主干网络，由于分类的类别数较多，我们需要提取较多的图像特征，因此也就需要较深的网络结构，然而当网络的深度过深时，传统的卷积网络或者全连接网络会出现信息丢失、梯度消失、梯度爆炸等问题，导致很深的网络无法训练。而 ResNet 即残差网络，能够很好地解决这一问题。其网络块结构如图 2所示



图ResNet的块结构

输入信息 X 经过两层权重映射后得到 ，也可以直接跳过权重层，残差块将

和 两者逐元素相加并用Relu 函数激活后输出得，则残差函数为：

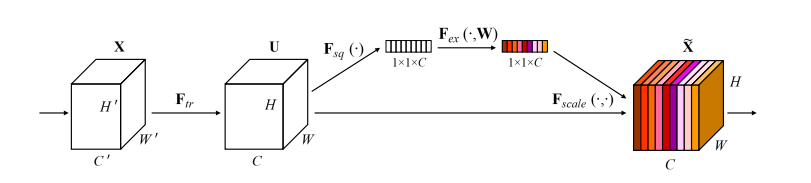


可以看到，输出中包含了输入原本的信息和输入经过权重层映射后的信息这样一来不仅能够防止信息丢失，更有效地利用信息，并且模型拟合残差函数要比直接拟合更简单，简化了学习目标，缩短了模型训练时间。

ResNet有不同的网络层数，比较常用的是 50-layer，101-layer。他们都是由图 2 的 ResNet块结构在一起实现的，101层的残差网络深度较深，能充分提取图像特征且相较于VGGNet其更网络结构更简单，因此我们以 ResNet101为网络结构搭建了图像分类神经网络模型。

(2)添加SE模块

为了合理利用有限的视觉信息处理资源，人类需要选择视觉区域中的特定部分，然后重点关注它。在神经网络中，注意力机制通常是一个额外的神经网络，能够硬性选择输入的某些部分，或者给输入的不同部分分配不同的权重。在神经网络中引入注意力机制有很多方法，以卷积神经网络为例，可以在空间维度增加引入Attention机制（如inception网络的多尺度，让并联的卷积层有不同的权重），也可以在通道维度（Channel）增加Attention机制，SE模块就是在通道维度（Channel-wise）上增加注意力机制。如下图所示：



图SE模块的结构图

首先进行压缩(Squeeze)，通过全局池化（global pooling），将每个通道的二维特征（H×W）压缩为1个实数，论文是通过平均值池化的方式实现。这属于空间维度的一种特征压缩，因为这个实数是根据二维特征所有值算出来的，所以在某种程度上具有全局的感受野，通道数保持不变，所以通过Squeeze操作后变为1×1×C，Squeeze的公式为：

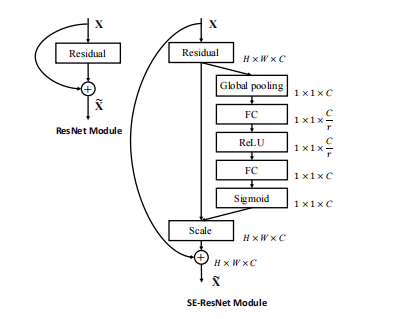


其次进行Excitation（Fex），通过参数来为每个特征通道生成一个权重值，这个权重值是如何生成就很关键了，论文是通过两个全连接层组成一个Bottleneck结构去建模通道间的相关性，并输出和输入特征同样数目的权重值。

最后进行Scale（Fscale），将前面得到的归一化权重加权到每个通道的特征上。论文中的方法是用乘法，逐通道乘以权重系数，完成再通道维度上引入attention机制，Scale的公式为：



在本网络结构中，选择把SE模块嵌入到ResNet当中，如下图所示，Global pooling就是Squeeze操作，FC + ReLu + FC + Sigmoid就是Excitation操作，具体过程是首先通过一个全连接层（FC）将特征维度降低到原来的1/r，然后经过ReLu函数激活后再通过一个全连接层（FC）生回到原来的特征维度C，然后通过Sigmoid函数转化为一个0~1的归一化权重。



图SE模块嵌入到ResNet结构图

(3)修改负样本挖掘策略(Hard Negative Mining Method)

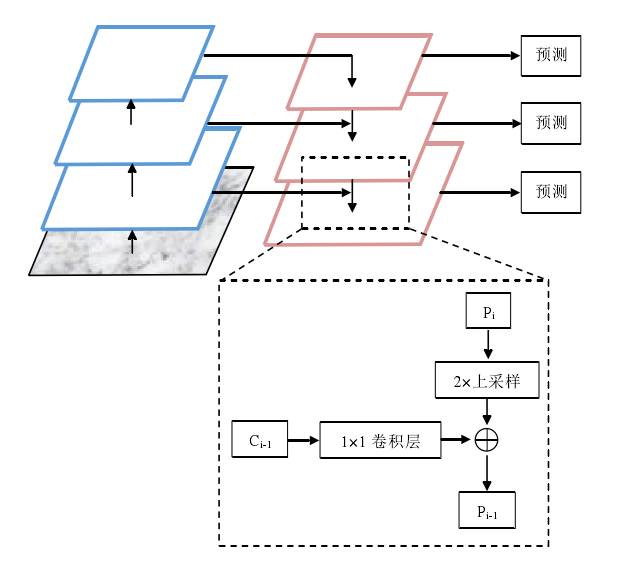
在 two-stage 的目标检测方法中，经过区域生成算法或者网络生成的 Region proposals 通常会经过正负样本的筛选和比例平衡后，才送入之后的检测网络进行训练。但是正负样本的定义和训练的比例是需要人为定义的。如果要筛选hard mining。通常使用的都是 hard negative mining的方法。但是此方法不适用于end-to-end的模型。因为会大大降低模型的训练速度，所以本模型使用OHEM方法来自动地选择 had negative 来进行训练，不仅效率高而且性能好。

对于一次 SGD 迭代，计算过程如下:先计算出特征图，可读 RoI 网络对所有 RoI 执行前向计算并计算每个 RoI 的损失，然后选择 hard RoIs。把这些 hard RoIs 输入到可读可写的 RoI 网络中执行前向前向计算和反向传播更新网络，并把可读可写的 RoI 网络的参数赋值给只可读的网络，一次迭代就完成了，速度和效率提升明显。

(4)添加特征金字塔网络(FPN)

在以往的Fastern RCNN进行目标检测时，无论是RPN还是Fastern RCNN，ROI都作用在最后一层，对于大目标的检测没有问题，但是对于小目标的检测就有些问题。因为对于小目标来说，当进行卷积池化到最后一层，实际上语义信息已经没有了，对于一个ROI映射到某个Feature map的方法就是将底层坐标直接除以Stride,显然随着网络的加深，映射过去后就越小，甚至可能就没有。所以为了解决多尺度检测的问题，引入了特征金字塔网络。

FPN的大致思路可以概成的深层特征，同时也有效地利用了网络中的浅层特征。FPN的大致思路可以概括为自底向上、自顶向下以及横向连接，示意图如所示。以ResNet模型为例，先根据模型的前向过程，自底向上地生成一系列尺寸不同的特征图{C2、C3、C4、C5}，这些特征图分别为ResNet 模型中conv2、 conv3\_ x、conv4\_ \_x以及conv5\_ x等残差模块的输出，相对于输入图像的步长分别为{4、8、16、 32},C1层由于尺寸过大且局部感受野过小，不纳入计算过程中。随后，在C5.上通过1x1的卷积操作进行维度变换得到P5，然后在特征图P5上通过最近邻上采样，将长宽各扩大1倍，向下获得尺寸与C4相同的特征图，将该特征图横向与维度变换后的C4相连，其中各元素相加融合得到P4，随后依次向下得到P3、P2。最后在P2~P5后分别加入3x3的卷积层，消除上采样过程中的混叠效应，输出特征均调整为256个通道。



图特征金字塔结构

(5)损失函数的修改

二分类模型的期望效果是模型总是给正样本输出1，负样本输出0，但模型的拟合程度有限，并不能完美做到这一点。在实际预测中，一般认为大于0.5的属于正样本，而小于0.5的属于负样本。

二分类模型表现不佳的原因通常在于正负样本比例不平衡，在检测过程中，锚框（Anchor Box）近似于滑动窗口（Sliding Window）的方式会使正负样本比例接近于1000：1，而且大多数负样本都是易于分类的样本，即easy example，这些easy example虽然loss很低，但是数量众多，占据了loss中的大部分，还主导了梯度，因此我们可以通过对损失函数的修改来平衡正负样本的重要性并易于区分hard example和easy example，降低简单样本的权重，让损失函数能够更关注于hard negative example，二分类问题的标准loss交叉熵损失的公式为：

对于二分类模型用Sigmoid函数激活，所以公式为

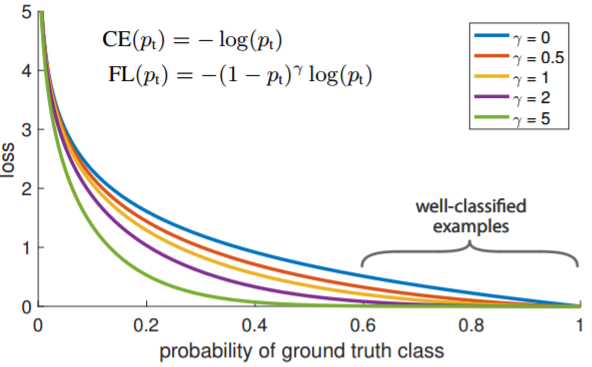


通过设定两个阈值，例如0.4和0.6，当模型对某个负样本的输出小于0.4，或者模型对某个正样本的输出大于0.6时，不需要再根据这个样本来更新模型，而只有在0.4~0.6之间的样本才会更新模型，这样就达到了让模型更关注于hard negative example的目的，从而使的训练后的分类效果更好

因此，交叉熵损失修改为[Kaiming He](https://link.zhihu.com/?target=https%3A//scholar.google.com/citations%3Fuser%3DDhtAFkwAAAAJ%26hl%3Den" \t "_blank)提出的Focal loss函数

γ的作用是调节权重曲线的陡度，例如在训练中负样本远比正样本多的时候，模型会倾向于数目多的负类，这时候负类的 就会比较小，而正类的 会较大，这时模型就会关注更多的正样本，如图所示。





图Focal loss改变参数伽马的损失曲线

在上图中，X轴表示预测真实值的概率，Y轴是给定预测值下的损失值。当γ=0时，Focal loss为交叉熵损失。从图像中可以看出，当模型以0.6的概率预测真实值时，交叉熵损失在0.5左右，因此为了减少损失，模型会以更高的概率预测真实值，这样会降低模型的鲁棒性，使得模型可能出现过拟合的情况。但是在类别不平衡的情况下，Focal loss会将模型的注意力转向稀有类别，这在一定程度上减小了过拟合情况出现的概率。

(6)标签平滑正则化

在深度学习样本训练的过程中，当采用one-hot标签去进行计算交叉熵损失时，可能会出现只考虑到训练样本中正确的标签位置（one-hot标签为1的位置）的损失，而忽略了错误标签位置（one-hot标签为0的位置）的损失。因此，模型虽然可以在训练集上拟合的很好，但是由于其他错误标签位置的损失没有计算，导致预测的时候，预测错误的概率增大。为了解决这一问题，标签平滑的正则化方法便应运而生。

为了达到这个目标，改进的方法是：在每次迭代时，并不直接将(xi,yi)放入训练集，而是设置一个错误率ε，以1-ε的概率将(xi,yi)代入训练，以ε的概率将(xi,1-yi)代入训练。 这样模型在训练时，既有正确标签输入，又有错误标签输入，训练出来的模型不会“全力匹配”每一个标签，而只是在一定程度上匹配。这样，如果出现错误标签，模型受到的影响会更小。

当没有标签平滑计算的损失只考虑正确标签位置的损失，而不考虑其他标签位置的损失，这就会出现一个问题，即不考虑其他错误标签位置的损失，这会使得模型过于关注增大预测正确标签的概率，而不关注减少预测错误标签的概率，最后导致的结果是模型在训练集上拟合效果非常良好，而在其他的测试集结果表现不好，模型泛化能力较差。

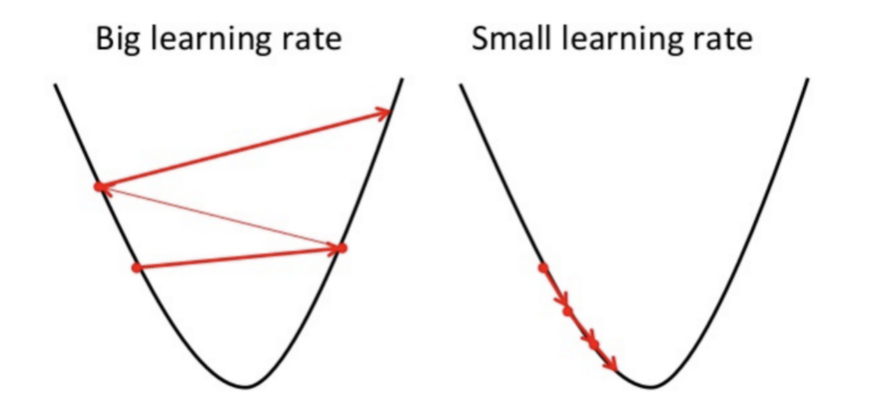
在进行平滑后的样本交叉熵损失不仅包含训练样本中正确的标签位置（one-hot标签为1的位置）的损失，也包含其他错误标签位置（one-hot标签为0的位置）的损失，使模型的学习能力提高，让模型往增大正确分类概率并且同时减小错误分类概率的方向前进。

标签平滑的实质是促使神经网络中进行Softmax激活函数激活之后的分类概率结果向正确分类靠近，即正确的分类概率输出较大（对应的one-hot标签为1位置的Softmax概率较大），并且同样尽可能的远离错误分类（对应的one-hot标签为0位置的Softmax概率较小），即错误的分类概率输出较小。

(7)优化方法修改

传统的深度学习优化方法是学习率调度器和自适应优化器，通常来说优化器都有一个学习率超参数，这是影响模型性能的最重要的超参数之一，在简单背景下，学习率是固定的，后来经实践发现，采用“模拟退火”的策略有助于模型更快的收敛，从而达到全局最优状态。

如果设置单一学习率，学习率的取值是难以确定的，在选择较大的学习率时，模型在早期会很快的逼近全局最优状态，但是在全局最优状态附近，大学习率可能会导致模型超过最优点，在最优点附近无法收敛；而较小的学习率会增加模型的迭代次数，延长训练时间，有可能会使模型陷入局部最优状态，从而不能收敛。因此应该保持学习率不断减小使其到达最优点，但是这种方法无法避免模型陷入局部最优状态，如图所示。



图模型陷入局部最优解示意图

而“模拟退火”的策略是指在模型训练的早期阶段，模型选择较大学习率，使得模型更快的逼近全局最优状态；而在训练后期，换成小学习率以逐渐收敛至全局最优状态；但是并不始终保持学习率一直减小，而是周期性的提高学习率，以便在模型陷入局部最优解的时候，学习率会向较高的学习率振荡以跳出鞍点。传统学习率调度器正是参考了模拟退火策略，选取base\_lr和max\_lr，让学习率lr在base\_lr和max\_lr之间周期性增加和减小，一次迭代（cycle）包含两个步长（step\_size），经过每次步长学习率lr会通过衰减函数scale\_fn更新一次，以此有效的加快收敛并提高准确率。

在传统学习率调度器的基础上，可以采用OneCycleLR（单周期学习率调度器）优化，设置整个训练过程只有一个cycle，学习率首先从初始值上升至max\_lr，之后从max\_lr下降至低于初始值的大小，一般设置为max\_lr的1/5或1/10，单周期的长度略小于要训练的周期总数，在最后的迭代中，将学习率设置为远低于max\_lr的值（如max\_lr的1/10）。以使得在学习率较大时，学习率可以作为正则化方法发挥作用，防止模型过拟合。

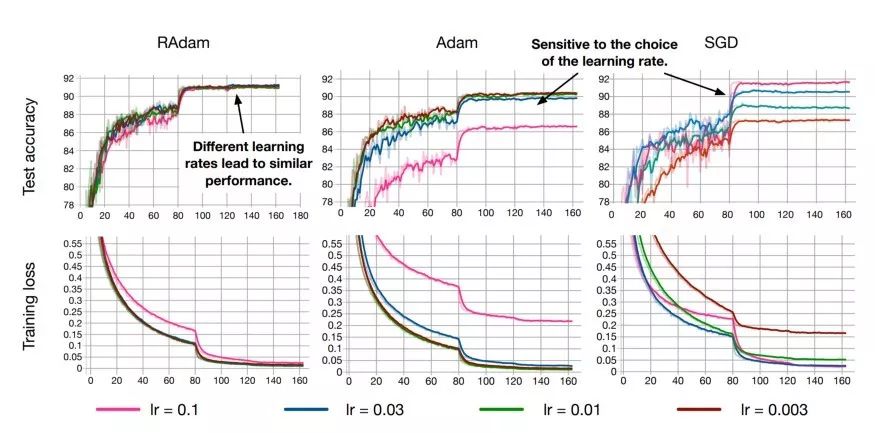
![MR$DCA]T`V1E_NKNUPUX](R](data:image/png;base64,)

图Onecyclelr的学习率变化曲线

自适应优化器并不使用单独的学习率调度器，而是将学习率优化直接嵌入到优化器本身，传统的自适应优化器如应用最广泛的Adam通过权重来管理学习率。Adam使用动量和自适应学习率来加快收敛速度，它的超参数非常少，改变它并不会对性能产生太大影响。

但是Adam具有鲁棒性不好的缺点，它常常会收敛到不太好的局部最优解，只能通过预热（warmup）策略来解决，即最初几次迭代，都用很小的学习率，以此来缓解收敛问题。这一问题出现的根源是模型早期缺乏足够的数据样本，因此学习率方差较大，自适应率的方差也较大。

因此改进模型采用优化算法RAdam来代替Adam优化模型。Radam和Adam相比降低了自适应学习率的方差，输入步长，衰减率，通过迭代计算梯度和动量的二阶矩，移动偏差的修正和方差修正范围，使用非自适应动量更新参数，RAdam的和Adam，SGD对比如图所示。



图RAdam的和Adam，SGD对比效果

Radam算法对于初始学习率具有鲁棒性，在一个很宽的范围内表现出了一致的性能，训练曲线末端高度重合。RAdam提供了一个动态启发式方法来提供自动化的方差衰减，从而消除了在训练期间预热所涉及手动调优的需要。此外，RAdam对学习速率变化具有更强的鲁棒性，并在各种数据集和各种AI体系结构中提供更好的训练精度和更强的模型泛化能力。