学号_2015302580088__

密级_____

武汉大学本科毕业论文

基于深度学习的黑烟车自动识别系统研究与实现

院（系）名 称：遥感信息工程学院

专 业 名 称 ：空间信息与数字技术

学 生 姓 名 ：张世杰

指 导 教 师 ：吴刚副教授 熊庆文教授

二〇一九年五月

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名：　　日期：

摘　要

计算机视觉是深度学习炙手可热的方向之一，得益于数据、硬件的发展，计算机视觉领域之目标检测出现了很多优秀的算法。本文尝试运用主流的目标检测方法实现对黑烟车的自动识别，减轻交管部门长期以来对黑烟车监管费时费力的问题，符合国家近年来对环境问题日益关注的政策方向。通过完成自建黑烟车数据集，大量的图像增广操作，搭建以 RestNet 为骨架的深度神经网络模型，迁移学习的应用以及网络模型参数调优等一系列工作后，我们得到的最终模型在测试集上达到了 0.9752 的 mAP。

关键词：深度学习；目标检测；黑烟车；卷积神经网络

ABSTRACT

Computer vision is one of the hottest directions of deep learning. Thanks to the development of data and hardware, many excellent algorithms have appeared in object detection, which is one of the fields of computer vision. This paper attempts to use the mainstream target detection method to achieve automatic identification of black tobacco trucks, alleviate the long-term problem of time-consuming and laborious supervision of black smoke trucks by traffic control departments, which is in line with the policy direction of the country's increasing concern for environmental issues in recent years. After completing a series of tasks, such as self-built data set of black smoke truck, a large number of image augmentation operations, building a deep neural network model based on RestNet, the application of transfer learning and the optimization of network model parameters, the final model we got reached 0.9752 mAP on the test set.

Key words: deep learning; object detection; black smoke truck; CNN

目　录

1 绪论

1.1 研究背景与意义

黑烟车即带病上路的机动车辆，正常行驶状态下，冒黑烟的车辆，其黑烟尾气，危害人体健康，严重污染大气。黑烟车（尤其以柴油车为主）的监督管理向来是车辆管理部门机动车污染防治的重中之重。

1.1.1 研究的背景

黑烟车所排放尾气中含有二百余类不同的化学物质，不仅会造成光化学烟雾污染，更是空气细小颗粒物超标的罪魁祸首。相关数据表明，二百辆国四排放标准小型汽车的排放总量，才与一辆国三排放标准重型柴油车排放的颗粒物相当[1]。

为了加快环境空气质量改善的步伐，中央所发布的《打赢蓝天保卫战三年行动计划》中，将排放达不到标准的柴油车辆列为重点监督对象[2]。《中华人民共和国大气污染防治法》也有相关条例，机动车驾驶人驾驶排放不达标的机动车上路行驶的，将交由公安机关交通管理部门依法予以处理。各地方政府先后出台了地方性条例，收紧对黑烟车的监管。

长期以来，交管部门都采用“逢车必查”的检测手段来查找黑烟车，还需要相关的尾气检测设备，这种传统检测手段的缺点十分明显，会导致道路交通更加繁忙，并且拦车检查时间有限、效率很低。

我们希望改变这一情况，借助于深度学习技术，实现对道路上黑烟车的自动检测、并对黑烟车排放的黑烟尾气进行抓拍，实现一个无需人员值守的黑烟车自动识别系统。

1.1.2 研究意义

1. 优化黑烟车治理与取证模式

通过“黑烟车自动识别系统”，无需大量人力投入即可准确找出道路上的黑烟车，使道路上的黑烟车无法逃脱。在自动识别黑烟车的同时，还可以做到相关视

频的自动截取保存，处罚过程有据可依。

2. 简化工作方式，提高整治效率

在过去，对黑烟车辆的控制只是通过人海的策略。对黑烟车辆的道路检查要求交管人员对道路上的车辆保持持续的注意力，这显然效率低下。我们期望通过“黑烟车自动识别系统”帮助有关工作人员减轻工作负担，不用再到公路旁盯着过往车辆，而只需对自动识别出的黑烟车进行二次审核即可。

3. 借助深度学习技术的成果解决生产中的实际问题。

深度学习技术发展如火如荼，通过网络我们可以获取大量训练图片，获取一块用于训练的 GPU 不再是一件难事，利用最新的深度学习技术成果解决实际问题是一件很有价值的事。

1.2 国内外研究现状及应用现状

目前国内针对具体的黑烟车识别这一问题的相关研究较少，国外并未发现相关文献。国内有江绮鸿、李文松等人[3]利用黑烟尾气中颗粒物对可见光中绿光吸收较多这一原理来进行黑烟尾气识别，具体做法是通过 CCD 传感器远距离探测机动车尾气管判定是否冒烟，根据 DSP 数据信号的有烟无烟，以及烟羽背后的深色和浅色目标之间对比度的差异，来衡量黑烟尾气的黑度。该方法本质上还是物理方法，需要相关传感器的支持，其实际工作识别准确率能达到 72%。

此外还有广东佛山、海南等地的相关部门使用黑烟车自动识别系统的媒体报道，但并没有详细介绍系统的实现方案，以及实际表现如何。

1.3 研究目标

本论文致力于依托深度学习技术搭建黑烟车自动识别检测系统，完成自动检测监控视频中黑烟车的任务，以解决交通监管部门在管理黑烟车困难的问题，通过技术手段为交通监管部门节省人力物力。具体研究目标如下：

1. 收集合适的黑烟车图片数据，制作数据集，并根据一定原则分为训练数据集与测试数据集，所制作的数据集中图片共同特征将成为下一步中模型优化的依据。

2. 搭建一个基于深度神经网络的黑烟车自动识别检测系统，训练系统所依托

的一个能够完成黑烟车目标检测的深度神经网络，并通过参数调整、数据处理等方法，尽量让模型取得较好的泛化能力。

1.4 研究内容

为了完成上一节的研究目标，我们计划开展了以下工作：

（1） 研究如何制作用于深度学习的数据集。通过各大搜索引擎搜索，利用爬虫收集黑烟车的图片，尽量涵盖了多种角度与场景，图片预处理，制作图片数据集并完成图片标注任务。

（2） 学习并研究计算机视觉中的目标检测方法，了解了目标检测算法主流的三类方法（Faster R-CNN/YOLO/SSD），比较了它们的优势与缺点，并根据实际需求，选择适宜当前工作的算法。

（3） 对主流的深度学习框架的优缺点进行比较，从中选择了使用方便、能快速上手、便于开展实验的开源框架。

（4） 训练一个能满足黑烟车自动检测的深度神经网络模型，并使用相应的手段，在增加模型识别准确率的同时，保证泛化能力。

2 相关技术研究

2.1 卷积神经网络（CNN）

早在上世纪就被提出的卷积神经网络，得益于如今快速发展的计算能力与充足的数据，计算机视觉领域因之取得令人振奋的成就。不仅限于计算机视觉，在众多其它领域，它的表现也超出人们的意料[4]，因此也得到了广泛的使用与发展。

单就计算机视觉而言，CNN 可以被用于物体检测、物体分割、风格转换等，但实际上 CNN 真正能做的，只是起到一个特征提取器的作用，以上的应用也都是建立在 CNN 对图像进行特征提取的基础上进行的。

卷积神经网络得名于网络中的卷积层，除此之外池化层、全连接层也是卷积网络的重要构成，它可以进行平移不变分类。

2.1.1 卷积层

传统神经网络（多层感知机）用全连接的方式联系输入层与隐藏层，这使得计算更加复杂，这种方式对于更大的图像输入将需要更多的输入单元，伴随而来的还有更加大量的网络参数，28*28 的图像需要 784 的输入单元，而更大的图像 96*96 将需要 9216 个输入单元，这会消耗更多的计算与内存资源。另一方面，普通神经网络将图像展开为一个个像素然后输入进网络，这样的方式会对物体在图像中的位置产生较大的依赖性，并且无法表征图像局部空间的特征。

我们以二维卷积为例介绍卷积层，输入矩阵同卷积核进行卷积运算输出另一二维矩阵，如图【2.1】所示，一个 3x3 的二维矩阵作为输入，采用 2x2 卷积核，等号右边输出的深色部分元素的计算方法为：0×0+1+3×2+4×3=19。

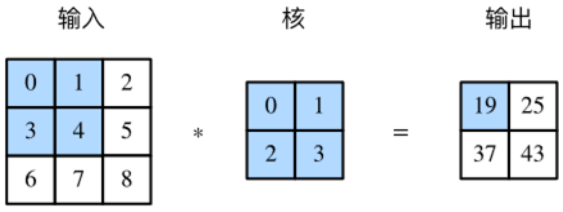

图 2.1 二维卷积运算

卷积层工作时卷积核会在输入的图像上滑动，可以检测出图像中像素变化的位置，而像素发生变化就意味着这是图像中物体的边缘，因此使用卷积核可以获得一种图像特征有效的表征方式。

卷积层的输出被称为特征图（feature map），因为其是输入图像特征的某种表征方法。对于输出中的某一个元素而言，输入中所有参与了它的计算的元素集合，被称为它的感受野（receptive field）。以图【2.1】为例，输入中深色部分的所有元素是输出中深色元素的 receptive field。我们将图【2.1】中的输出表示为 y，现在要进一步扩展卷积神经网络，用 y 和另一个 2x2 的卷积核进行卷积运算，可以得到只有一个元素的输出（z），此时，z 在上 y 的 receptive field 包括 y 中所有元素，在输入上的 receptive field 包括输入中所有元素。

2.1.2 池化层

池化层(pooling layer)不仅能减轻卷积层对图像中物体位置的敏感性，还能达到特征降维的目的。

上一节中，利用卷积层我们可以检测出物体的边缘，在真实的场景中，我们所要检测的目标是在图像中出现的位置是有多种可能性的，而不是某个固定的位置，因此我们不能直接把卷积层输出直接用于后面的模式识别，卷积层具有过度的位置敏感性，而这是我们所不期望的。

与卷积层十分类似，池化层也通过一个固定形状的窗口依次在输入数据上进行滑动计算，此时我们把这个窗口称为池化窗口。二者区别之处在于，卷积层是将输入数据与卷积核的进行卷积计算，而池化层的计算相对简单，通常有直接计算窗口内最大值（最大池化层）或者平均值（平均池化层），对应的操作也被称为最大池化(max pooling)或平均池化(mean pooling)。通常认为，平均池化往往能从较大的范围感知整体的数据特征，有助于背景信息的保留；而最大池化则恰好相反，其能更好的突出局部的纹理特征。

以二维最大池化层为例，简述池化过程：池化窗口从输入数组最左上角开始，依次在输入数组上滑动，滑动过程同卷积层，池化窗口每滑到一个位置，将找到此时池化窗口中的最大值，并将其输出。池化层也可以通过填充以及修改步长，来得到打通大小的输出，其填充和步长的工作机制与卷积层一致。

2.2 深度卷积神经网络代表

现代 CNN 的起点始于上世纪末出现的 LeNet，然而 LeNet 在被提出之后就陷入了沉寂，数据的缺乏与硬件计算能力的薄弱，无法让 CNN 的潜力得到释放，SVM 等手工设计的特征也一度受到人们的追捧。可是部分研究人员仍然没有放弃这片领域，他们发现了更简单有效激活函数（ReLU），找到了一些网络训练的技巧（dropout），紧接着随着互联网世界数据量的爆炸以及硬件计算能力的快速发展。2012 年 AlexNet 一鸣惊人，取得 ImageNet（国际著名的计算机视觉竞赛）竞赛第一，让人们真正认识到了 CNN 的能力，紧接着 AlexNet 之后，更多的基于卷积神经网络的深层模型被提出，并继续更新竞赛记录，如 VGG、GoogLeNet、ResNet。

深度卷积神经网络发展的趋势是深度越来越深，可实际上想要获得一个有效的深度模型并不仅仅是加深深度那么简单，网络越深，梯度消失越明显。因此出现了批量归一化以及残差网络这样一些方法。

2.2.1 LeNet

LeNet 最早被提出是用来解决手写数字识别的问题，采用了梯度下降来训练网络，其准确度达到了商用级别，图【2.2】展示了其网络结构。LeNet 用实际向人们表明[5]，CNN 可以在手写数字识别这一问题上取得相当好的效果，这是一项具有开创性的工作，CNN 从此走进人们的视野[5]。

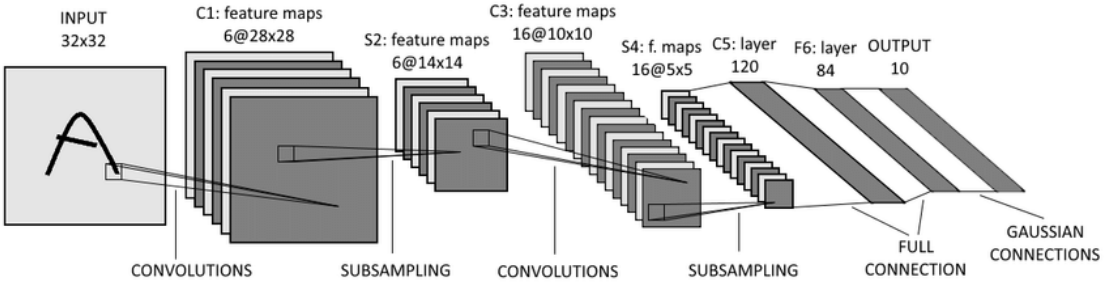

图 2.2 LeNet 网络结构[5]

如图【2.2】所示，LeNet 先使用卷积层后接全连接层，卷积层块中利用卷积层可以提取出图像中的边缘信息，多次卷积过后可以提取出组合的边缘信息，我们知道卷积层对位置具有敏感性，而后接的最大池化层的作用就在于此。图【2.2】中含有两个卷积层，前一个卷积层输出通道数为 6，后一个卷积层比前一个卷层输

入的高宽更小，为了让两个卷积层的参数数量大体一致，将后一个卷积输出通道增加到 16。卷积层块中使用了池化窗口 2×2、步长为 2 的最大池化层，这样的设计导致池化窗口的每次滑动的区域不会相互覆盖，并且池化会可以将特征图大小减半。

当数据被传到全连接层块时，经过第一个全连接层块（如图【2.3】所示，其输出大小为 120）后，数据的形状将会变为二维，分别是批量大小以及单个样本被 flatten 后的一维向量表示，而卷积层块输出的形状是一个四维向量，分别是批量大小、通道、高、宽。不难看出，单个样本被 flatten 后的向量长度为通道数 x 高度 x 宽度，在图【2.3】中全连接层块一共有三个全连接层，其中最后一层输出数为 10，这既是我们识别的数字的个数（0-9）。

2.2.2 AlexNet

LeNet 提出后，并没有得到人们在其它领域的广泛应用，这是因为 LeNet 在 MNIST（手写数字图片集）这样小型而简单的数据集可以工作的很好，但是一到一些较大而复杂的数据集上后，其表现完全不能满足实际应用的需要。其中的原因是两方面的，其一，神经网络多层、多通道的设计所带来的的大量网络参数以及复杂的计算过程，在当时那个没有 GPU 这样有强大计算能力硬件的时代，训练的时间可能长到令人难以等待；其二，由于当时人们对神经网络没有给予足够的重视，所以对于参数如何初始化，神经网络中复杂的非凸函数如何更快下降、如何避免局部最优点以及鞍点等一些问题的研究并不是很充分，这又进一步加剧了复杂神经网络训练困难的局面。

2012 年的 ImageNet 竞赛冠军 AlexNet 为这一切带来了转机，经过了十几年的技术发展与沉淀，外部条件逐渐成熟，AlexNet 得以在 LeNet 的基础上实现一个更加复杂的深层神经网络模型（八层），并通过实战再次证明了 CNN 的巨大潜力，自此以后通过 CNN 网络学习到的特征全面超越手工设计的特征[6]。

AlexNet 同 LeNet 一样也使用了卷积层、最大池化层、全连接层这样的一些网络结构，除了拥有更深的网络深度外，它还使用了以下一些新技术点：

1.激活函数不再使用 sigmoid 而改用 ReLU，由于没有 sigmoid 中的求幂运算，ReLU 的计算更加容易。此外，ReLU 可以使梯度下降更快，模型训练更加容易，

sigmoid 在很大的区间上的梯度都趋近于零，而 ReLU 在整个正区间都没有这个问题。

2.丢弃法(Dropout)的引入，训练时 AlexNet 会随机忽略全连接层中的一部分神经元，避免模型过拟合训练数据，提升泛化能力。

3.图像的增广的大量使用，不仅减轻了过拟合，显著扩充了训练数据集，而且模型的泛化能力也得到了了保证。

4.AlexNet 选择了最大池化层作为池化方法，并且步长小于池化核的大小，最大池化有效保留了图像的显著特征，小步长让池化窗口相互重叠，增加了池化输出的冗余，丰富了图像特征。

2.2.3 VGG

GoogleNet 摘得 ILSVRC 2014 冠军，此次比赛中的亚军 VGG[7]也同样值得关注，后续的大量实验表明，VGG 相比于 GoogleNet 更加适合于 transfer learning。此外VGG还让我们看到了一种有效的构建深度模型的新手段，可以把一些卷积层、池化层以一定的次序连接从而形成块结构，然后在网络模型中，反复使用使用这些块。

VGG 通过 VGG 基础块构建，通常使用如图【2.3】这样的一个表来描述 VGG 的网络结构，基础块中每一个卷积层所使用的卷积核均为 3x3，为了在卷积层不改变输入的尺寸大小，还对输入做了 1 的填充，之后跟的是池化核大小 2x2 的最大池化层，池化核的步长均为 2，通过这样的设计池化层将把输入的尺寸减半。

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

图 2.3 VGG 网络结构[7]

2.2.4 ResNet

理论上，当我们在神经网络模型中添加一个新的层时，充分训练的模型应该能够更有效地减少训练误差。可是实际情况并不符合我们的预期，在我们向深度模型中添加更多的层时，模型的实际表现并没有改善反而下降了，我们将此称为深度模型的“退化”。批量归一化等一些手段对该问题也显得无能为力。

针对于模型深度过深出现的退化问题，ResNet 被提出[8]，其基本结构是如图【2.4】所示的残差块。

为了理解残差网络，我们将 x 视为网络的输入，F(x)视为网络的输出，H(x)为模型拟合的目标。在之前的网络中我们的训练目标都是要让 F(x)无限 x 趋近于

H(x)，而 ResNet 的不同之处在于，其对输出 F(x)进行了处理，为其加上了输入，此时实际的输出成了 F(x)+x，训练的目标也相应变为了 F(x)=H(x)-x，这就是为什么我们将其称为残差网络。

假设有一个十分复杂的任务，需要神经网络的层数达到很深才能完成，并且对于这个任务，我们知道一定存在最优网络，但是对于它的结构我们无从知晓，为了将这个最优网络涵盖到我们的网络结构中，我们为网络添加了很多的层，因此这其中必然有一部分是多余的，只有当这些多余的层不会改变其输入时，我们的网络表现才能与最优网络一致。

当多余层的拟合目标为 H(x)=x 时，就能达到我们所需的这种恒等变化，此时之前的网络输出目标就变为了 F(x)=x，而对于 ResNet 它只需要网络输出 F(x)=0，就能实现多余层的恒等变化，这显示是一个更加容易的目标。

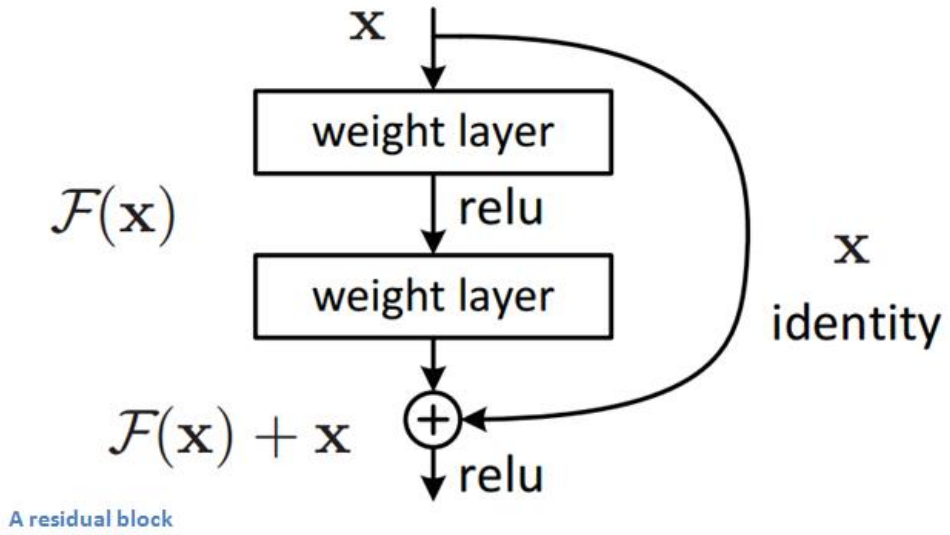

图 2.4 残差块[8]

2.2.5 MobileNet

网络模型发展的趋势是越来越深，而这会带来更高的模型复杂度（无论计算还是存储），ResNet 可以达到 152 层，可是当我们实际在某些应用场景（移动端、嵌入式）中使用模型时，这些模型往往会由于过于复杂而无法被使用。

MobileNet 是由 Google 提出的一种小巧但高效的 CNN 模型[9]，其在保持模型识别准确度的前提下降低了模型复杂度，同时对模型速度也进行了提升。

深度级可分离卷积是 MobileNet 的基本结构，该结构将卷积操作分解成了两个

更小的操作：depthwise convolution 和 pointwise convolution。通常卷积核会被应用于所有输入通道，而 depthwise convolution 所谓的 depth，是对于每个通道采用不同的卷积核，具体就是一个卷积核应用于一个输入通道。而 pointwise convolution 所谓的 point，就是采用 1x1 卷积核。

具体计算过程如图【2.5】所示，左侧为 depthwise convolution 示意，通过在不同输入通道上分别进行卷积，右侧为 pointwise convolution 示意，其作用在于将 depthwise convolution 的输出进行融合，经过这两步操作后我们得到一个和传统卷积操作输出十分接近的结果，却只使用了更小计算量和参数量。

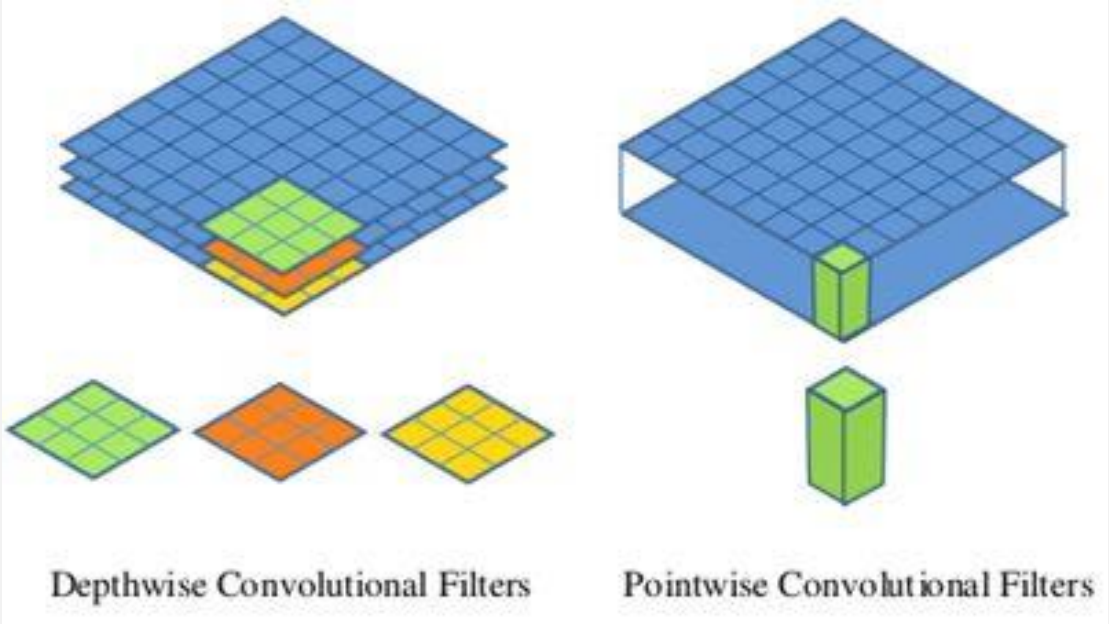

图 2.5 depthwise convolution 和 pointwise convolution[9]

2.3 目标检测相关概念

目标检测高于图像分类任务并建立于其基础之上，我们不仅要知道所感兴趣物体的类别（classification），还需要在图像中将其标注出来（location）。

目标检测有着多样的应用场景，例如，在自动驾驶中，借助目标检测我们可以将道路上的其他车辆、行人、障碍物检测出来，并标出位置，以此避开[10]；自动化工厂中，通过目标检测机器人可以自动识别相关物体，以此完成指定的任务；机场安检处也可以借助目标检测，检出旅客行李中所携带的危险物品。

目标检测要解决的核心问题：

1. 物体在图像中出现的位置是不固定的。
2. 物体大小是多尺度的（可能很大，也可能很小）。
3. 物体可能有不同的形状。

在目标检测里，我们通常使用如图【2.6】所示的边界框（bounding box）来描述目标位置[11]。边界框是一个矩形框，可以由矩形左上角坐标以及长宽确定。我们根据上面的图的坐标信息来定义图中狗和猫的边界框。

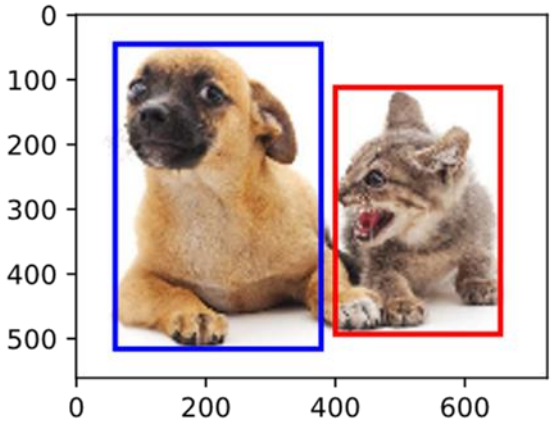

图 2.6 边界框

2.3.1 锚框

我们知道在目标检测中，感兴趣的物体出现的图像中的位置是不固定的，并可能有各种不同的形状。一个自然的思路是使用一些不同尺寸大小的滑窗在整幅图像的各个像素上滑动，可是这样的做法计算复杂度过高。

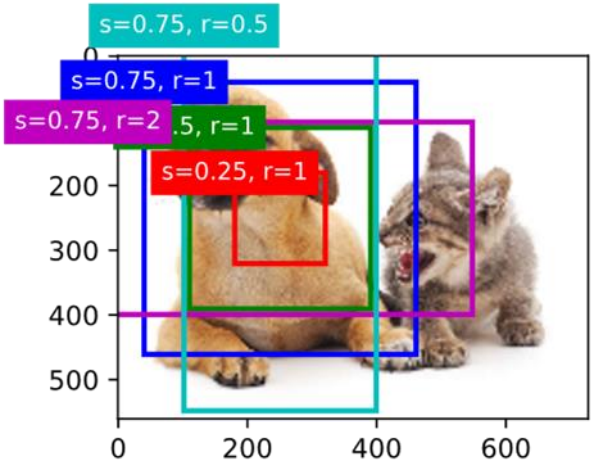

图 2.7 锚框

如何高效的给出一些物体在图像中可能出现的区域呢？一般来说，目标检测算法会将整幅图像分割成一个个小的区域，对于划分出来的每一个小的区域，再进一步判断是否存在感兴趣的物体，若存在感兴趣的物体则还会对边界框进行调整、精修，以此确保预测出的物体位置是准确的。不同的算法采用不同的策略去产生这些可能存在物体的区域，其中一种常用的手段是如图【2.7】所示的锚框，其通过为一定间距的像素生成具有不同纵横比以及大小的多个边界框来生产可能的区域建议。

2.3.2 交并比

假设生成的某个锚框恰好完全框住了感兴趣的物体，并且我们已经为该物体标注出了其真实边界框，我们该如何去衡量某个锚框好于其它的锚框呢？在集合中，通常使用 Jaccard 系数（Jaccard index）去评估两个集合的相似程度(交集除以并集)。

若把边界框内的像素区域视为一种特殊的像素集合，类似于集合的方式，Jaccard 系数也可以被用于评估预测边界框与真实边界框的相似程度。此时，我们将 Jaccard 系数称为交并比，如图【2.8】所示用两个边界框相交面积除以相并面积，当结果为 0 表示两个边界框完全不相似，为 1 表示两个边界框高度相似。

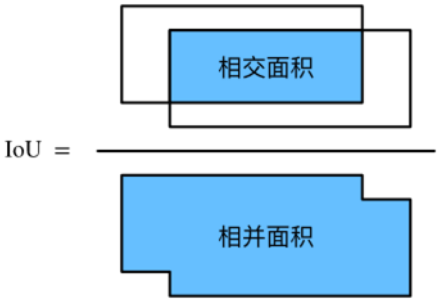

图 2.8 交并比

2.4 目标检测算法代表

目标检测的相关研究也已有近二十年的历史。随着深度学习技术的再次获得了人们的关注的视线，目标检测算法脱离了 SVM 统治时期的手工特征提取方法，诉诸于以 CNN 作为基础特征提取网络。从 R-CNN 再到 Fast R-CNN、Faster R-CNN、YOLO、SSD，在很短的时间内，更优秀的算法不断出现，检测策略有一阶段和二

阶段，多尺度检测方法有单尺度网络、特征金字塔网络，PC 端到移动端全面覆盖[12]，这些算法在公开数据集上的表现也不断刷新人们的预期。

常见目标检测算法的策略为下面两种：

（1） 一阶段策略，先把整幅图像均匀拆分小的规则区域，比如上面所提到的锚框的方法，接着在卷积神经网络输出的特征图上直接进行分类与边界框回归，该类算法速度快，准确度低，代表 YOLO。

（2） 两阶段策略，先是通过 CNN 网络或启发式方法产生一系列的可能的候选区域，再对这些区域进行类别预测与边界框精修，该类方法准确度高，速度慢，代表 R-CNN。

2.4.1 Faster R-CNN

R-CNN 通过启发式的选择性搜索算法来产生一系列可能的兴趣区域，之后对每个兴趣采用卷积神经网络进行类别预测[13]。

Fast R-CNN 在 R-CNN 的基础上，尝试解决其效率极低的问题，其采用兴趣区域池化来共享计算量，提高了模型的计算效率[14]。

基于 R-CNN 和 Fast RCNN 所取得的成果，Faster R-CNN 在结构上将特征抽取、proposal、bounding box regression、classification 都整合到了网络中，较大地提升了模型的综合表现，在检测速度方面尤为明显[15]。

Faster R-CNN 包含四个关键构成：

1. Conv layers。Faster R-CNN 通过卷积神经网络得到图像的特征图，特征图同时被区域建议网络和全连接层使用。

2. Region Proposal Networks。区域建议网络将生成区域建议（region proposals），该层首先判断锚框属于前景物体还是背景，然后利用边界框回归修正锚框的位置。

3. RoI Pooling。将之前的特征图和区域建议输入兴趣区域池化，产生建议特征图，之后用于目标类别判断。

4. Classification。计算 proposal 的类别，同时利用边界框回归获得物体边界框的精确位置。

Faster R-CNN 针对 Fast R-CNN 中存在的问题有以下一些改进，一是通过使用

区域建议网络，简化候选区域的生成；二是为了简化了训练让区域建议网络和Fast-RCNN 使用了同一套参数。

2.4.2 YOLO

在 YOLO（you only look once）之前，一般都是首先用区域建议来生成相当数量的、也许包含感兴趣物体潜在边界框，再去对各边界框评估是否含有感兴趣物体，所属类别的置信度，如 R-CNN 系列算法。

YOLO 不再把目标检测当作分类任务，而是看成回归任务，其只对每张图片“过一遍”就能得出是否存在感兴趣的物体以及感兴趣物体所在区域[16]。

检测速度快是 YOLO 的显著优点，其很适合用于实时目标检测，但是其物体检测精度低于其他主流方法，对两个距离较近的物体或者小物体的检测效果不如人意（其每个方格只能被确定为一类物体）。

其后的 YOLO-v2 添加了 batch norm，基础网络改用了 Darknet-19，提高了 2% 的 mAP[17]。YOLO-V2 从 Faster R-CNN 学习到了锚框，并使用 K-Means 来选择更好的候选区域，在卷积层增加候选框预测，算法表现有所提升。小物体的识别也有了改善，通过将深浅层的特征融合，有效利用了图像的细粒度特征

2.4.3 SSD

SSD（Single Shot MultiBox Detector）可以认为是 Faster RCNN 和 YOLO 的结合[18]，相比较于 Faster RCNN 没有生成区域建议，而是直接预测边界框的类别与坐标。针对多尺度目标检测问题，SSD 发现利用不同卷积层的特征图直接进行综合所产生的效果也同样符合要求。

SSD 对多尺度特征图进行融合，这有效改善了模型对多尺度目标的识别问题，网络前面的较大的特征图，因为存留有较丰富边缘信息，可以被用来检测小目标，而网络中靠后的小特征图，通过对之前特征图中不同特征的组合含有较高维的信息，可以用来检测大目标。

SSD 也使用了锚框的概念，每个像素根据不同大小、长宽比生成不同的潜在边界框。每个物体的真实边界框可能会与多个潜在边界框相交，根据交并比找出与真实边界框相似程度最高的潜在边界框与其匹配，匹配结束后对与每个真实边

界框匹配的潜在边界框进行边界框回归。

SSD 检出小尺寸物体的效果要比大尺寸物体差。因为当到达网络顶层时，小尺寸物体在特征图中所占像素已经很少了，但是通过增加输入大小可以一定程度上缓解这一问题，还可以对原图像进行随机裁剪起到放大原图像的作用。

3 模型分析与选型

3.1 目标检测算法的分析选型

我们将从 Faster R-CNN、YOLO、SSD 中选择一个目标检测算法用于我们的黑烟车检测任务。

Faster R-CNN 优点在于较高识别准确率，短板在于训练繁琐、识别速度慢。

YOLO 相较于其他目标检测算法检出速度得到了保障，但这是以识别准确率为代价换取的。

SSD 兼顾了准确率和速度，但容易漏检图像中的小尺寸对象（仅占几个到几十个像素）[19]。

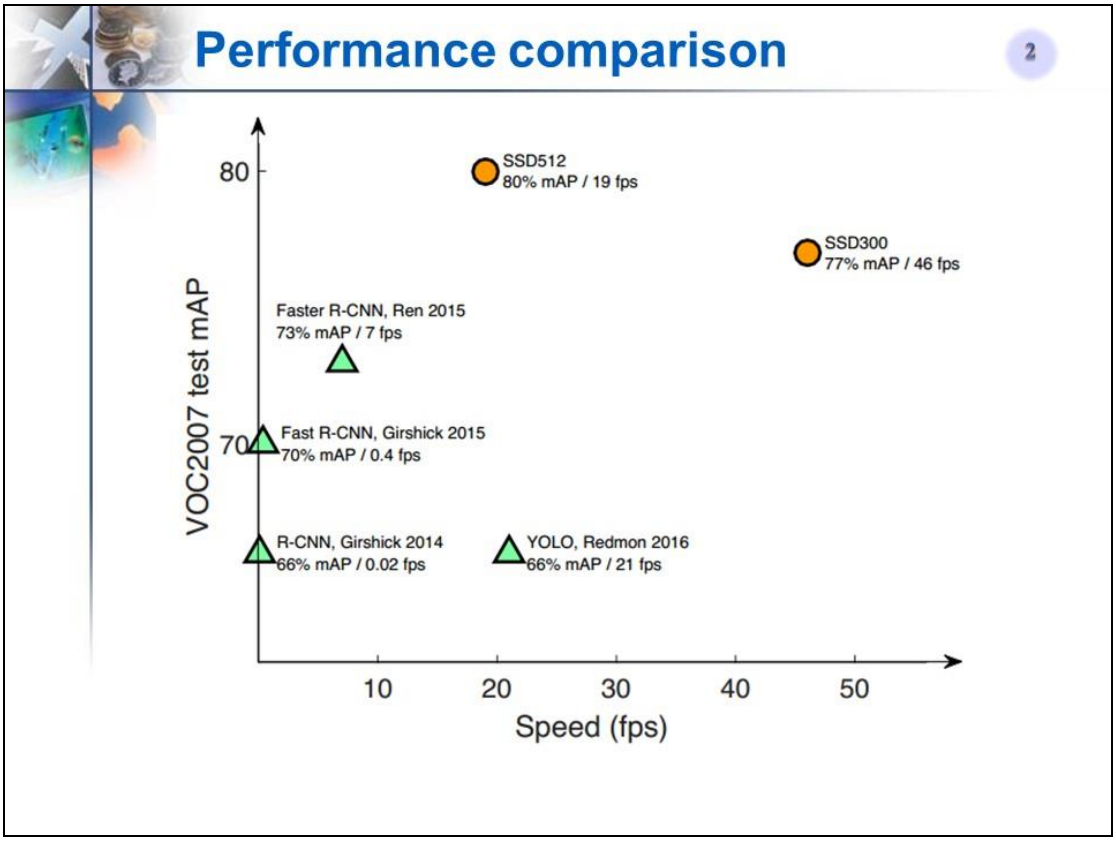

图 3.1 目标检测算法性能比较[19]

根据图【3.1】，在本文中我们选择 SSD 作为我们的目标检测算法，为了缓解 SSD 对小尺寸对象识别率不高的问题，我们将网络模型输入的图片大小设置为

512*512。

3.2 深度学习框架的分析选型

深度学习框架为我们封装了深度学习中的常用操作，可以让我们不需要自己写 CUDA 就能使用 GPU，或者是自动帮我们求解复杂函数的梯度，大大提升进行深度学习工作研究的效率。

在本文中我们选择深度学习框架的主要依据是其文档齐全程度，以及新手上手难易程度，因此我们选择了 Amazon 官方支持的 MXNet，其文档相对丰富，并且有循序渐进的系列教程帮助入门，其在设计过程中吸收借鉴之前深度学习框架的特点，掌握了 MXNet 后，再迁移到其它平台也会相对较容易。

除此之外，MXNet 生态圈内还有专门用于计算机视觉应用的 Gluoncv 库，Gluoncv 为我们提供了很多在 PASCAL VOC 或者 COCO 这样的公开数据集上预训练好的模型，通过使用这些预训练的模型可以减少我们自己训练模型的时间。

3.2.1 TensorFlow

- Github stars 数目: 126,816
- 由谷歌开源，以及开发维护，强大的社区支持。
- 有着良好的可视化支持，拥有 TensorBoard 这样的可视化工具。
- 文档丰富却混乱，官方提供的教程在逻辑上很不连贯，新手很难找到一个真正循序渐进的入门教程。

3.2.2 Keras

- Github stars 数目: 40,840
- 仅是一个深度学习的前端库，必须有一个后端（TensorFlow、CNTK）才能工作，简化了接口、上手快、丰富的资料。
- 一个神经网络的搭建往往只需要要极少量的代码
- 过度封装导致灵活性丧失，Keras 速度较慢，由于接口封装层次太多，容易让人对深度学习如何工作不明就里。

3.2.3 Caffe

- Github stars 数目: 27,982
- 几乎不需要编写代码，借助配置文件来定义模型、训练过程。
- 设计之初就是针对图像类问题而开发的，对于 CNN 网络支持较好。
- 性能优异，但文档不够完善

3.2.4 Torch

- Github stars 数目: 27,662
- Torch 的接口使用小众的编程语言 Lua，PyTorch 重新使用 python 设计实现了 Torch。
- PyTorch 的设计追求最少的封装，灵活快速，简单易用
- 除了官方文档以外，只有有限的参考资源

3.2.5 MXNet

- Github stars 数目: 16,809
- 起源于开源社区的项目后得到 Amazon 的推广支持，广泛使用于 AWS，进入 Apache 孵化器。
- 站在之前框架的肩膀上，多 GPU、分布式支持很好。
- 资源利用率高，对深度学习计算做了专门优化，计算效率高，节省 GPU 显存，在同样大小的显存下可使用更大的批大小。
- 文档丰富，生态圈内还有专注于计算机视觉的 GluonCV，以及专用于自然语言处理的 GouonNLP。

4 模型构建

4.1 数据收集与预处理

算法、数据、算力被称为深度学习的“三架马车”，一个好的可被应用于实际生产环境的深度学习模型的获得，必定依赖于数量足够、组成多样的训练数据集。在本文中，为了训练出一个能够准确识别出道路上的黑烟车的深度学习模型，通过爬虫爬取了百度、必应、谷歌等各大搜索引擎对相关关键字的图片搜索结果，此外也通过视频帧截取的办法获取了部分图片数据。

在实际操作过程中，我们发现通过上述方法得到的图片数据集图片质量低(分辨率不高、清晰度不够、太亮或太暗)，并且难以涵盖不同角度、光线，因此我们对图片数据集进行了直方图均衡化、翻转、旋转、光照变化等操作以期望模型能在实际情况中取得更好的泛化效果。

4.1.1 图片数据收集

图片数据集主要来源为各大搜索引擎的关键字搜索结果。通过国内外搜索引擎搜索相关图片，对搜索结果进行爬取，一共爬取了来自国内搜索引擎的黑烟车图片 234 张，来自国外搜索引擎的黑烟车图片 287 张，原始图片共计 521 张。

爬取自国内搜索引擎的图片更加符合我们对黑烟车的定义，该类别图片多为对道路上公交车、柴油车的抓拍，并且黑烟的形态也更符合我们日常生活中所见到的汽车黑烟；而来自国外搜索引擎的图片结果更多的是国外改装的卡车，其中大量车辆有两个向上的排放口排放黑烟，其黑烟更浓更黑，并且黑烟通常会占据图片像素的很大一部分，这部分图片中的黑烟略显夸张。

我们将来自国内搜索引擎的图片称为一类图片，见图【4.1】；来自国外搜索引擎的图片称为二类图片，见图【4.2】。

图 4.1 一类图片示例

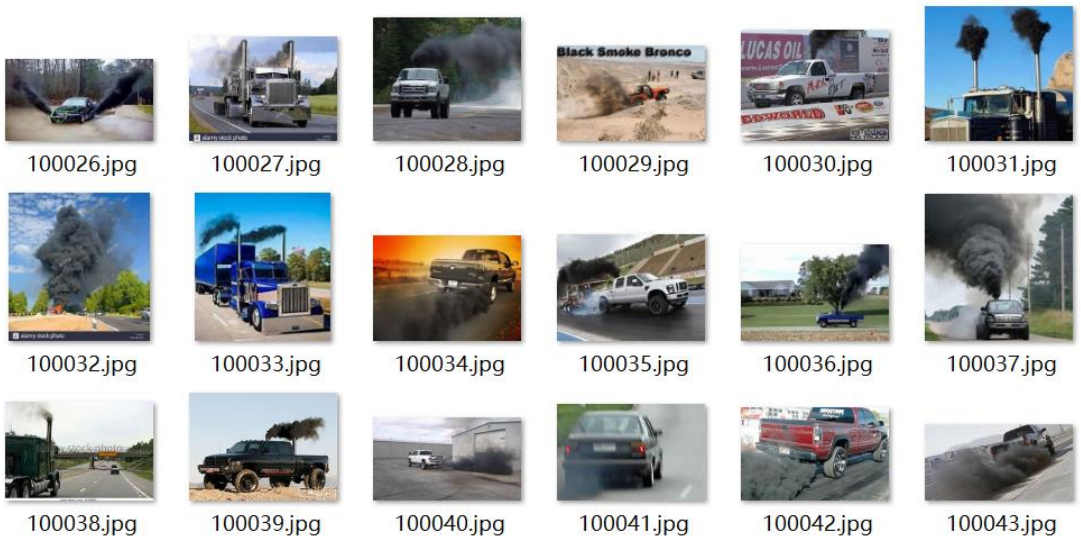

图 4.2 二类图片示例

4.1.2 图片数据预处理

在介绍图片数据预处理之前，先介绍一下图像增广（image augmentation）。

图像增广技术为了产生一系列图像内容相同但展现不同的训练样本，会对原始训练图像做一些的随机变化，这样做有两个好处，一是可以有效的将训练数据集的大小成倍扩充，二是通过对原始训练图像的改变能够减轻模型对某些特征的敏感度，可以得到更加有效的模型。比如说，通过亮度的明暗变化、对比度增减，可以缓解模型对颜色的依赖性；通过对原始训练图像进行随机裁剪，得到多种感兴趣物体在图像中位置的可能性，我们又可以缓解模型对感兴趣物体位置的敏感

度。图像增广技术有多重要？在之前的我们谈过的 AlexNet，之所以能取得显著进步，图像增广技术是对此贡献巨大。

具体来说，图像增广技术被分类两种：

a) 离线增广：在尚未开始训练前对图像进行增广操作，当我们只能收集到少量数据时，该方法可以让数据集的数量成倍的增加。

b) 在线增广：对于已经较大的数据集，不能承担像前者那样成倍的扩充，其在训练过程中提取训练图片时可进行在线图像增广。

我们同时使用了二者，使用直方图均衡化、翻转、饱和度变化等方法对数据集进行离线增广，显著扩大了训练数据集，在训练过程中又会对图像进行裁剪、旋转等增广方法进行在线增广，以期望保证模型的泛化能力。

通过上一节的方法获取到的图片在数量上并没有达到我们的预期，因此我们使用了直方图均衡化、水平翻转、光照变化、饱和度调整等一系列方法来扩充我们的数据集。我们使用 Pillow、scikit-image 这两个 Python 图片处理库进行了图像处理。

直方图均衡化（histogram equalization）通过调整图像直方图使其更加平均而改善原始图像对比度，还原更多的图片细节。该方法简单而有效，可以缓解原图像灰度“扎堆”，分辨率不明显，图像不清晰。通过实际比较了全通道展开然后进行直方图均衡化，与分别在 RGB 三个通道进行直方图均衡化两种方法的效果，在本文中，我们选择了后者作为我们的直方图均衡化方法，直方图均衡化效果如图【4.1】所示。

图 4.1 直方图均衡化

为了降低深度学习模型对黑烟尾气飘动方向的敏感度，我们对图片做了水平翻转、小角度旋转(小于 30 度)等操作进一步扩充了数据集。上一节中谈到，从国外搜索引擎获取到的黑烟车图片中的黑烟太过于浓黑，而实际中并不是所有黑烟

车的黑烟尾气都如此夸张，所以我们对图片进行了亮度、饱和度、对比度调整，以此让模型对一些不那么黑浓的黑烟尾气也能做到准确识别。

通过本小节的图片数据预处理后，我们的图片数据集数量扩充到 2605 张。

4.1.3 制作 Pascal VOC 数据集

PASCAL VOC 挑战赛（The Pattern Analysis, Statistical Modeling and Computational Learning Visual Object Classes）是一个世界知名计算机视觉类竞赛，这项挑战赛已于 2012 年停止举办了，但是研究者仍然可以在其服务器上提交预测结果以评估模型的性能。

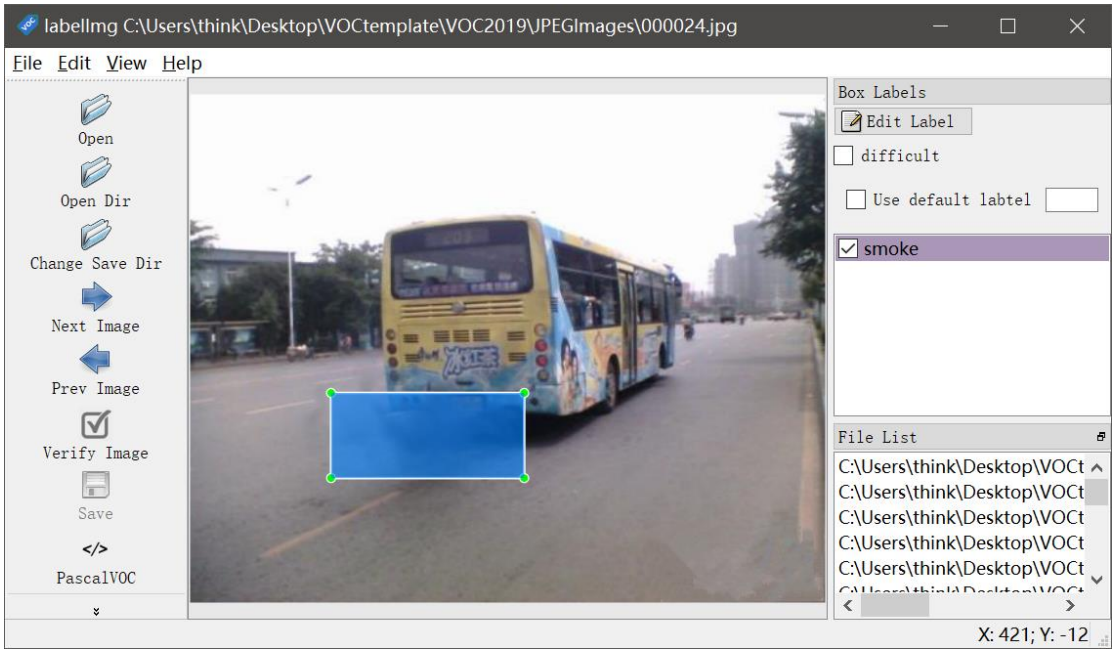

图 4.2 labelImg 界面

该竞赛于 2005 年开始举办，每年的内容都有所不同，从最开始的分类，到后面逐渐增加检测，分割，人体布局，动作识别（Object Classification、Object Detection、Object Segmentation、Human Layout、Action Classification）等内容，数据集的容量以及种类也在不断的增加和改善。该项挑战赛催生出了一大批优秀的计算机视觉模型（尤其是以深度学习技术为主的）。数据集共有四个大类车辆,居家物品,动物,人，以及 20 个小类[20]。

我们根据 Pascal VOC 官方数据文件夹组织结构，创建文件夹 VOC2019，并为其创建三个子文件 Annotation、ImageSets、JPEGImages，将图片文件放置到

JPEGImages 文件夹下，Pascal VOC 对于目标检测的标注采用 xml 来存储，标注文件存放于 Annotation 文件夹，ImageSets/Main 中用来保存训练集以及测试集的索引文件。

```xml
<?xml version="1.0" ?><annotation>
    <folder>JPEGImages</folder>
    <filename>000001.jpg</filename>
    <path>JPEGImages/000001.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>357</width>
        <height>262</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>smoke</name>
        <pose>Unspecified</pose>
        <truncated>1</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>1</xmin>
            <ymin>57</ymin>
            <xmax>161</xmax>
            <ymax>169</ymax>
        </bndbox>
    </object>
</annotation>
```

图 4.3 PASCAL VOC 目标检测标注格式

图片中物体标注工具为开源软件 labelImg，其界面如图【4.2】所示，我们选择中打开的文件夹以及标注文件夹存储位置后，并选择标注文件的格式为 PascalVOC 即可进行图片标注工作。Pascal VOC 目标检测标注的 XML 文件内容如图【4.3】所示。

4.2 深度神经网络构建与训练

本文实验环境为租用的带 GPU 的云服务器，基本配置如下：

硬件环境：

- CPU：Intel(R) Xeon(R) CPU E5-2660 v3 2.60GHz（3 核）

- GPU：Nvida TITAN X 显存 12G
- 内存：16G

软件环境：

- 操作系统：Ubuntu 16.04
- CUDA 版本：10.0
- cuDNN 版本：7.4
- Python：3.76
- MXNet 版本：mxnet-cu100 1.4.0.post0
- gluoncv 版本：gluoncv 0.4.0.post0

4.2.1 数据加载

我们使用小批量随机梯度下降算法(SGD)，因此一个好的小批量数据加载方法十分重要，本节将介绍本文所采用的数据加载策略。

首先，我们根据上一步制作的 PASCAL VOC 数据集 ImageSets/Main 文件夹中训练集索引文件(train.txt)以及测试集索引文件(test.txt)确定训练集与测试集的图片范围。

然后，将图片加载进内存，每张图片将会被保存为一个形状为(通道数，高，宽)的一个 tensor（多维数组），图片各通道的灰度值变换到[0,1)范围内，对于其标签将使用一个长度为 5 的一维数组进行保存，物体所属类别将被保存在数组首位，类别为-1 表明这个边界框是用于填充的无效边界框，将会被网络忽略。其余四个元素分别表示左上角坐标（x, y），右下角轴坐标。

最终，该方法将会返回一个含两个 tensor 的元组，元组中第一个四维 tensor（批量大小，通道数，高，宽），第二个三维 tensor（批量大小，n，5），其中 n 为单张图像中含有的边界框数的最大值，5 为保存单个边界框标注的数组长度。我们之所以需要将每张图像的边界框数目都填充到 n，是因为需要利用矩阵运算来加速小批量计算，这样需要我们每次数据输入的形状都一致。非法边界框中的物体类别为-1。

此外，为了得到一个高泛化能力、相对健壮的网络模型，我们还对图片进行了一系列的在线图像增广操作。对于训练集我们使用了随机缩放、随机裁剪、色

彩扭曲、随机翻转等图像变换方法，相比较而言对于测试集我们使用的变化方法相对简单，仅包括图像缩放和色彩变换。

4.2.2 初次训练与使用微调

本节实验中，我们选择 500 张图片作为测试集，剩下的作为训练集。为了缩短训练时间，尽快验证方法的可行性，初次训练选择较小的网络模型 MobileNet V1 作为我们的特征提取网络。

初次训练中不使用任何在其他公开图片数据集上已经训练好的网络模型参数，从头训练一个随机初始化后网络模型，将网络输入图片大小设置为 512*512，优化算法为 SGD，batch size 为 32，learning rate 为 0.01。在迭代了 100 个 epoch 后，模型在训练集上的准确度达到了 0.8478，图【4.4】为训练过程中在测试集上 mAP 的变化曲线，以及 loss 变化曲线。

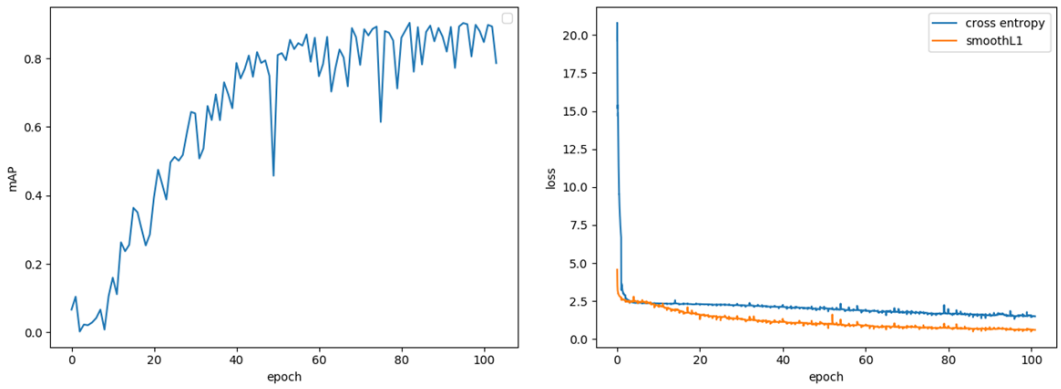

图 4.4 从头训练的 mAP 和 loss 变化曲线

可以发现在训练了一定 epoch 之后，模型在测试集上 mAP 趋于稳定，不再继续上升。并且在整个训练过程中，测试集上的 mAP 抖动明显，这可能与我们训练数据集与测试数据集结构不太合理有一定关系。最为关键的是从头训练网络模型将会花费很多时间，并且最终效果还不一定符合我们的预期。

为了解决以上问题，我们使用迁移学习（transfer learning）的方法，该方法可以大大缩短训练时间，并且训练出的模型在测试集上的表现也更好[21]。

在实际应用深度学习时，我们常常会碰到，因为数据集规模太小，涵盖的场景不够丰富而导致模型在训练集上过拟合。针对这一问题，最容易想到的办法是进一步收集更多训练数据，扩充训练数据集。可是，搜寻数据再为数据添加标签

会耗费可观的人力和财力，比如 ImageNet 数据集的制作耗费就高达百万美元。

另一种可行的办法是使用迁移学习，将模型在其它数据集学习到的知识迁移到目标数据集上，图【4.5】展示了迁移学习的策略。我们之所以可以这样做，因为在源数据集上训练的模型能够学习到一些共同的图像特征，这些共同的边缘、纹理等特征对于识别我们感兴趣的物体也许一样会有帮助。

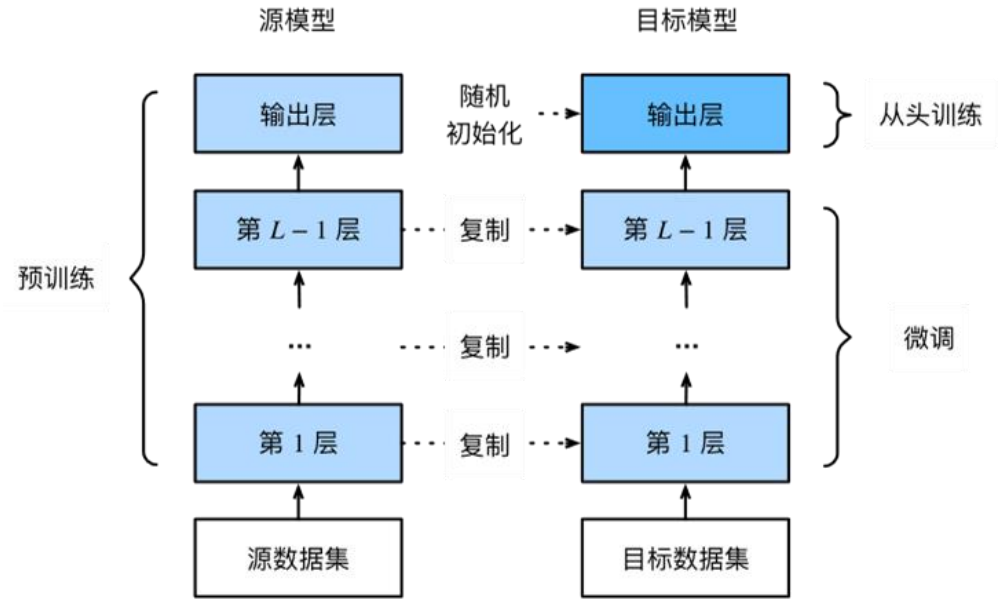

图 4.5 迁移学习

应用迁移学习的一种常用技术手段是微调，一般由以下一些步骤组成：

1. 在更大的数据集上进行源模型的预训练，源数据集一定要足够大，并且覆盖多样的场景。

2. 通过改变源模型的输出层得到目标模型，这样保留了其余所有层的模型参数，而其中就蕴含着源模型在源数据集上学习到的共同图像特征。目标模型的输出层数目为我们要预测的物体类别数目，选用恰当的方法初始化该层。

3. 在目标数据集上训练目标模型，目标目标的输出层将被从零开始训练，而其余层将会在训练过程中得到轻微修改。

接下来，我们将使用在 PASCAL VOC 数据集上预训练的 MobileNet V1 作为源模型，我们将源模型的输出层进行修改，将其输出个数修改为 1，我们仅关注黑烟尾气这一个对象，保留其他层的所有参数。模型输入图片大小仍为 512*512，优化算法为 SGD，batch size 为 32，learning rate 为 0.01，开始训练模型。训练结果如

图【4.6】所示，在迭代了 40 个 epoch 后，我们已经达到了 0.9051 的识别准确度，训练效率远高于从头开始训练模型。

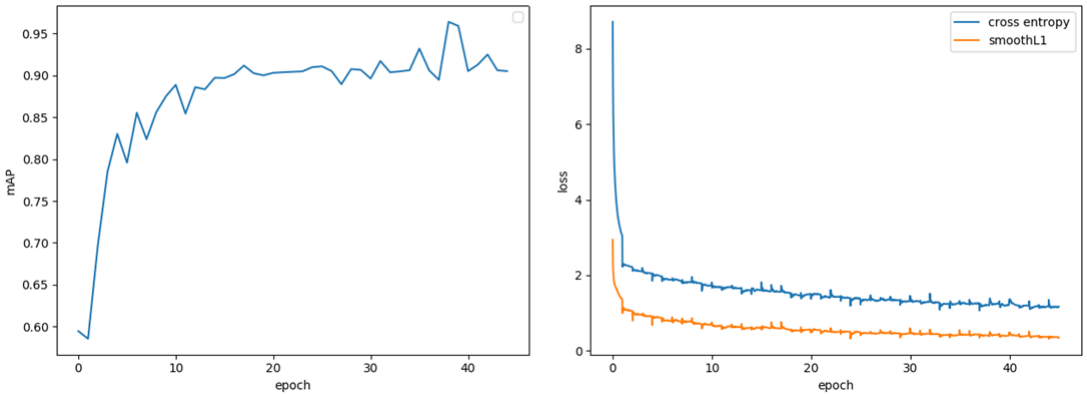

图 4.6 微调的 mAP 和 loss 变化曲线

4.2.3 基础网络模型选择

接下来我们尝试寻找最佳的基础网络模型，即我们用来提取图像特征的 CNN 网络。为了加快实验速度、节省资金，本节中将使用 Gluoncv 为我们提供了一系列训练好的模型，我们将使用 Gluoncv 在 PASCAL VOC 数据集上预训练好的 mobilenet1、resnet50、vgg16_atrous 三个基础网络模型，并使用上一节中所用的迁移学习方法，修改基础网络模型的输出层，然后以较小的学习率进行训练微调。

具体的，我们使用 512*512 大小的图像作为网络模型的输入，优化算法为 SGD，batch size 设置为 16，learning rate 设置为 0.001 分别对三个预训练好的模型进行微调，训练完 25 个 epochs 后，各模型在测试集上的 mAP 变化如图【4.7】所示：

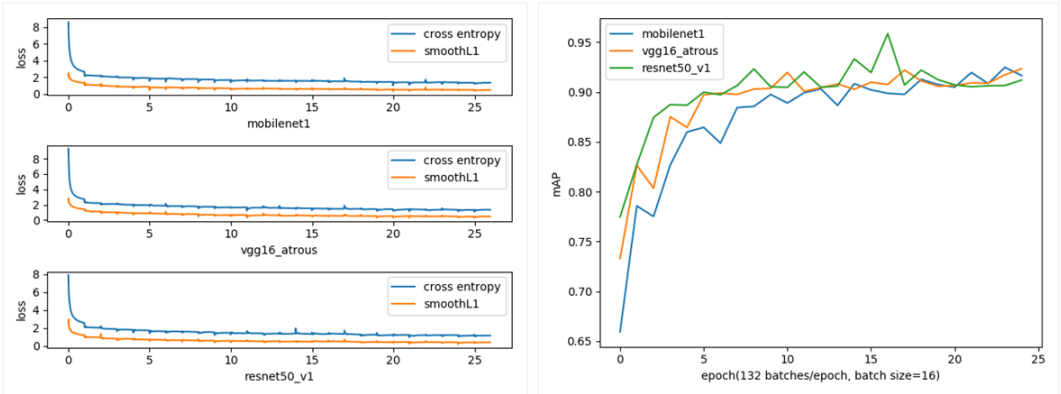

图 4.7 不同网络模型的训练过程

再结合表【4.1】可以看到，三个模型在迭代到第 25 个 epoch 左右时表现十分

接近，整个训练过程 cross entropy loss 和 smoothL1 loss 不断缓慢下降，迭代结束时也都将至一个较低的水平。

表 4.1 不同网络模型训练情况

网络模型	CrossEntropy Loss		SmoothL1 Loss		测试 mAP
	起始	结束	起始	结束	
mobilenet1	8.590	1.331	2.215	0.469	0.916
vgg16_atrous	9.312	1.342	2.584	0.464	0.921
resnet50_v1	7.899	1.139	2.502	0.387	0.924

mobilenet1 在最开始的识别准确度远低于另外二者，迭代了一定次数后，缓慢追上了另外二者；相比较而言 vgg16_atrous 的表现更加稳定，迭代一定次数后，识别准确率逐步缓慢上升；resnet50_v1 在第 16 个 epoch 时达到了最高的识别准确度，然后开始缓慢下降，并逐渐平稳，结合 loss 图像观察其 loss 变化情况可以猜测其在训练集上发生了过拟合现象。

在检测速度方面，当我们使用型号为 Intel(R) Core(TM) i7-7700HQ CPU @2.80GHz 的 CPU 时，发现三种网络模型平均都可达到 0.02-0.03s 处理完一张图片的速度，这对于实际使用已经足够，因此我们主要将检测的准确度作为基础网络模型选择的主要依据，因此我们将使用 resnet50_v1 作为我们的基础网络模型。

4.3 模型调参

在深度神经网络模型的一次训练过程中，我们选择一些个样本作为一个批量输入给网络模型，假设网络模型输出结果为 y1，真实值为 y，我们将 y1 与 y 输入损失函数可计算出本轮训练的损失，得到损失后利用反向传播算法从尾到头计算出模型参数的梯度，得到梯度后再根据学习率即可对参数进行优化。

4.3.1 最佳批大小

批大小（batch size）是小批量随机梯度下降算法中一个关键的超参数，若显存允许可将批大小设置为全体数据集的大小，利用全体数据集来计算的梯度（梯度下降法），另一个极端是只输入一个样本（严格意义上的随机梯度下降法，简称SGD），也可以进行综合，只输入一部分样本让其完成本轮训练（小批量随机梯度下降法）。

显然，在同等时间下，使用全样本的收敛速度要远慢于使用小样本的情况，全样本集每次迭代时间更长，迭代次数更少。然而在实际中，我们通常借助 GPU 进行训练，得益于 GPU 强大的并行能力以及矩阵运算加速的能力，训练一个样本与训练多个样本的时间成本几乎是一样的（磁盘读取时间仍有差距）。

通常来说，在 GPU 显存允许的情况下，我们一般尽量设置较大的 batch-size，但是 batch-size 越大，模型越有可能陷入局部最优点（local minimal）或者是鞍点。小样本将导致更大的方差，当梯度下降到 local minimal 或者鞍点时，更大的方差会使梯度在当前位置不稳定，从而有机会脱离 local minimal，从而有机会寻找更优点。

在本节中，我们进行了一系列实验，尝试寻找适合我们当前深度模型的最佳batch-size大小。我们使用resnet50_v1作为基础网络结构，learning rate设置为0.001，分别设置 batch size 为 8、16、24、32、40，使用微调的训练方法，迭代 20 个 epoch。实验结果如图【4.8】所示：

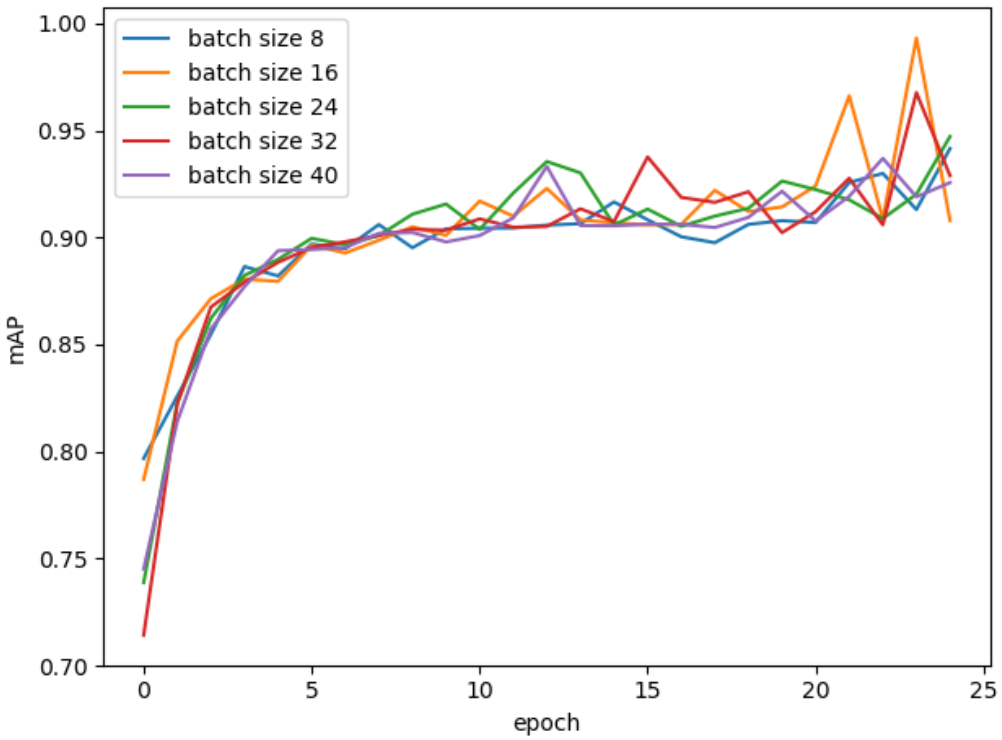

图 4.8 寻找最佳批大小

可以看到，当 batch size 为 8 时，模型并不像我们之前假设的那样拥有更高的迭代效率，其 mAP 上升速度相比较更缓慢，这可能的原因是我们数据集的样本分布不是太好，导致模型在学习通用的图像特征时有困难。

batch size 为 16 时模型在测试集上有最高 mAP，并且迭代效率相较也更加快速，其余几个 batch size 表现其实也同样优秀，最终 mAP 也都均能达到 0.92 以上，只不过 batch size 越大模型迭代后表现越稳定（mAP 不会剧烈波动）。再加上考虑到我们当我们使用大于 16 的 batch size 后，我们需要两个 GPU 才能训练模型，这就需要双倍的租用费用，所有我们将我们的最佳 batch size 选择为 16。

4.3.2 最佳学习率

学习率（learning rate）是深度模型训练中另一个关键的超参数，其值的大小直接影响模型的收敛状态，若学习率过大，每次迭代后网络参数修改幅度更改过大，将会导致损失函数震荡无法收敛；而学习率设置的过小，则会使模型收敛速度过慢，显著增加训练时间。

在本节中，我们进行了一系列实验，尝试寻找适合我们当前深度模型的最佳

learning rate 大小。我们使用 resnet50_v1 作为基础网络结构，batch size 设置为 16，learning rate 分别设置为 0.1、0.01、0.001、0.0001、0、00001，使用微调的训练方法，迭代 20 个 epoch。

实验发现，当学习率为 0.1、0.01 时，模型训练无法收敛，计算出的 loss 为无穷大（nan）。图【4.9】为当学习率为 0.001、0.0001、0.00001 时，smoothL1 loss 和 cross entropy loss 的变化曲线。

可以看到，此时当学习率越小，loss 下降越慢，符合我们的经验，0.001 的学习率优于另外二者，限于 GPU 租用费用，并且 0.001 的学习率已经可以训练出很好的深度模型了，我们没有再进一步探索其他更优的 learning rate。

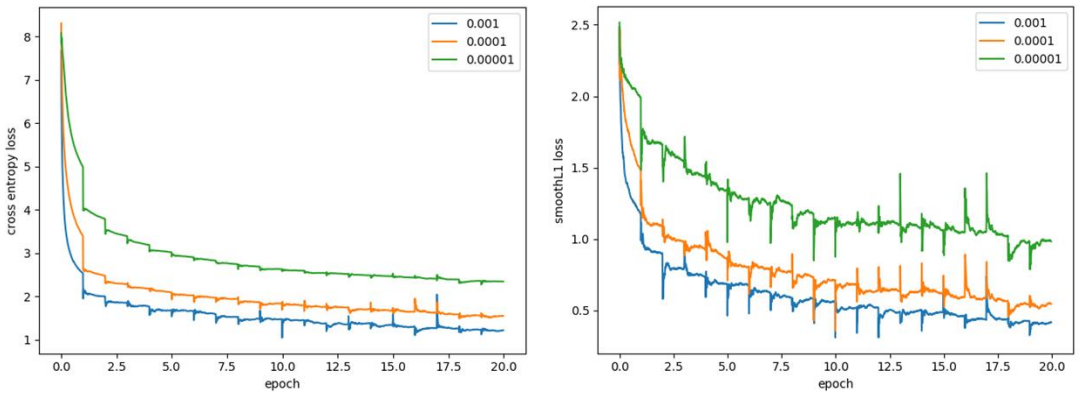

图 4.9 不同学习率时 loss 变化情况

4.4 最终检测效果

最终我们使用 512*512 的图像作为输入，特征提取网络使用 resnet50_v1，优化算法为 SGD，batch size 为 16，learning rate 为 0.001，对在 PASCAL VOC 数据集上预训练的模型进行微调，迭代了 30 个 epoch 后，得到的最佳模型在测试集上的 mAP 达到了 0.9752。

我们得到的模型在一类图和二类图上的识别效果如图【4.10】【4.11】所示，可以发现我们的模型拥有一定健壮性,对于一些较小、较远或者较淡黑烟尾气目标也都能正确识别，黑烟尾气的不同形态以及黑烟的黑度也不会对检测造成困难。

这是因为我们使用了大量的图像增广方法，因此得到了一个泛化能力很好的模型。黑烟位置预测也十分准确，这对我们之后想进一步对黑烟进行黑度测试提

供了很好的支持。

图 4.10 一类图片检测效果

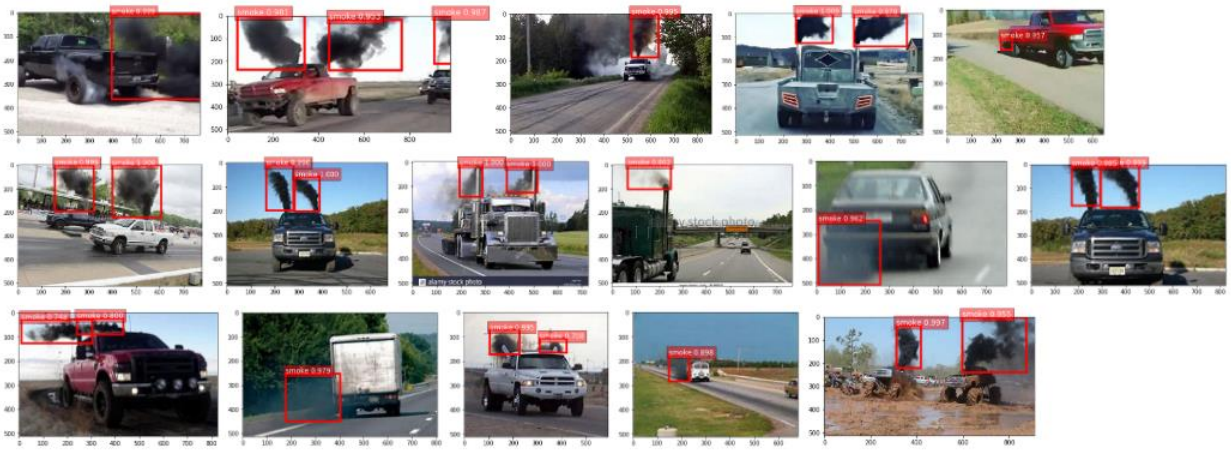

图 4.11 二类图片检测效果

同样值得注意的是，模型对黑烟尾气的检测也存在一些误识别，如图【4.12】所示。第一幅图中，由于图像拍摄时镜头抖动，导致图像存在运动模糊，模型误识别了图像中较阴暗的树。第二、三幅图，模型误识别了颜色较黑的阴影，可以发现此时模型对误识别对象置信度很较低，只有 0.5 左右。第四、五、六幅图中，模型容易误识别图像正前方的人头。

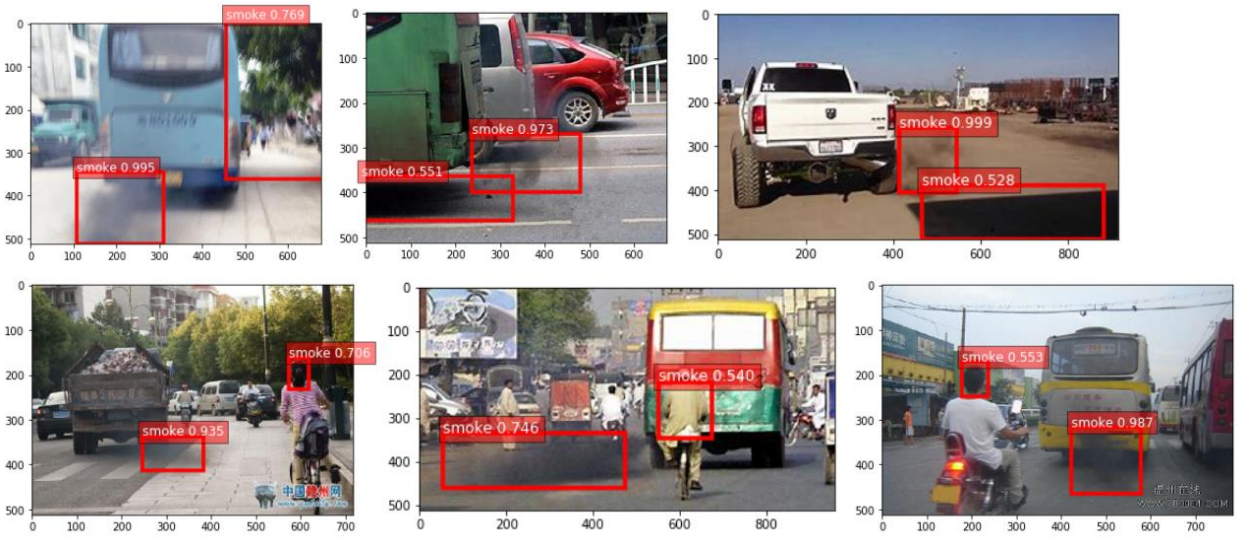
smoke 0.769
smoke 0.995
smoke 0.973
smoke 0.551
smoke 0.999
smoke 0.528
smoke 0.706
smoke 0.935
smoke 0.540
smoke 0.746
smoke 0.553
smoke 0.987

图 4.12 误识别

结论

本文尝试使用深度学习的方法解决交管部门长期以来对黑烟车整治人力成本高的问题，期望能利用主流的目标检测方法训练出一个能有效识别黑烟车并且泛化能力强的深度神经网络模型。为了完成这一目标，主要完成了如下几项工作：

1. 通过各大搜索引擎爬取了一定数量的黑烟车图片，尽可能包含了多种角度与场景，并完成了黑烟尾气的标注工作，运用图像增广的方法对数据集进行了扩充，制作了 PASCAL VOC 目标检测格式数据集，划分出训练集图片 2105 张，测试集图片 500 张。

2. 学习并比较了目标检测领域具有代表性的三类算法 Faster R-CNN、YOLO、SSD，并根据实际情况，权衡检测速度与精度之后，选择了 SSD 作为当前任务的目标检测算法。

3. 比较了主流深度学习框架，选择了文档丰富、教程连贯的 MXNet 作为实验所使用的深度学习开源框架。

4. 在使用 mobilenet1 进行初次训练的过程中，发现直接从头训练网络耗费时间长，并且最终识别准确率也不理想。采用了迁移学习后，通过在预训练模型上进行微调，仅需要迭代很少的次数就可以得到很好的模型。

5. 对比了目前计算机视觉领域流行的几个神经网络在目标数据集上的检测效果，比较分析了它们训练的过程与最终训练的结果，最终确定了将 resnet50_v1 作为我们的基础特征提取网络。

6. 尝试调整了小批量随机梯度下降算法 batch size 和 learning rate 超参数，根据实验情况，找到了相对较优的参数，最终确定了实验的 batch size 为 16，learning rate 为 0.001。

7. 最终使用 SSD 目标检测算法，特征提取网络使用 resnet50_v1，小批量随机梯度下降的 batch size 为 16，learning rate 为 0.001，对在 PASCAL VOC 数据集上预训练的模型进行微调，迭代 30 个 epoch 后得到的最佳模型在测试集上的 mAP 为 0.9752。

本文实验还存在诸多不足，多方面都有待改善：

1. 数据集还有较大的提升空间，可以加入更多的来自实际应用场景的图片，涵盖更多的天气、夜晚、远景图片。

2. 针对不同的应用场景，可以发掘各应用场景的特点，针对这些特点对算法进一步改进，提高模型检测能力。

3. 实际应用中，黑烟车检测是基于视频的，视频相较于图像，多了时间维度的信息以及更多的上下文相关性，这可以进一步对模型进行扩展。

参考文献

[1] 海康威视. 海康威视黑烟车智能电子抓拍系统上线[EB/OL]. http://www.afzhan.com/news/detail/69283.html, 2018-09-07

[2] 生态环境部网站. 专家解读《打赢蓝天保卫战三年行动计划》[EB/OL]. http://www.gov.cn/zhengce/2018-07/06/content_5303964.htm, 2018-07-06

[3] 江绮鸿, 李文松. 黑烟车智能监控系统的应用[J]. 化工管理, 2018, No.493(22): 97-98.

[4] 卷积神经网络研究综述[J]. 计算机学报, 2017, 40(6):1229-1251.

[5] Yann LeCun. LeNet-5, convolutional neural networks [EB/OL]. http://yann.lecun.com/exdb/lenet/, 2019-05-01

[6] Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E. ImageNet classification with deep convolutional neural networks[J]. Communications of the Acm, 2012, 60(2):2012.

[7] Simonyan K , Zisserman A . Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.

[8] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. 2015:770-778.

[9] Andrew G. Howard, Menglong Zhu. MobileNets: Open-Source Models for Efficient On-Device Vision [EB/OL]. https://ai.googleblog.com/2017/06/mobilenets-open-source-models-for.html，2017-06-14

[10] 尹宏鹏, 陈波, 柴毅,等. 基于视觉的目标检测与跟踪综述[J]. 自动化学报, 2016, 42(10):1466-1489.

[11] Lars Hulstaert: A Beginner's Guide to Object Detection [EB/OL]. https://www.datacamp.com/community/tutorials/object-detection-guide，2018-04-19

[12] Prince Grover: Evolution of Object Detection and Localization Algorithms [EB/OL]. https://towardsdatascience.com/evolution-of-object-detection-and-localization-algorithms-e241021d8bad，2018-02-14

[13] Dhruv Parthasarathy: A Brief History of CNNs in Image Segmentation: From R-CNN to Mask R-CNN [EB/OL]. https://blog.athelas.com/a-brief-history-of-cnns-in-image-segmentation-from-r-cnn-to-mask-r-cnn-34ea83205de4，2017-04-22

[14] Girshick R. Fast R-CNN[C]. IEEE International Conference on Computer Vision. 2015.

[15] Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks[J]. 2015.

[16] Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[C]// Computer Vision & Pattern Recognition. 2016.

[17] Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger[C]// IEEE Conference on Computer Vision & Pattern Recognition. 2017.

[18] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector[C]// European Conference on Computer Vision. 2016.

[19] Tarang Shah: Measuring Object Detection models—mAP—What is Mean Average Precision? [EB/OL]. https://towardsdatascience.com/what-is-map-understanding-the-statistic-of-choice-for-comparing-object-detection-models-1ea4f67a9dbd，2018-01-27.

[20] PASCAL VOC: The PASCAL Visual Object Classes Homepage [EB/OL]. http://host.robots.ox.ac.uk/pascal/VOC/，2018-01-27.

[21] Pan S J, Qiang Y. A Survey on Transfer Learning.[J]. IEEE Transactions on Knowledge & Data Engineering, 2010, 22(10):1345-1359.