摘要

目标检测技术在深度学习和神经网络相关研究的高速发展的背景下，成为了具有优秀前景和应用价值的技术。以无人机航拍图像识别为背景，在复杂场景下为了实现轻量化的要求，采用 You Look Only Once 算法（简称 YOLO）作为基础的目标检测模型。YOLO 是最流行的单阶段目标检测模型，而经过不断迭代，YOLOv5 是目前发展比较成熟的 YOLO 模型的版本。YOLOv5 模型应用了多尺度检测，改进的 bottleneck CSP 层（C3 层），并且使用了 SPPF 作为瓶颈层，改进了损失函数等。为了让目标检测网络模型可以在低算力的嵌入式平台上运行并达到试试检测的标准，本项目通过使用更轻量化的卷积，比如 GHOST 卷积，以及知识蒸馏等方式来减少模型的参数量，达到轻量化的目的。而为了让模型面向具体项目时能够有更加优秀的表现，在调整模型的同时还尝试添加额外的特征提取层和辅助目标检测或者目标分类的模块来帮助模型得到更好的结果，如 Transformer 模块。最后通过目标检测模型的指标来对模型进行评价。

关键词：目标检测，YOLO 模型，算法优化，模型轻量化

Abstract

With the rapid development of deep learning and neural network related research, object detection technology has become a technology with excellent prospects and value. In order to achieve lightweight requirements in complex scenarios, the You Look Only Once algorithm (YOLO) is used as the basic object detection model based on the recognition of UAV aerial images. YOLO is the most popular single stage object detection model, and after continuous iteration, YOLOv5 is currently a mature version of the YOLO model. YOLOv5 model applies multi-scale detection, improved bottleneck CSP layer (C3 layer), SPPF as the bottleneck layer, improved loss function, etc. In order to achieve the object detection network model to run on the embedded platform with low computing power and meet the standard of real time detection, this project uses lighter convolution, such as GHOST convolution, and knowledge distillation to reduce the number of parameters of the model, so as to achieve the purpose of lightweight. In order to make the model more project-oriented, we try to add additional feature extraction layers and auxiliary object detection or object classification modules to help the model get better results, such as the Transformer module. Finally, the model was evaluated by the indicators of the object detection model.

Keywords: object detection, YOLO, improved algorithm, lightweight model

目录

1 绪论

1.1 引言

目标检测问题是目前计算机视觉方面的热点问题，这是因为完备的目标检测技术可以应用的工业领域广泛。随着目前对于流程的自动化和智能化发展的需求越来越迫切，目标检测技术通过赋予机器智能感知能力令其有潜力替代各行各业传统流程中的很大一部分人工流程。令企业达到减少人力成本，提高生产效率，提高生产流程安全性等目的。目前，对于目标检测技术的研究已经有了非常多的成果和技术，但是目标检测技术在不断取得进展的同时，对于目标检测的效果和本身检测标准的要求也有了进一步的提升。同时，目标检测在实际应用时也会因为应用场景的变化而产生新的问题。首先是目前许多实际应用的需求都需要目标检测可以在复杂场景下达到需要的精度和效率。其次是目前能够运行目标检测程序达到令人满意的精度和效率的设备成本较高。因此如何让目标检测技术克服场景的多变性和降低技术对设备算力的要求就成为了目标检测领域比较关键的难题。

本文以无人机航拍图像识别为背景，考虑到无人机航拍图像环境复杂，且无人机本身因为功耗等原因算力较低，因此将在达到识别复杂场景的检测精度的同时，降低模型参数量，使得目标检测模型更加轻量化，从而达到无人机实时目标检测的目的。

1.2 研究背景

近年来，人工神经网络的兴起让很多之前由于计算量过大或者样本过少的问题得到了一定程度的解决。而图像识别和目标检测领域可以说是其中进步最为明显的领域。通过训练神经网络模型可以以极高的准确度来识别出某类特定物体。目标检测可以应用于目前比较新兴的大数据分析领域，训练好的神经网络可以识别大量的图片从而得出需要的数据。比如某时段某地街道行车数量或者行人流量的统计。目标检测同样也可以应用在人脸识别领域，通过在火车站、飞机场等地的闸机设置人脸识别系统来验证身份，可以提高身份验证的效率。在物流领域，目标检测也同样被大量应用于物流快递自动分拣。由此可见，目标检测技术是可以应用于多个领域的热门技术，对于目标检测技术的研究是对于现实生活的改善有巨大推动作用的。目标检测网络的功能主要有两个，第一个是将目标用方框或者是轮廓线标定出来，第二个是将标记的目标进行检测并分类。

在目标检测领域中，使用轻量化网络去检测复杂场景中的目标是较为困难的，目前主流的方法往往在精准度、速度和设备成本上还无法达到令人满意的水平。因为现实场景中会有很多干扰，如亮度变化，障碍物遮挡，与背景相似的颜色导致无法识别等问题，而一般的轻量化网络往往难以排除以上问题，从而导致精准度或者速度两者无法兼具。因此对于复杂场景的轻量化目标检测网络的研究是有必要的。

1.3 研究意义

目前的目标检测领域技术的主要应用方向有汽车自动驾驶，人脸识别，区域自动监控等方面。目标检测模型通过分析汽车外部传感器实时传入的环境图像来检测路牌内容，前方车辆距离，附近的障碍物和行人等。在机场火车站这类场所对于验证身份有较高需求，目标检测模型通过在闸口进行高效率人脸识别，来减少身份验证方面的人力资源投入。在一些企业或者场所的重要区域，目标检测模型可以在无关人员进入该区域时立即检测到人员位置并发出报警。可以发现上述目标检测技术的应用都偏向于实时目标检测，并且运行模型的平台往往是嵌入式平台，因此对于计算量要求比较高，或者检测速度较慢的目标检测模型是很难完成这方面的工作的。

YOLO 算法让实时目标检测成为可能，进行过轻量化操作的 YOLO 算法网络模型的参数量较少，可以搭载到低功耗的嵌入式平台进行实时目标检测，这让目标检测网络的应用领域更加广阔，所以在考虑建立面对复杂场景的轻量化目标检测网络模型时，YOLO 算法是非常好的选择。

1.4 目标检测以及研究方法

1.4.1 目标检测基本概念

目标检测[1] 是一种涉及到计算机视觉和图像处理的技术，一般通过图像中物体的特征来识别并分类图像中的实例，特征是图像中被计算机所提取的用来体现目标的一部分属性的信息，基于分类的精度要求不同，对于不同物体的特征的选择也会随着实际变化。比如在分类车辆和行人时的特征依据，小轿车和货车的特征依据，不同型号的轿车的特征依据都是不同的。并且，为了达到更高的准确性，目标检测在分类不同物体时的特征依据也不唯一，每个物体往往拥有多个相对于其他物体而言独特的特征。目标检测的原理在于提取当前需求分辨的物体集合之中不同物体的独特的特征来帮助进行物体的分类，并根据提取的特征对物体加以分辨。

1.4.2 目标检测的研究方法

目前目标检测方法有无神经网络方法和通过神经网络进行目标检测的方法。对于无神经网络方法而言，一般使用一些方法来定义不同物体的特征，然后通过支持向量机（SVM）来进行分类。而通过神经网络进行目标检测的方法是一种端到端的目标检测方法，不需要人工定义物体的特征，这些方法都是基于卷积神经网络（CNN）进行的。无神经网络目标检测方法主要有 Viola-Jones 通用目标检测框架[2] 尺度不变特征变换（SIFT）[3] HOG 特征（Histograms of oriented gradients）[4] 等。通过神经网络进行目标检测的方法则有基于候选区域的 R-CNN[5]，Fast R-CNN[6]，Faster R-CNN[7] 等，以及 SSD[8]（Single Shot MultiBox Detector），YOLO[9]（You Look Only Once），Retina-Net[10]，可变形卷积网络[11]（Deformable convolutional networks）等。

1.5 国内外研究现状

1.5.1 传统目标检测网络

传统目标检测网络主要有三个步骤组成，分别是区域选择，特征提取和分类器分类。

区域选择：区域选择策略在传统目标检测策略中通常采用滑动窗口的方式进行目标检测，从理论上讲这种穷举的方式可以覆盖到输入图片上的每个区域，然而这种滑动窗口模式的时间复杂度太高，会产生很多冗余窗口，因此所选择的滑动窗口尺寸是固定几个尺寸的。如果图片中的目标尺寸变化幅度过大时，可能无法选择出合适的滑动窗口的尺寸，并且在滑动窗口阶段可能遍历完图片的所有区域后仍然难以找到合适的窗口位置。

特征提取：传统目标检测方法主要通过滤波器提取图片的颜色特征，纹理特征，形状特征和空间关系特征，然后将提取的特征转换为向量形式，然后送入分类器中。因为目标本身就具有多样性，再加上背景和光照的变化，这导致提取的特征难以与实际图片中的目标所匹配。因此传统目标检测方法的特征提取过程鲁棒性较弱。

分类器：经过有监督训练之后的分类器对向量进行分类，最后输出结果。

传统目标检测过程步骤图如图 1-1

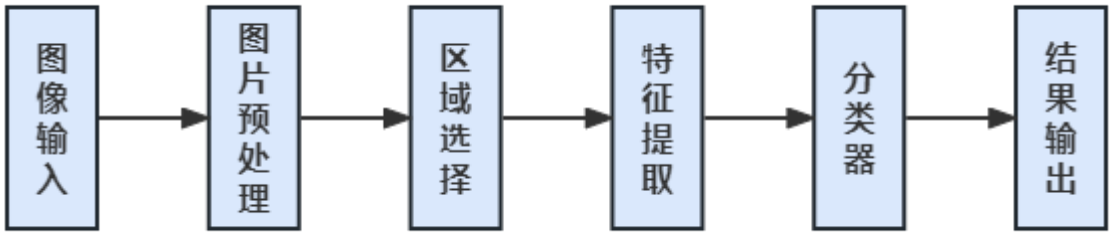


图 1-1 传统目标检测过程示意图

1.5.2 基于深度学习的目标检测网络

（1） 基于区域的卷积神经网络（R-CNN）

R-CNN 目标检测网络基于先将图片中的目标筛选出来，然后再对目标进行分类的思想，这种方法相比其他基于深度学习的目标检测网络具有更高的精度，并且可以做到实例分割这种对于同类物体进行精细分割的功能，但是由于效率的原因更多用于单次的目标检测。

王艳等[12] 提出了基于 Faster R-CNN 的算法，用于辅助显微镜图像诊断，在原本 Faster R-CNN 算法的基础上添加了卷积滤波器和残差网络，并运用了一些数据增强手段。

龙洁花等[13] 提出了对于番茄果实成熟度的检测方法，采用了网络融合的方法将 CSPNet（跨阶段局部网络）和 ResNet[14] （残差网络），进行跨阶段拆分和级联，最终有效地提高了准确率和降低了模型计算量。

（2） YOLO

在 YOLO 模型被提出之后，基于 YOLO 模型的各种目标检测方面的应用在国内外

研究人员的共同努力下提出了许多 YOLO 模型的变体。

比如有 Martin Simon 等[15] 提出的用于汽车自动驾驶的基于激光雷达的三维物体检测，就是基于 YOLO 模型进行了修改提出了 Complex-YOLO，这是一个适用于实时点云的 3 维目标的 E-RPN 网络模型。在构造出 YOLOv2 的扩展网络后通过特有的的复杂回归策略来测量笛卡尔空间中的多种类的三维物体。因为 YOLOv2 仅仅支持二维平面图片的目标识别，因此只输入来自雷达的点云图像并转换为鸟瞰图，然后通过 YOLOv2 进行识别后建立与地面垂直的三维锚框来标定汽车上搭载的激光雷达所扫描出的障碍物。

Xingkui Zhu[16] 等使用改良的 TPH-YOLOv5 模型用于无人机俯视角拍摄的图片的目标检测，当无人机在不同海拔高度飞行时，同一物体在图像中的尺度大小变化会变得非常显著，这会给神经网络的学习带来负担。而且高速低空的飞行会导致密集物体的成像变得模糊，也会给物体辨识带来巨大的挑战。他们使用了 TPH（Transformer prediction head）检测头来代替原本的检测头。同时他们集成了卷积注意力模块（CBAM）以便在大场景中寻找密集物体的注意力区域，而且对于小目标检测，他们额外添加了一个在传统目标检测网络中经常被使用的目标分类器，用于增强模型在相似类别之间进行区分的能力。

自从 YOLOv5 被 Jocher G[17] 等提出后，YOLOv5 在不断完善之下逐渐成为目前最流行的单阶段目标检测网络模型。这是因为 YOLOv5 可以通过修改模型宽度和模型深度两个参数来调整模型的复杂度。

1.6 研究内容和安排

目标检测模型因为图像目标特征多，为了达到较高的精度，往往模型参数量大，浮点运算量高，普通的嵌入式平台难以搭载目标检测模型。因此，本文针对该现象，对比研究了较多经典的目标检测算法的特点以及性能之后，选择了 YOLOv5s 为基准模型，通过对其网络结构进行进一步的轻量化，同时尽可能减少检测性能上的损失。最后将改进后的模型在 VisDrone2019[18] 数据集中进行性能上的验证。主要研究内容如下：

（1）为了减少模型的参数量，将模型中的卷积模块替换成 Ghost 卷积，使网络更加轻量化。并且 Ghost 卷积使用线性变换可以产生和普通卷积相同数量的供网络学习的冗余特征图，这样可以保证训练后的网络性能不会损失过多。

（2）为了让背景环境光干扰和图片中的噪声干扰降低，让模型提高对于小目标的关注度，在 YOLOv5 的瓶颈层中增加了 Transformer 模块，使目标检测模型可以提高对于目标本身的关注，更好的处理学习到的特征信息，同时抑制不相关信息的干扰，把注意力放在目标的关键特征上，并且增强了对于小目标特征信息的提取，也提高了整体的目标检测精度

（3）为了进一步提高轻量化模型的准确度，让浮点运算数减少的轻量化模型拥有更好的表现，训练了 YOLOv5m 模型并使用知识蒸馏（Knowledge Distillation）方法将该

模型的知识迁移到 YOLOv5s-Ghost 中，令其更好的学习训练集的特征，增强信息提取，从而提高轻量化模型的精度。

（4）出于航拍图像中目标的普遍特性，为了更好的识别航拍图像的小目标，并且适应航拍图像中目标的尺度变化范围大，额外添加了一个小目标检测框，让聚类生成的框具有更多种尺度。另外额外进行一次上采样并添加一个检测层来更好的对图片中的小目标进行检测。

2 相关理论基础

2012 年，AlexNet[19] 的出现让深度神经网络在深度学习领域中崭露头角。由于深度神经网络的学习能力远高于之前所提出的算法，深度学习在自然语言处理，音视频处理，目标检测等方面获得了长足的进步，推动了目标检测，图像分析，自然语言处理，音频生成等领域的发展。而 AlexNet 正是基于卷积神经网络达到了极高的运算效率和分类精度。

2.1 卷积神经网络

卷积神经网络主要由卷积层，池化层和全连接层组成，是一种具有深度结构的前馈神经网络。如图 2-1 为卷积神经网络结构示意图，数据规模从$8 \times 128 \times 128$降低到了256。

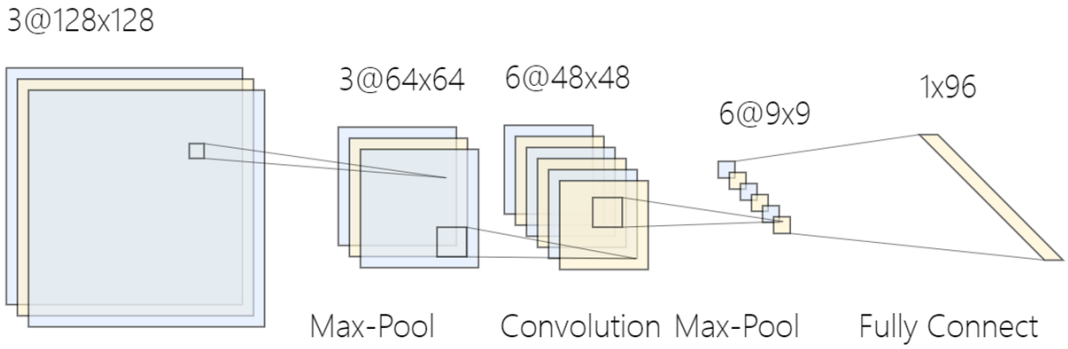


图 2-1 卷积神经网络结构示意图

卷积层：卷积层是将输入数据进行特征提取，如图 2-1 所示，卷积层将原本矩阵中的所有规定大小的子矩阵通过卷积操作按顺序生成代表这个子矩阵特征的输出。如此一来，每次卷积操作的输出就代表了这个子矩阵的特征，同时因为所有的子矩阵都要进行卷积操作，所以卷积操作还强调了矩阵内所有子矩阵的关联性。子矩阵的大小被称为卷积核（Convolution Kernel）。

池化层：由于深度神经网络输入的数据规模大，为了保证网络训练的快速性，同时防止网络发生过拟合的线性，通过池化操作来提取数据中最有代表性的特征。在神经网络中一般使用最大池化，即将矩阵分割成规定的矩阵大小，然后取分割出来的每个子矩阵的最大值赋值给子矩阵的其他值。这本质上降低了数据的维度，同时可以提高深度神经网络的鲁棒性。

全连接层：直接将上一层的输出映射到一个一维向量上，全连接层整合了网络所提取的一组数据的局部特征，神经网络可以通过全连接层输出的数据来判断输入数据的种类和特征等。

2.2 目标检测网络算法

目标检测网络主要分为两阶段目标检测算法和单阶段目标检测算法两种，前者是先通过算法生成图片中需求检测的目标的候选框，再通过卷积神经网络进行目标分类。后者则不生成候选框，将两个步骤并作一步，直接把目标边框定位的问题转化为回归问题，即应用模型训练得到的知识检测到目标的特征并标定。正是由于两种方法的差异，这两种检测算法在性能上也具有较大差异。前者在检测准确率和定位精度上占优，后者在算法速度上占优。

2.2.1 两阶段目标检测网络

两阶段目标检测网络以基于区域的卷积神经网络（R-CNN）为代表。两阶段目标检测网络是先提取目标区域，然后再对区域进行分类识别，两个步骤分开执行，相比单阶段目标检测网络在准确性方面具有极大的优势，两阶段目标检测网络几乎都是由 R-CNN 网络改进而来，并以此为基础不断改进，Fast R-CNN[6] 在 R-CNN 网络的末端使用了 ROI Pooling（Region of interest Pooling）进行分类，这让目标检测网络能更快的训练和检测目标，提高了 R-CNN 的检测速度。而 Faster R-CNN[7] 将 ROI 生成层直接集成到了神经网络之中，进一步加快了检测速度。随后出现的 Mask R-CNN[20] 和 Mesh R-CNN[21] 也都是基于 R-CNN 进行的改进，其中 Mesh R-CNN 是可以进行三维目标检测的网络。然而这些目标检测网络都需要扫描两次图片才能依次完成检测目标和对目标进行分类两个任务，因此多用于处理数据，难以达到实时检测的效果。

2.2.2 单阶段目标检测网络算法

为了解决目标检测的实时性问题， YOLO（You Only Look Once）模型应运而生，YOLO 是正如其名字一样，是一个典型的单阶段网络，每张图片只需要扫描一次就可以检测到目标并分类。 YOLO 是一种广泛使用的算法，也是目前最流行的单阶段目标检测算法。2015 年，Redmon 等提出了 YOLO 的第一个版本。在过去的几年里，目标检测领域的研究人员不断提出 YOLO 改进版本，分别是 YOLO V2、YOLO V3、直到 YOLOv8。YOLO 的版本一直在改进和更新，每个版本更新作者都优化了模型的精准度和运算效率。YOLO 模型摒弃了卷积神经网络复杂的框架，直接使用回归的方法来输出结果。[22]

YOLOv1 将图片分割成 7*7 的网格，然后直接预测每个格子中可能存在对象的边界框，并且输出框中所含有的对象分类的概率。YOLOv1 默认有二十种或者二十种以内数量的物体分类，每个网格预测两个边界框，同时输出两个边界框的置信度，这样 YOLOv1 输入一张 448x448x3 的图片后在输出端输出 7*7 个三十维的向量（其中包括两个边界框的长宽和边界框中心的坐标轴位置共 8 个数据，属于二十种物体中的任意一种的概率共 20 个数据，以及两个边界框的置信度一共 2 个数据）。其中置信度的计算公式见式 2-1。

$$\text{Confidence} = \Pr(Object) * IOU_{pred}^{truth} \tag{2-1}$$

SSD[8] 也同样是单阶段目标检测网络，其中单次检测指明了 SSD 算法属于单阶段方

法，并且为多尺度预测模型，这让 SSD 的小目标检测能力得到了巨大的提升。SSD 直接采用 CNN 进行检测，而 YOLOv1 在全连接层之后再进行检测。因此 SSD 的效率要高于 YOLOv1。同时 SSD 率先采用了多尺度检测，先验框输入等方式，因此 SSD 对比 YOLOv1 具有更加优秀的平均精准度（mAP），随后 YOLO 目标检测模型也同样采用了 SSD 的一些目标检测方面的思路。

YOLOv2 采用了 darknet[23] 作为主干特征提取网络，框架如图 2-2：其中有 19 个卷积层和 5 个最大池化层，主要采用 3*3 的卷积和 1*1 的卷积。相比 YOLOv1，这样的主干网络并不能提高精确度，但是可以提升网络的训练效率。

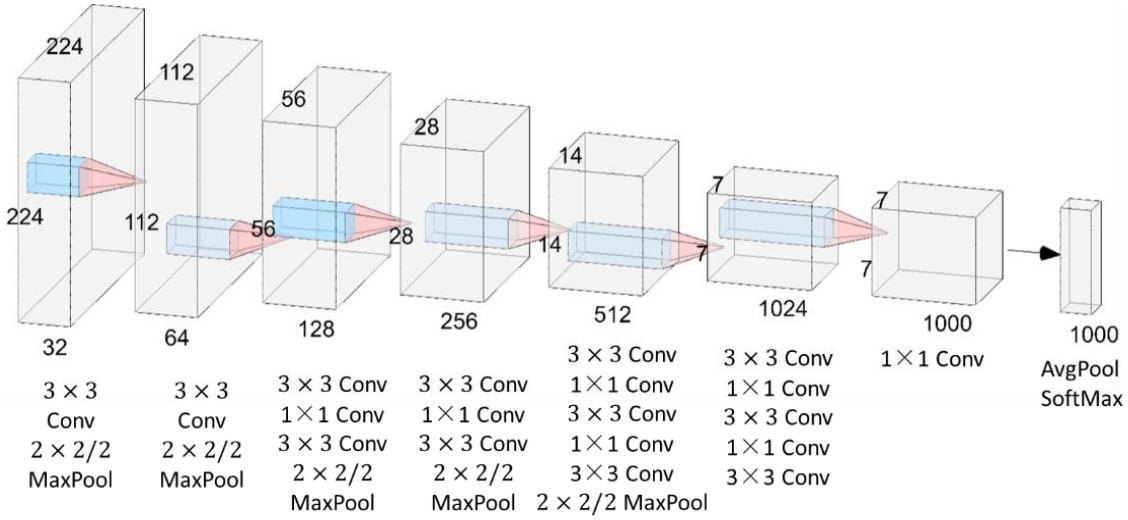


图 2-2 DarkNet 网络结构

相比原本的 YOLOv1，YOLOv2 的主干网络去除了全连接层和一个池化层，减少一个池化层可以让模型识别更高分辨率的图片，而去除全连接层可以减小模型训练时的计算量，并且直接使用带锚框的卷积来更好地对物体进行检测，这是因为卷积的特性让模型更加关注图片的中心点，而往往图片的中心就是主要目标所在的位置。YOLOv2 采用了如下手段对带来了进一步优化。

（1）批归一化：YOLOv2 将数据放缩或者偏移来避免卷积层输出的数据分布发生变化，实质上就是多出了γ和β参数来负责正态分布归一化后的放缩和偏移，这个参数是通过学习得到的。ϵ是无穷小参数，防止方差为 0 的情况发生。在添加了批归一化层来代替丢弃层后，能提升模型收敛的速度，并且可以防止模型产生过拟合现象。公式如下：

$$x^* = \frac{\gamma(x-\mu)}{\sqrt{\sigma^2+\epsilon}} + \beta \tag{2-2}$$

（2）提高预训练分类网络的分辨率：因为 YOLOv2 的模型输入为 448*448，如果按照主流方法使用小于 256*256 尺寸的图片在 ImageNet 上进行预训练，模型会不适应高分辨率图片的输入，因此 YOLOv2 的特征提取器是在 ImageNet 上以 448*446 的分辨率对网络进行预训练来避免 YOLOv2 灵敏度降低的问题。

（3）使用 k-means 聚类学习生成先验锚定框：相比 v1，v2 不预测边界框的大小，

而是预测与生成的最接近正确答案的锚定框的偏差，使用 k-means 聚类分析生成的先验框数据要比人手动设定的先验框大小更加客观，平均的重合度也会提升。这样只通过微调锚定框能极大减少训练时的计算量。比如当目标检测任务中某一类为站立的人时，增加聚类产生的锚定框可以让生成的框相对较高且细，可以让模型更加容易学习。因此如果有 20 种类别需要分类时，k-means 就需要根据训练集生成二十个不同尺寸的锚定框。

（4）绝对位置预测策略：因为是预测锚定框的偏移量，而且没有对偏移量进行限制，这样会让模型的计算效率大大降低，因此 YOLOv2 限制了锚框的中心点的最大偏移不能超过 7*7 的网格边缘。

（5）细粒度特征:YOLOv2 将图片进行特征重排，将大尺寸图片行列隔点采样，有利于小目标检测。

YOLOv3 升级了 v2 中的 darknet 网络结构[24]，改为全卷积神经网络，去掉了所有的池化层。每个卷积层后接着批归一化层，并且使用 Leaky Relu 作为激活函数。为了防止 53 层卷积层所导致的梯度消失问题，YOLOv3 在每组 1*1 和 3*3 卷积层后增加了残差层。网络结构如下图 2-2：

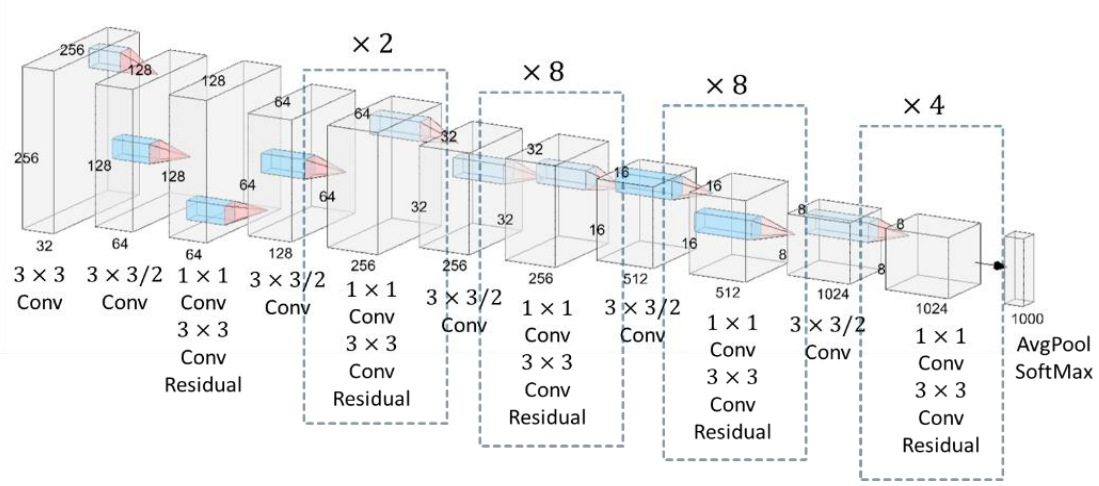


图 2-2 YOLOv3 网络结构

YOLOv3 的多尺度检测：多尺度检测是指输出经过不同层数的分辨率的图片，在 YOLOv3 中，会检测三种不同采样倍数的图片，采样倍数越小，分辨率越大，语义信息越弱，但是更加适用于小目标检测，采样倍数越大，语义信息越强，适用于较大目标的检测。最后将三种检测结果进行张量拼接。

之后的 YOLO 在迭代中增加了很多数据增强手段，比如 Mosaic 数据增强，Copy paste（主要用于实例分割），随机仿射变换（(Random affine(Rotation, Scale, Translation and Shear)），改变透明度并进行图片融合（MixUp），滤波，均衡化（Albumentations），随机调整色度和饱和度（Augment HSV(Hue, Saturation, Value)等策略。

YOLO 在不断迭代的过程中同样微调了诸如损失（loss）计算公式，以 Darknet 为基础修改了部分网络结构，同时增加了许多可选的模块，这些模块在原理上都有提升模型表现，改进模型精度或者降低模型参数量的特点，然而针对不同的数据集，或者是在不

同的应用场景中，有选择性地加入这些模块可以有效地提升模型的性能，比如使用 SPPF（空间金字塔池化）的瓶颈层增大了模型感受野，增加 CBAM[25]，SE[26]，SAM[27] 等模块来引入注意力机制，又或者使用 PAN[28]，BiFPN[29] 进行特征集成，用 Swish、Mish 来替换原本的 SiLu 激活函数，用 soft NMS[30]（Non-Maximum Suppression）或者 DIoU NMS 等方法来改进处理方法改进。

2.3 本章小结

本章主要借助主流的目标检测算法介绍了目标检测算法的理论基础。目标检测网络算法分为两阶段目标检测网络和单阶段目标检测网络。其中两阶段目标检测网络精确度更高，而单阶段目标检测网络则具有更好的检测效率，更容易在实时性方面达到要求。单阶段目标网络中最为常用的就是 YOLO 算法，YOLO 算法经过了多次迭代，现在成为了一种非常灵活的目标检测算法，其模型的泛用性和针对特殊情况的性能改进策略都有了很大程度的提升和积累，属于在技术上发展较为成熟的单阶段目标检测网络模型。

3 基于 YOLOv5 的轻量化目标检测网络

YOLO 这一系列算法一直以来因为检测速度快，模型轻量化而被广泛使用。YOLO 每一次迭代都显著的提高了算法的综合性能，因此选择 YOLOv5 模型针对无人机航拍图像场景目标检测模型轻量化进行改进。为了减少模型的计算量，在研究中主要从令模型计算步骤轻量化入手，同时为了改善轻量化模型的准确度，提出了加入 Transformer 模块来引入注意力机制，将大模型的知识迁移到轻量化模型上，加入额外的检测层，这三种方案。

第一种方案主要针对航拍图像数据集噪声多，背景因素干扰强的特点，利用注意力机制来更好的提取图像中的关键特征，提高模型对于目标的注意力。

第二种方案则是通过知识蒸馏的方式提高模型的识别精度，同时增加模型的泛化能力。

第三种方案是通过增加上采样层和检测层的方式来更好的提取到图像中小目标的特征。

3.1 YOLOv5 模型选择

YOLOv5 模型以 YOLOv5l 模型为基准，较小的模型拥有更小的模型深度和模型宽度。可以在模型结构中设置 depth_multiple 参数和 width_multiple 参数的数值来修改模型的大小。其中 depth_multiple 参数用于控制网络中部分子模块的数量，来修改模型的复杂度。width_multiple 参数用于修改模型的通道数量，同样，通道数的变化也会改变模型的复杂度。对于航拍图像识别项目，需要让模型尽可能轻量化，因此选择 YOLOv5s 或者 YOLOv5n，然而 YOLOv5n 基准模型虽然更加轻量化，具有更快的推断速度，但是识别精度过低。因此最后选用了网络深度为 0.33，网络宽度为 0.25 的 YOLOv5s 模型。

3.2 YOLOv5 输入

3.2.1 自适应锚框

YOLOv5 的锚框（anchor box）可以手动填写，也可以自动生成。如果手动输入锚框信息，YOLOv5 会根据训练数据集的标注信息计算标注信息对于手动输入锚框的最佳查全率（Recall）,如果 best possible recall< 0.98则重新计算符合数据集标注的锚框。

YOLOv5 通过 k-means 聚类算法迭代生成最佳查全率大于 0.98 并且锚框宽高比小于要求的最大阈值的锚框。

3.2.2 数据增强与数据处理

YOLOv5 使用了多种数据增强和数据处理手段，具体如下：

（1） 图片自适应填充，在数据集的图片中填充灰边，使其可以输入到目标检测网

络之中。效果如下图 3-1（a）。

（2）Mosic：Mosic 数据增强会除了加载的一张图片之外再在数据库中随机选择三张图片，并将这四张图片以随机方式拼接并剪切拼接好的图片以适应网络输入尺寸。拼接完成后获取所有目标标签并剔除被剪切的部分，这可以让模型能在更小范围内识别图像， 并减少网络对电脑 GPU 的显存占用，同时增加了模型的鲁棒性。图片进行 mosic 变换并获取标签后效果如下图 3-1（b）。

（a） （b）

图 3-1（a）图片自适应填充效果图；（b）Mosic 数据增强效果图

（3）随机仿射变换（RandomPerspective）：通过旋转、平移、剪切、放缩、镜像等方式令图片目标的形变达到视角变换的效果。效果如下图 3-2。

（a） （b）

图 3-2 随即仿射变换（a）原图；（b）仿射变换效果图

（4）图片直方图均衡化：通过改变图片的色彩，饱和度和亮度进行均衡化，提高图片的局部对比度，效果如图 3-3。

（a） （b）

图 3-3 直方图均衡化（a）原图；（b）均衡化效果图

（5）MixUp：将随机两张图片进行重叠，让网络可以在同一张输入的图片上提取到原本不同图片的特征。效果如图 3-4 所示：

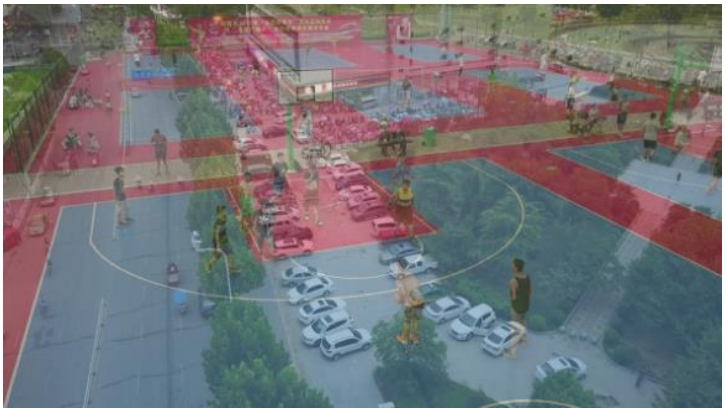


图 3-4 MixUp 效果图

3.3 主干神经网络

YOLOv5 在网络结构上主要被分为三部分，backbone，neck，head。其中 backbone 就是 YOLOv5 的主干神经网络。主干神经网络主要是将输入图片的特征进行提取并输出。

3.3.1 Focus 模块

YOLOv5 骨干网络中的第一个模块是 Focus 模块，该模块将高分辨率的输入图片切分成四份，将单张高分辨率图像拆分成四张低分辨率图像。原理如图 3-5。首先将图片进行间隔采样，然后将间隔采样的采样结果进行拼接。相当于进行了一次 1/2 的下采样并在通道维度进行了信息扩充。这在本质上相当于一个简单的扩张卷积操作，保证了图像信息没有丢失，但是增加了网络学习的速度。在 YOLOv5s 中，$640 \times 640 \times 3$尺寸的输入图片经过 Focus 模块后输出为$320 \times 320 \times 12$的尺寸。

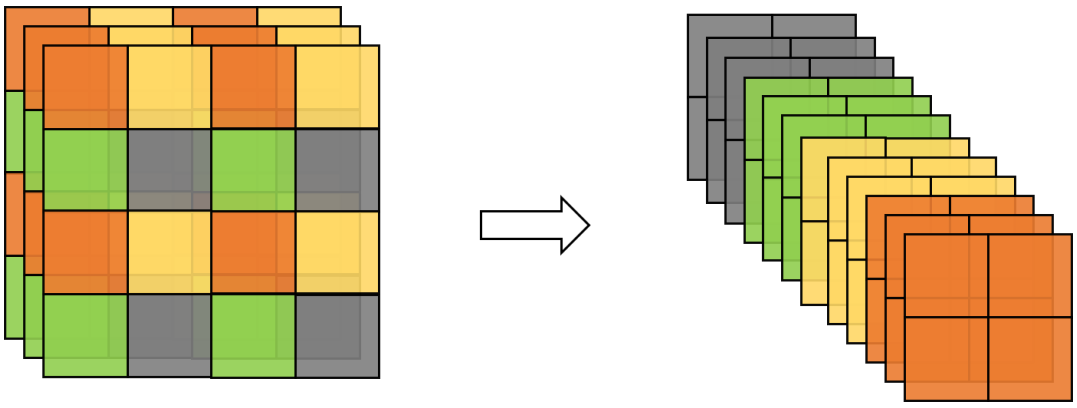


图 3-5 Focus 模块原理示意图

3.3.2 YOLOv5 网络模型基本原理

YOLOv5 模型基本结构如下图 3-6：

YOLOv5 首先通过卷积进行下采样，这里进行下采样主要是为了进行特征提取，不同采样深度的图片所提取出来的特征会用于网络训练。下采样深度到达图片原本尺寸的 1/32 时停止下采样，然后再进行上采样，同时与之前下采样的小尺寸图片进行 Concat(图

片合并)，图片合并是为了进行特征融合来提高模型的准确率和鲁棒性。最后三个检测头来进行最终的目标检测和结果输出，多个检测头体现了多尺度检测的思路。

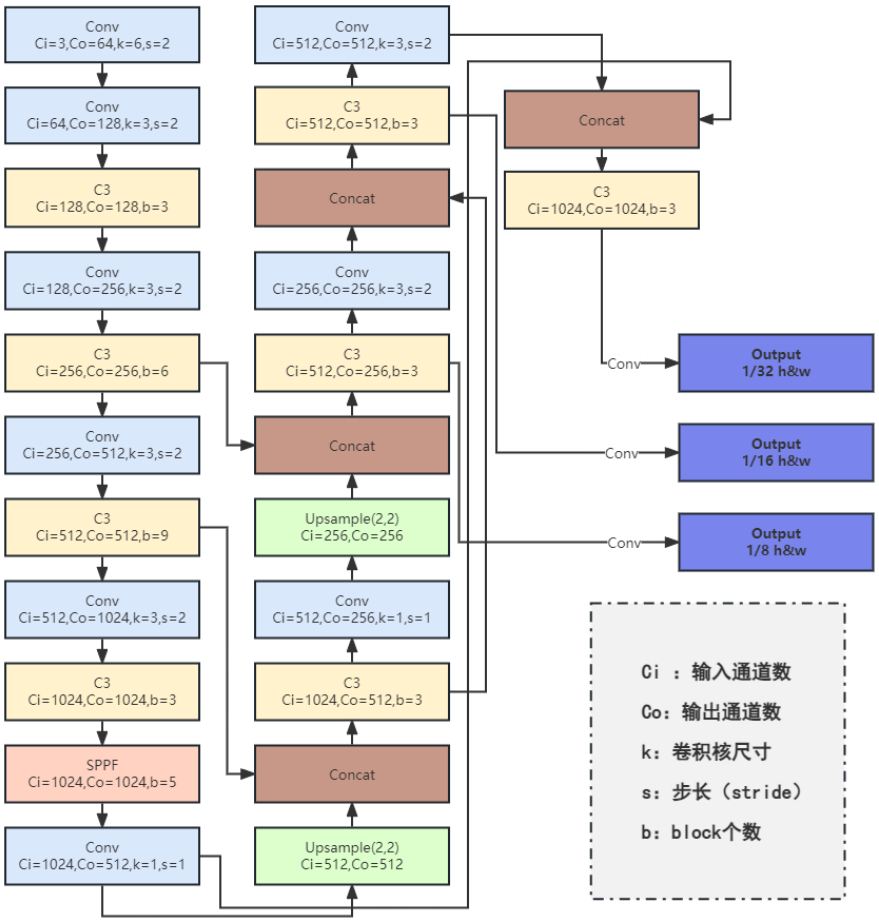


图 3-6 YOLOv5 模型结构图

为了提高 YOLOv5 的卷积效率，适应当前电脑的并行处理运算特点，YOLOv5 的卷积层使用了分组卷积令其可以在多核 GPU 进行分布式卷积运算，减小了单次卷积产生的参数量，原理如下图 3-7：

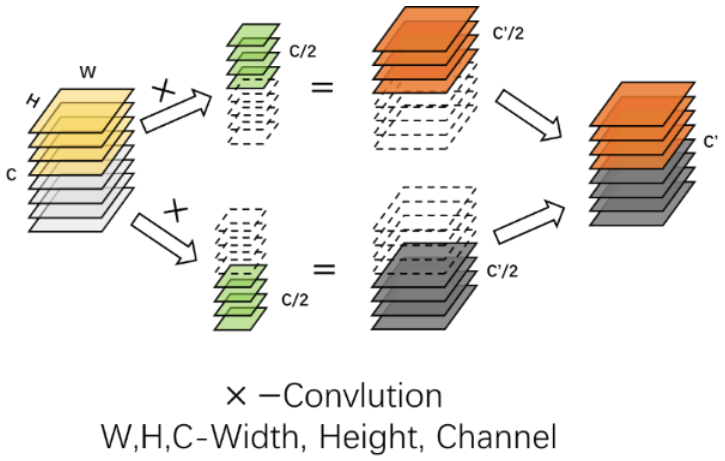


图 3-7 分组卷积原理示意图

可见原本的卷积参数量为：

$$w \times h \times C \times C' \tag{3-1}$$

分组后的卷积参数量为：

$$\mathrm{w} \times \mathrm{h} \times \frac{\mathrm{C}}{2} \times \frac{C'}{2} \times 2 = \frac{1}{2} \times \mathrm{w} \times \mathrm{h} \times C \times C' \tag{3-2}$$

可见减少参数量到了原来的$\frac{1}{2}$，当分组数为N时，减小的倍数同样为N。

在 YOLOv5 中，卷积层采用的激活函数为 SigmoidRelu[31]（SiLu）激活函数，公式如下：

$$\mathrm{SiLU} = \frac{x}{e^{-x}+1} \tag{3-3}$$

SiLU 在$x < 0$时有负值，可以防止梯度弥散消失。同时 SiLU 通过实验得知[32] 具有一定的自稳定性和隐形正则效果。

YOLOv5 中的 CSP 模块以及 C3 模块如图 3-8 所示。CSP 模块借鉴了 CSPNet，目的是为了解决网络训练时梯度重复计算的问题，以此来减少模型推断时所消耗的时间。YOLOv5 中的 CSP 模块在主干网络中使用了残差网络块来代替普通卷积，而在瓶颈层中，YOLOv5 的 CSP 模块使用了普通的卷积块来进行运算。这样避免了模型在训练过程中梯度值被重复计算。并且提高了推理速度，在反向传播的过程中因为搭载了残差块，也可以加强网络的特征提取能力，同时减少梯度消失的情况出现。YOLOv5 中卷积层之后是改良过的 C3 层，相比 bottleneckCSP[33] 层，将其中的部分 concat 操作替换成了 add 操作。两者的主要差别在于减少了一个卷积层，bottleneckCSP 层有四次卷积，C3 只有三次卷积。减少一个卷积层之后 C3 的前向传播速度和反向传播速度都有一定程度的提高，并且精度保持不变。

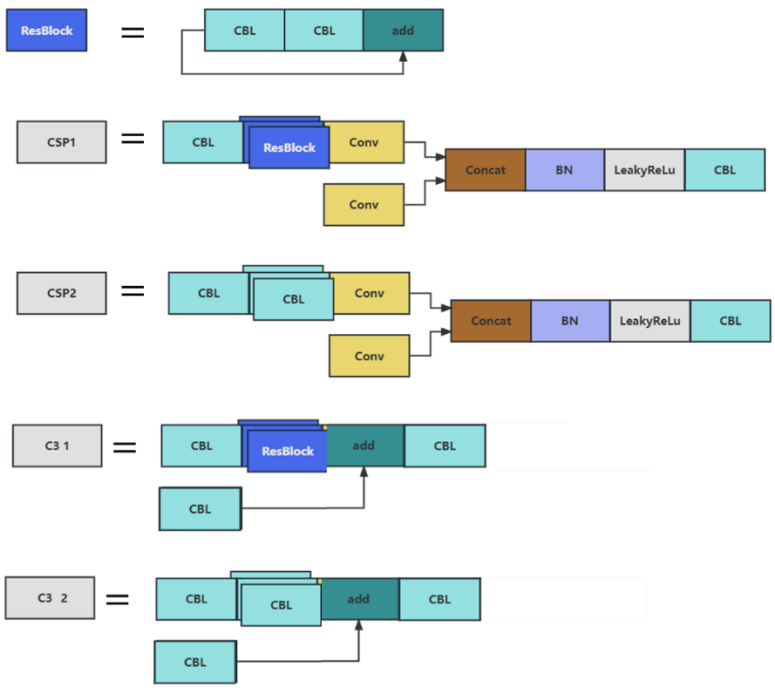


图 3-8 CSP、C3 结构对比示意图示意图

3.4 Neck

SPP[34] 的大致原理如图 3-9，具体为将所有通道的特征图都添加上三个尺寸的网格，然后根据这个单个网格尺寸进行最大池化，池化后再叠加可以让任意尺寸的特征图变为同一尺寸。SPPF，为 SPP 的改进形式，将并行的三个池化层改为串行，增加了模型的性能并且模型精度不变。

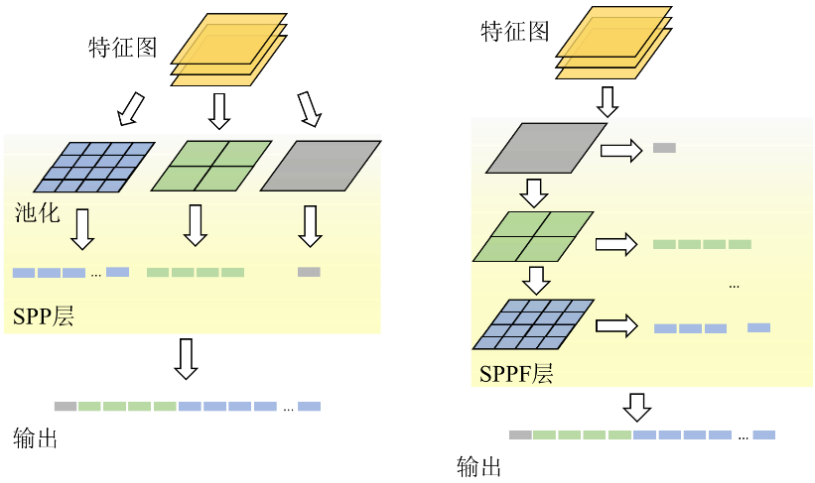


图 3-9 SPP、SPPF 金字塔池化示意图

3.5 Head

3.5.1 边框预测方法

边框预测公式如下：

$$b_x = (2\sigma(t_x) - 0.5 + c_x) * \text{stride}) \quad (3\text{-}4)$$

$$b_y = (2\sigma(t_y) - 0.5 + c_y) * \text{stride}) \quad (3\text{-}5)$$

$$b_w = \left(2\sigma(\mathrm{t_w})\right)^2 * anchor - grid \quad (3\text{-}6)$$

$$b_h = \left(2\sigma(\mathrm{t_h})\right)^2 * anchor - grid \quad (3\text{-}7)$$

t_x，t_y，t_w，t_h为模型预测输出，b_x，b_y，b_w，b_h是最终预测的目标边框的中心点位置和边界框宽高。而 YOLOv3 和 YOLOv5 中的边框预测公式去掉网格坐标后对比如下：其中（8）为 YOLOv3 的公式，（3-9）为 YOLOv5 的公式

$$x_{offset} = \sigma(t_x) \quad (3\text{-}8)$$

$$x_{offset} = 2\sigma(t_x) - 0.5 \quad (3\text{-}9)$$

发现两者偏移量值域空间有变，v3 在经过 Sigmoid 函数后值域为$(0,1)$，v5 由于正采样逻辑的变化，值域为$(-0.5,1.5)$，帮助正采样扩充。

3.5.2 损失函数

YOLOv1 使用了 IOU（重叠度）来计算损失函数，YOLOv3 使用了 L2 计算损失，YOLOv5 采用了 CIOU 来计算损失[43] ，本文采用了 SIoU 损失函数公式。其中 SIoU 是由 GIoU[35] 迭代而来，公式如下：

$$IoU = \frac{|A \cap B|}{|A \cup B|} \quad (3\text{-}10)$$

$$GIoU = IoU - \frac{|C(A \cup B)|}{|C|} \quad (3\text{-}11)$$

$$DIoU = IoU - \frac{\rho^2(b, b^{gt})}{c^2} \quad (3\text{-}12)$$

$$CIoU = IoU - \frac{\rho^2(b, b^{gt})}{c^2} - \alpha v \quad (3\text{-}13)$$

$$v = \frac{4}{\pi^2}\left(arctan\frac{w^{gt}}{h^{gt}} - arctan\frac{w}{h}\right)^2 \quad (3\text{-}14)$$

$$\alpha = \frac{v}{(1 - IoU) + v} \quad (3\text{-}15)$$

$$Loss_{IoU} = 1 - IoU \quad (3\text{-}16)$$

GIoU 在 IoU 的基础上考虑多了非交叉面积比例，即图 3-10 中斜向虚线覆盖的区域。

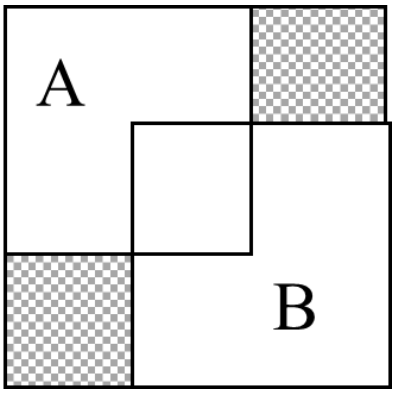


图 3-10 GIoU 计算示意图

对比 L2 损失函数，GIoU 具有尺度不变性，当边界框等比放大时，损失能依旧保持同样的大小，无需针对大小不同的边框来分别处理。对比 IoU 损失，GIoU 具有偏离趋势度量能力，当两个边框没有交集时，IoU 为 0，但是 GIoU 会在两个边框距离越远 GIoU 越接近-1。然而 GIoU 对斜向偏差敏感，GIoU 会趋势模型预测边界框分布于真实边框的上下左右方向，对斜方向预测结果施加更大损失。

为了解决 GIoU 的问题，采用了 DIoU[36] 来计算损失，DIoU 去除了非交叉面积比例的考量，而是直接在损失函数中增加了中心点距离占比惩罚项，如图 3-11 其中惩罚项分子是预测边框中心点与真实边框中心点的距离，分母是预测边框与真是边框的最小包围框的对角线长。对比 GIoU 损失，DIoU 能更好度量预测边框和真实边框的中心点距离和方向，表现如图所示，绿色真实边框，红色预测边框，当预测边框与真实边框互相包含，或者互相垂直交叉，水平交叉。GIoU 会退化成为 IoU，从而失去非交叉占比的惩罚项，而 DIoU 依旧能为模型提供更好的梯度方向：

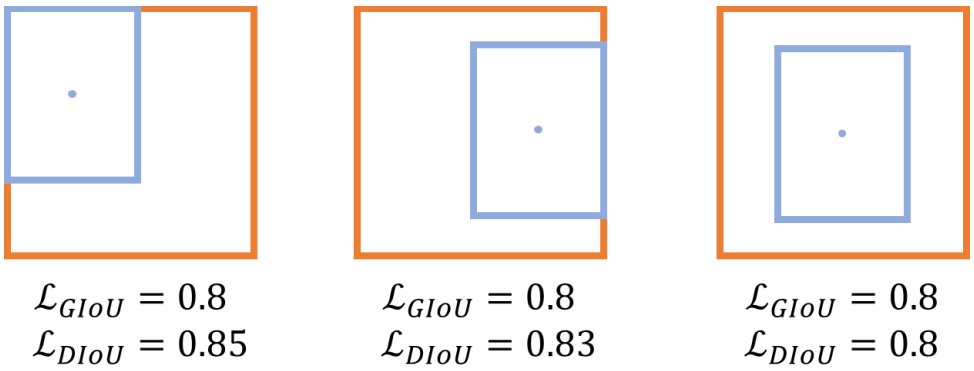


图 3-11 DIoU 计算示意图

与 GIoU 损失一样，DIoU 也具有尺度不变性，意味着当目标边框等比放大时，损失能依旧保持同样的量级，无需针对大小不同边框分别处理。DIoU 损失值域空间同样为[0,2]，当完美拟合损失 0，当距离无限远且不交叉时，损失是 2。

CIoU[37] 损失在 DIoU 的基础上，增加了宽高比惩罚项，其中υ为真实边框与预测边框的宽高比损失，α为宽高比损失系数：

对比 DIoU 损失，当预测边框和真实边框的中心点重合，CIoU 具有更好的宽高拟合效果，如果预测边框与真实边框中心点重合，DIoU 损失中的中心点距离惩罚项=0，DIoU 损失退化成 IoU 损失，但是此时 CIoU 仍有宽高比损失惩罚，能进一步调整宽高比例： CIoU 综合了 IoU 的交叉面积占比损失，DIoU 的中心点偏移损失，以及自身宽高比损失 3 种度量优点。

SIoU[19] 除了 CIoU 所考虑的区域重叠，中心点距离和宽高比等，还考虑了两个框中心点和边垂线之间的角度问题。如图 3-12 所示。

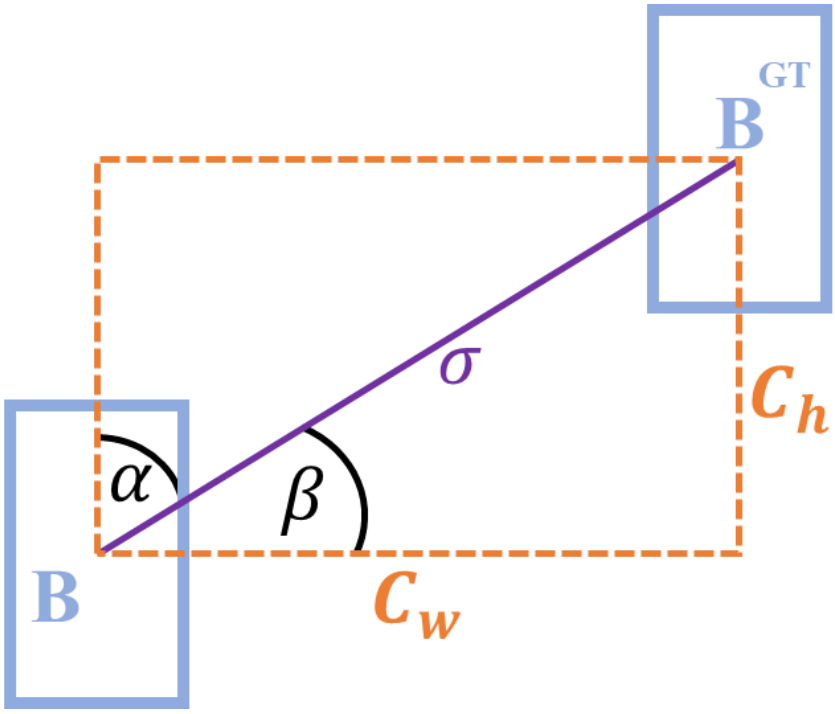


图 3-12 SIoU 计算示意图

SIoU 由四个函数构成，分别是角度惩罚函数，距离惩罚函数，宽高比惩罚函数，重叠度惩罚函数。首先假设α和β中较小的为$\alpha <= \frac{\pi}{4}$，然后定义损失函数组件：

角度惩罚函数：

$$\Lambda = 1 - 2 * sin^2\left(arcsin(\chi) - \frac{\pi}{4}\right) \tag{3-17}$$

其中，

$$\chi = \frac{C_h}{\sigma} = sin(\alpha) \tag{3-18}$$

$$\sigma = \sqrt{\left(b_{c_x}^{gt} - b_{c_x}\right)^2 + \left(b_{c_y}^{gt} - b_{c_y}\right)^2} \tag{3-19}$$

$$C_h = max\left(b_{c_y}^{gt}, b_{c_y}\right) - min\left(b_{c_y}^{gt}, b_{c_y}\right) \tag{3-20}$$

距离惩罚函数：

$$\Delta = \Sigma_{t=x,y}(1 - e^{-\gamma\rho_t}) \tag{3-21}$$

其中，当α角度越接近$\frac{\pi}{4}$，则距离惩罚函数的权重越大。

$$\rho_x = \left(\frac{b_{c_x}^{gt} - b_{c_x}}{c_w}\right)^2, \rho_y = \left(\frac{b_{c_y}^{gt} - b_{c_y}}{c_h}\right)^2, \gamma = 2 - \Lambda \tag{3-22}$$

形状惩罚函数：

$$\Omega = \sum_{t=w,h}(1 - e^{-\omega_t})^\theta \tag{3-23}$$

其中，θ的值表示了形状惩罚函数的权重。

$$\omega_w = \frac{|w - w^{gt}|}{max(w, w^{gt})}, \omega_h = \frac{|h - h^{gt}|}{max(h, h^{gt})} \tag{3-24}$$

重叠度惩罚函数：

$$IoU = \frac{|B \cap B^{GT}|}{|B \cup B^{GT}|} \tag{3-25}$$

于是得出回归损失函数：

$$L_{box} = 1 - IoU + \frac{\Delta + \Omega}{2} \tag{3-26}$$

3.3 Ghost 模块

一般的特征提取方法是通过逐点卷积来对输入进行跨通道的特征提取，然后再通过深度卷积处理空间信息，然而这样的深度卷积操作会占用大量的计算机内存和每秒浮点运算次数（FLOPs）。并且通过多层卷积叠加来进行特征提取会增加大量参数，从而导致模型参数量增加。

Han[38] 等通过分析卷积神经网络所提取的特征图发现，深度卷积生成的一组特征图中有一些冗余特征图可以将普通卷积生成的特征图通过一些廉价的线性函数进行变换，通过线性变换生成这些特征图可以减少网络层数和计算量，即通过廉价线性变换来达到深度卷积的效果。这种结合了普通卷积和廉价线性变换的模块被称为 Ghost 模块。Ghost

模块如图 3-13，输入为X，通过普通卷积后得到的特征图为Y，Y经过线性变换之后成为冗余特征图Y'，最后将两组特征图同时输出。相比其他的卷积网络架构，对特征图的处理仅限于移位操作和深度卷积，但是 Ghost 模块的线性变换部分还具有多样性，有更大的对于特征图的处理空间。

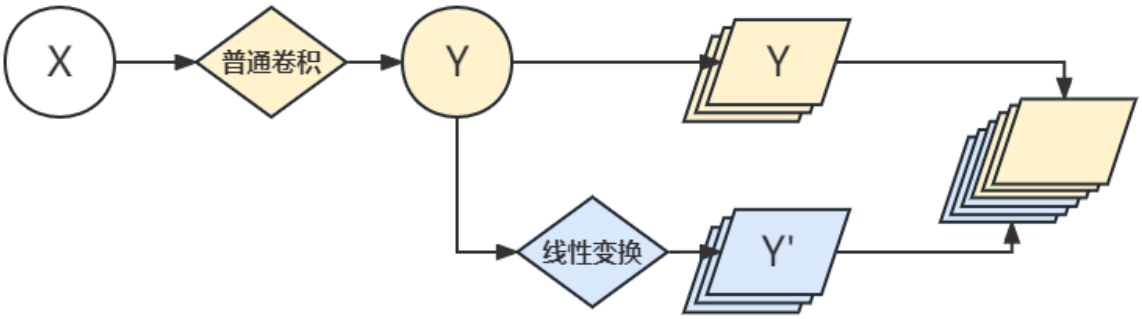


图 3-13 Ghost 模块

一张图片通过卷积层生成N张特征图如下式 3-27

$$Y_O = X * f + \mathrm{b} \tag{3-27}$$

其中$*$为卷积操作，b为偏置项，$Y_O \in \mathbb{R}^{\mathrm{h}' \times w' \times n'}$是$c$个通道下的输出特征图，$f \in \mathbb{R}^{c \times k \times k \times n}$为这一层的卷积核，$k \times k$为卷积核尺寸，$w'$和h′为输出数据的宽和高，这个过程所需要的浮点计算数（FLOP）为$n \times h' \times w' \times c \times k \times k$。在传统的深度卷积提取特征过程中，卷积核个数$n$和通道数$c$通常较大，导致需要的浮点计算数（FLOP）大幅度增长。

如果只需要原输入的内在特征，那么只需要通过更少的卷积次数来生成，如式 3-28。

$$Y' = X * f' \tag{3-28}$$

其中$f' \in \mathbb{R}^{c \times k \times k \times m}$为该卷积所使用的卷积核，$m$为得到原始输入内在特征所需要的卷积次数。因为$m$远小于深度卷积的卷积核个数$n$，所以偏置项b可以忽略。

为了提取出足够的冗余特征来增加模型的精度，Ghost 模块将m维特征图通过廉价线性变换得到$n - m$维冗余特征图，从而保证 Ghost 模块和原本的深度卷积在空间尺度上相同。线性变换见下式 3-29。

$$\mathrm{y_{ij}} = \Phi_{\mathrm{i,j}}(y'_i), \ \forall i = 1, \ldots, m, \quad j = 1, \ldots, s, \tag{3-29}$$

其中y'_i为Y'中第i个卷积生成的特征图，$\Phi_{\mathrm{i,j}}$是将第i个特征图通过线性变换得到y'_i的第j个冗余特征图。当$j = s$时，则保留y'_i特征图本身。因此对于任意的i，特征图y'_i将通过线性变换得到特征图集合：$\{\mathrm{y_{ij}}\}_{j=1}^{s}$，包括$s - 1$个冗余特征图和1个原本的特征图。而对于$Y'$，相当于增加到了$n = m \times s$个包括冗余特征图在内的特征图。原始输入在通过 Ghost 模块后得到的与深层卷积输出Y_O同样空间尺度的$\{Y\} = \{\mathrm{y_{11}}, \mathrm{y_{12}}, \mathrm{y_{13}}, \ldots, \mathrm{y_{1s}}, \ldots, \mathrm{y_{is}}\}$。

既然 Ghost 模块和深度卷积网络的输入输出在空间尺度上相同，那么可以很轻易地计算出两者在运算过程中所需要的浮点运算次数。所以任意选择普通卷积过程中的一个

特征图，这一特征图将通过$m \times (s-1) = (n/s)(s-1)$个线性变换关系得到其余的冗余特征图。Ghost 模块相比传统的卷积理论的速度提升比例r_s（Ratio of Speed）见式 3-30。

$$r_s = \frac{n \cdot h' \cdot w' \cdot c \cdot k \cdot k}{\frac{n}{s} \cdot h' \cdot w' \cdot c \cdot k \cdot k + (s-1) \cdot \frac{n}{s} \cdot h' \cdot w' \cdot d \cdot d} = \frac{c \cdot \mathrm{k} \cdot \mathrm{k}}{\frac{1}{s} \mathrm{c} \cdot \mathrm{k} \cdot \mathrm{k} + \frac{s-1}{s} \mathrm{d} \cdot \mathrm{d}} \approx \frac{s \cdot \mathrm{c}}{s + c - 1} \approx s \tag{3-30}$$

其中$d \times d$为 Ghost 模块的线性核尺寸，与$k \times k$在大小上相同，因此可以在计算的分式中抵消。并且由于$s \ll c$，所以最终得到理论速度提升比例$r_s = s$。

而理论压缩比r_c（Ratio of Compression）见式 3-31：

$$r_c = \frac{n \cdot \mathrm{c} \cdot \mathrm{k} \cdot \mathrm{k}}{\frac{n}{s} \mathrm{c} \cdot \mathrm{k} \cdot \mathrm{k} + (\mathrm{s}-1) \cdot \frac{n}{s} \mathrm{d} \cdot \mathrm{d}} \approx \frac{s \cdot \mathrm{c}}{s + c - 1} \approx s \tag{3-31}$$

可见使用 Ghost 模块代替普通深度卷积网络时，理论压缩比r_c和理论加速比r_s相等。

Ghost 模块的瓶颈层（Bottleneck Layer）结构如图 3-14 所示

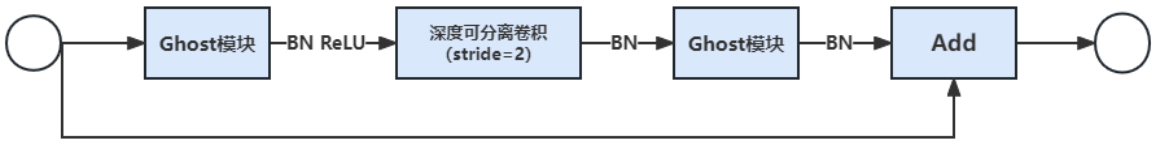


图 3-14 Ghost 瓶颈层结构图

Ghost 模块的瓶颈层结构上类似 ResNet[14] 的残差块，使用了一个 shortcut，两个 Ghost 模块之间有一个步长为 2 的深度可分离卷积[39] 层，这是为了减少模型的参数计算量。并且模块和卷积层之间都有一个 BN（Batch Normalization 批归一化）层进行放缩和偏移来矫正数据。

3.4 知识蒸馏

深层卷积神经网络训练生成的大模型是基于深度学习的目标检测的基石，理论上网络规模越大，层数越深，则模型的准确率就越高，这是因为在训练阶段，目标识别任务从规模大，冗余性高的数据集中提取的特征精确度会提升。然而这会导致模型参数过多，需要的计算量大大增加，在实际场景中进行应用时可能会因为计算资源的匮乏，对于识别延迟的高要求导致这种方法训练出的模型失去可用性。因此 Hinton[40] 等提出了知识蒸馏（Knowledge Distillation，KD）方法来对模型进行压缩。知识蒸馏简单来说是一种将大型网络的知识迁移到轻量型神经网络中的一种方法，并且这种方法可以提升几乎所有机器学习的轻量化算法的性能。

然而如果直接使用大模型的训练结果会让小模型的知识泛化能力变低。下图为一个模型的部分训练结果经过不同温度的蒸馏之后的结果，可以看到蒸馏温度提高后，数据的精准度降低，但是数据的相对大小和数据之间的关联性更强了，如果直接使用精确度极高的大模型的训练结果和直接使用训练集进行训练的结果会相差无几，这是因为在精确度的尺度上大模型远高于小模型，对于小模型来说，大模型迁移来的知识和训练集提供的知识相差无几。因此，需要将大模型的训练数据进行“软化”。软化的知识信息量比原本的知识信息量更大，具有更多的冗余信息。所以可以让小模型训练出更好的结果。

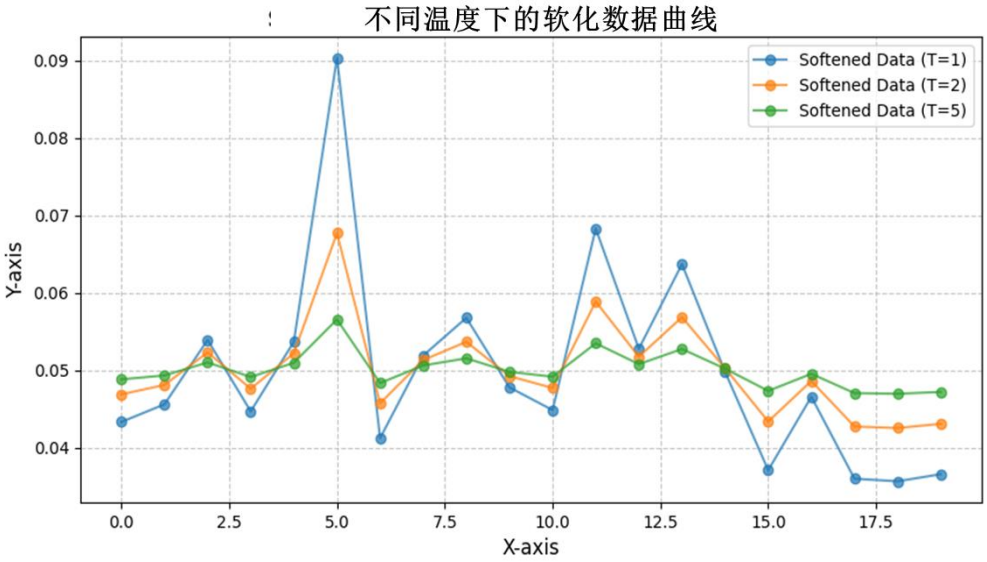


图 3-15 不同温度下的软化曲线，T=1 时为 Softmax

一般来说，使用 Softmax 来对数据进行软化是在多分类问题中最常见的解决方式。其公式如下：

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{c=1}^{C} e^{z_C}} \tag{3-32}$$

其中z_i为第i个分类的输出值，C（Class）为类别个数。又因为在平均值公式中加入了指数函数e^{z_i}，这样令不同类型之间的差别放大了，但是$Softmax$并不能解决数值上远小于 1，但是不同数值之间的数量级差距很大的数字之间的区分问题，比如10^{-6}和10^{-9}，由此，在$Softmax$基础上加入了一个新参数 T(Temperature)。得到公式：

$$q_i = \frac{e^{z_i/T}}{\sum_{c=1}^{C} e^{z_C/T}} \tag{3-33}$$

该公式与式 3-32 几乎相同，仅在所有指数项下除以了 T 参数。并将改进后的结果称为软目标。加入 T 是为了调整输出q_i的差距，主要是为了将原本经过$Softmax$后的极小值抬升，让小型网络可以利用到。 如下图为具体蒸馏过程

首先使用大型网络进行训练，得到 hard target（即没有进行软化的大型网络训练结果），和 soft target（进行软化后的训练结果）。然后分别与小型网络的两个输出计算交叉熵，其中 λ 是两个输出的比重。需要注意的是 soft target 因为在指数项中加入了 T 参数，因此与 hard target 的梯度规模不同，为了在梯度下降时将两者进行统一，需要在 soft target 的输出结果中再乘以$\frac{1}{T^2}$。

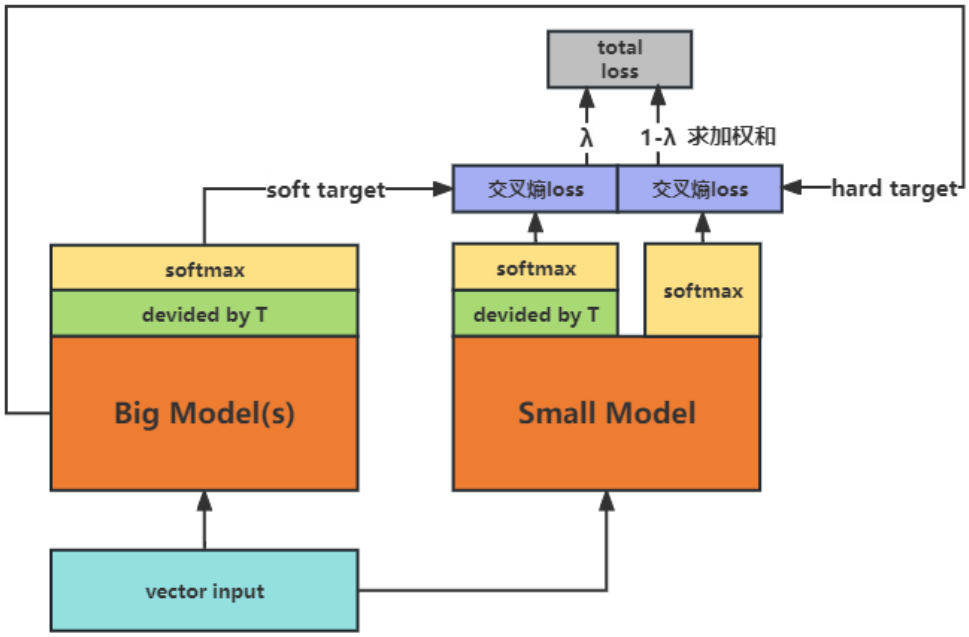


图 3-16 知识蒸馏原理示意图

3.5 注意力机制

注意力（attention）机制最开始由 Minh[41] 在 2014 年提出。在深度学习领域，注意力机制在功能上与人实际生活中的注意力焦点机制非常相似，注意力机制会专注于视野中更关键的信息，并忽略其他信息，因此注意力机制本质上是给视野中的不同区域增加权重，并更具权重来判断需要更加关注哪些区域，这让加载了注意力机制的模型可以捕捉到注意力机制所推荐的信息熵最高的特征，以此提高模型的学习能力和一定程度的可解释性。

假设一个键值对(K,V)，其中 K 为一张图片的矩阵形式，V 是一个表达了图片的注意力聚焦区域的矩阵。 现在根据(K,V)来执行一个 Query（查询），首先需要计算 Query 和 Key 的相似度公式$F(Q,K)$，一般公式如下(a)(b)(c)：

$$Similarity(Query, Key_i) = Query \cdot Key_i \quad (3\text{-}34)\ (a)$$

$$Similarity(Query, Key_i) = \frac{Query \cdot Key_i}{||Query|| \cdot ||Key_i||} \quad (3\text{-}34)\ (b)$$

$$Similarity(Query, Key_i) = MLP(Query, Key_i) \quad (3\text{-}34)\ (c)$$

然后将求得的Sim_i使用$Softmax$方法进行归一化，并放大不同相似度之间的差异。式如下：

$$\alpha_i = Softmax(Sim_i) = \frac{e^{Sim_i}}{\sum_{c=1}^{C} e^{Sim_c}} \quad (3\text{-}35)$$

进行了归一化后将得到的值乘以权值V再求和即可得到注意力评估值（Attention Score）。见式 3-36：

$$Attention(Query, Key_i) = \sum_{c=1}^{C} \alpha_i \cdot Value \quad (3\text{-}36)$$

这样就可以通过具体的注意力评估值来对键值对(K,V)进行训练。

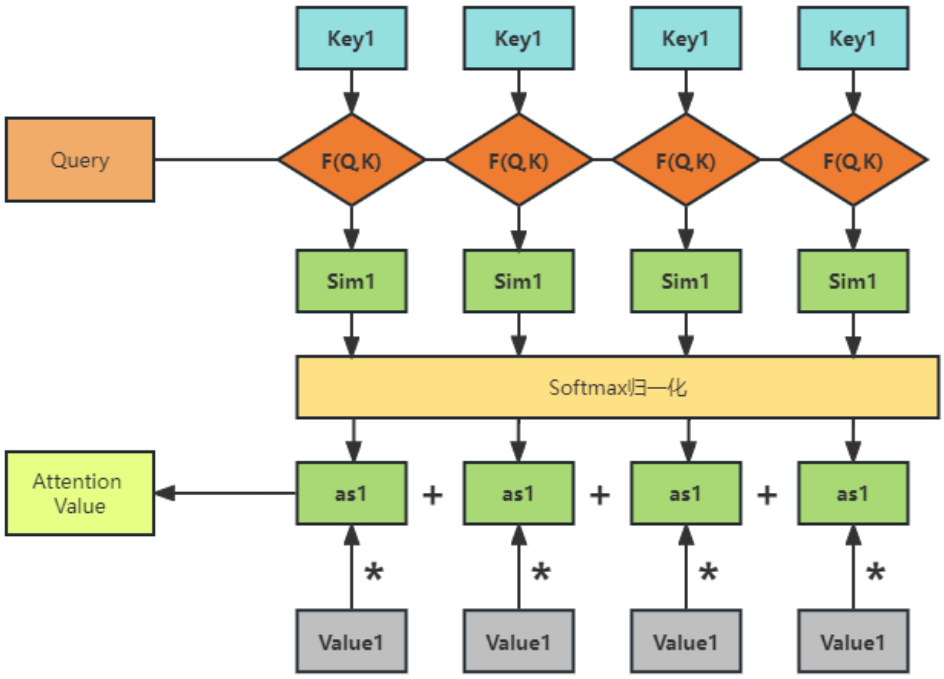


图 3-17 注意力机制原理示意图

Google 团队[42] 在 2017 年所发表的 Transformer 进一步完善了注意力机制，在目标检测模型中仅仅使用了简化之后的 Transformer 编码器，其结构如图 3-18，输入的内容经过嵌入（embedding）后通过叠加位置编码矩阵（positional encoding）偏移到下一个矩阵，即假设输入为$\mathcal{A}$，$\mathcal{A}$的尺寸为$[batch\ size, sequence\ length, embedding\ dimension]$：

$$\mathcal{A}_{embedding} = \mathcal{A}_{embedding} + \mathcal{A}_{pos} \tag{3-37}$$

然后将$\mathcal{A}_{embedding}$通过设计好的线性函数映射产生上文中的 Query，Key 和 Value，因此这三者与编码矩阵的维度式相一致的。然后经过探测头数量为$embedding\ dimension$的注意力机制，这里为了数据方便处理，在相似度计算上做了一些改变：

$$Attention(Query, Key_i) = \sum_{c=1}^{C} Softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \cdot V \tag{3-38}$$

其中$1/\sqrt{d_k}$将注意力矩阵变为正态分布。最后经过一个前馈网络层。前馈网络层和多头注意力层可以多个进行串联。

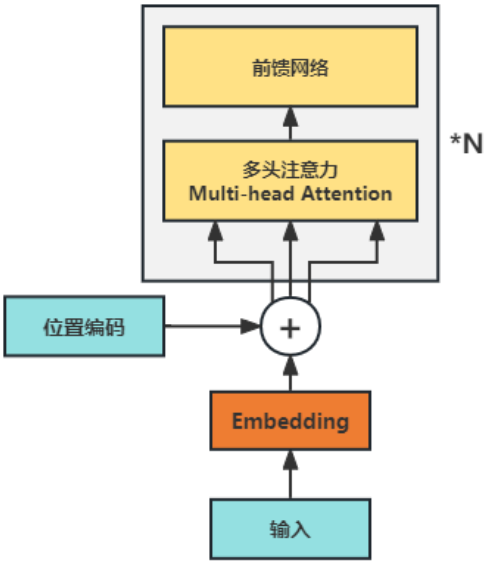


图 3-18 Transformer 模块中的编码器原理示意图

3.6 针对小目标的额外特征提取层

YOLOv5s 的基准版本中，在主干网络共进行 5 次下采样操作，其中第一次下采样使用了扩张卷积的方式，采样深度为 1/2，第一次采样是为了达成上文中所提到的 Focus 模块的效果。5 次采样后，采样深度达到 1/32。这一部分与本文所使用的模型相一致。

但是本文的模型在检测头部分额外增加了一层上采样层和一个检测头如图 3-19 中的虚线部分。在原本的采样深度下，检测头所检测的特征图尺度分别为20×20，40×40，80×80。但是在 VisDrone2019 数据集中，航拍图像的小目标尺寸甚至会小于3×3像素，这会导致80×80的特征图尺度仍然无法让模型成功检测到数据集中的小目标，因此在添加了额外的采样层和检测头之后，第四个检测头所检测的特征图尺寸为160×160，可以有效地提升模型对于小目标的检测精度，虽然这会导致模型的计算量有一定程度的上升，但是根据实验结果对比来看，增加特征提取层所带来的检测精度的提升效果时值得的。

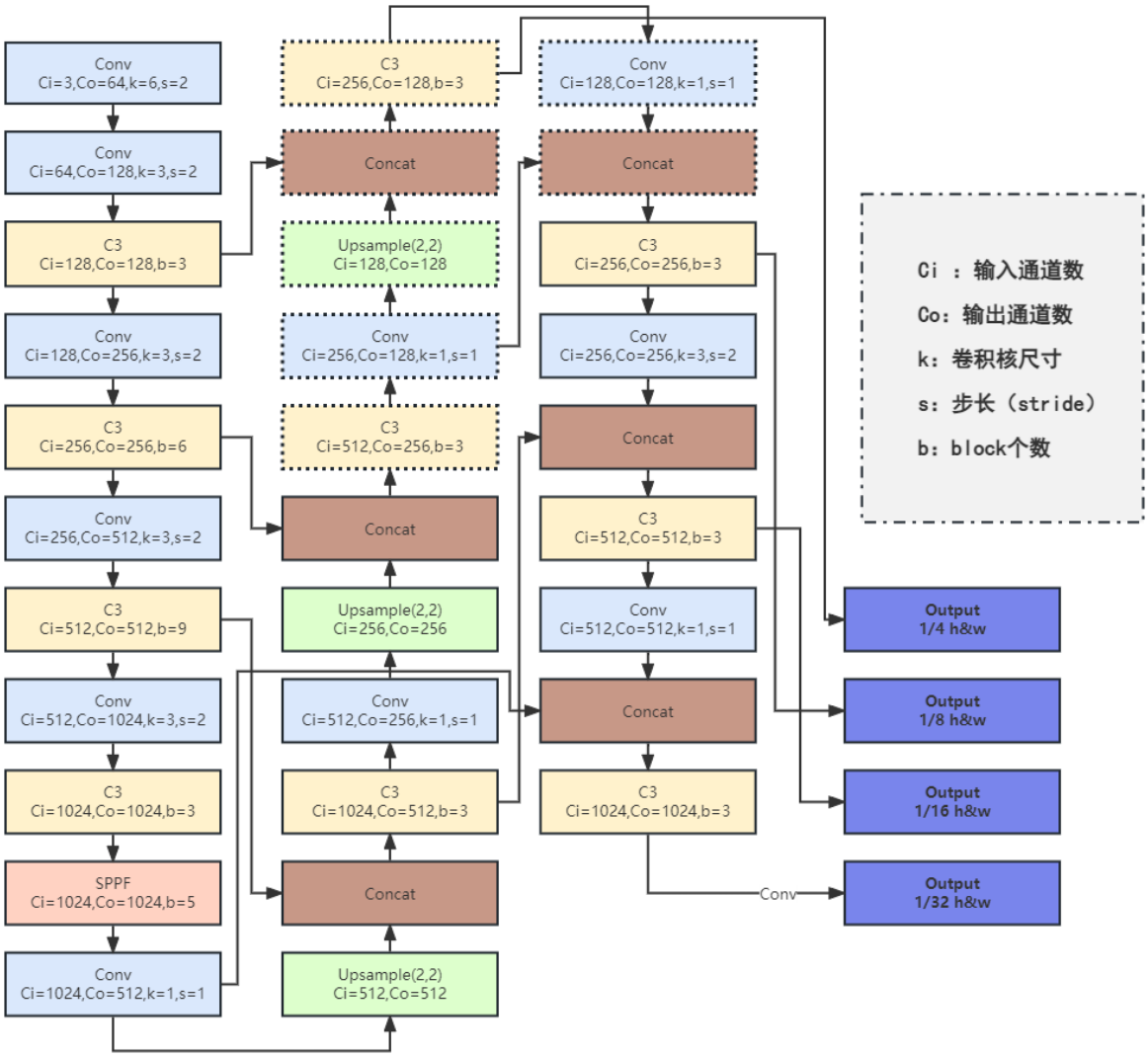


图 3-19 模添加了额外特征提取层的 YOLOv51 模型框架示意图

3.7 模型整体框架结构

在主干网络层中，为了提取图片的特征，进行五次下采样，第一层下采样使用普通卷积充分提取原图的特征，后三次下采样使用 Ghost 卷积，第五次下采样使用普通卷积并在之后的 CSP 瓶颈层中加入了 Transformer 编码器模块，为模型提供特征图中的关键的注意力区域。瓶颈层使用了 SPPF，检测头部分共有四个检测头。

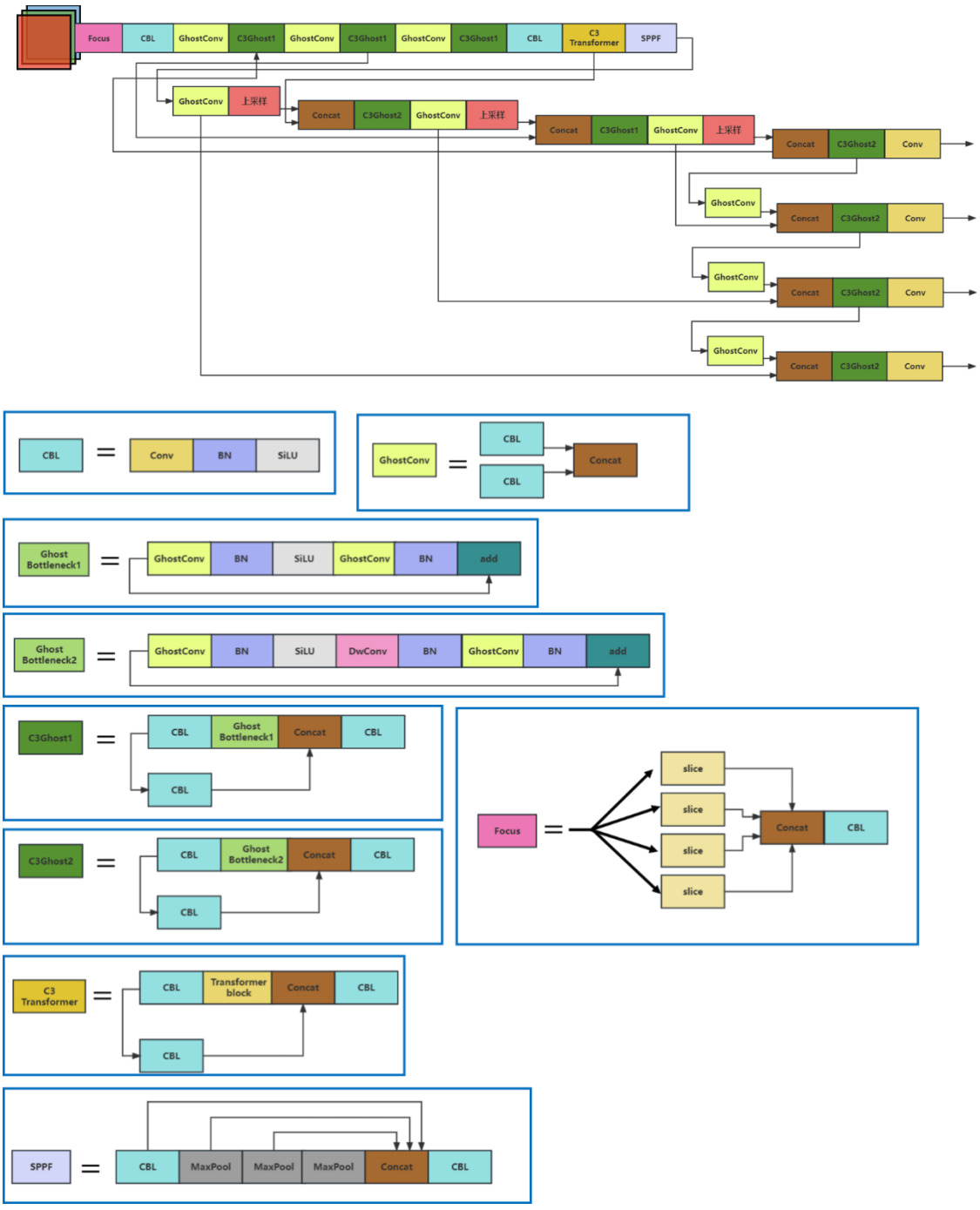


图 3-20 模型整体框架示意图

3.7 本章小结

在这一章节中按照 YOLOv5 的运行流程为顺序详细地介绍了整个 YOLOv5 算法的运行模式，本文从数据预处理开始，介绍了算法的数据增强策略，然后介绍了 YOLOv5 基准模型的网络结构，提出了四个改进方法来使模型轻量化并提高模型的准确率，在最后则展示了改进后的模型的框架结构。

本章中所提出的四个改进措施如下：（1）将基准模型结构中原本的普通卷积换为计算量更少的 Ghost 卷积。（2）在模型的瓶颈层中加入 Transformer 模块来引入注意力机

制，提高网络对于目标区域的关注度，滤除背景干扰。（3）使用知识蒸馏的方式将YOLOv5l 的知识迁移到改进的模型上，令其具有更高的准确度和泛化能力。（4）再加入额外的一组上采样层和检测层，让模型对小目标更加灵敏，从而提高对小目标检测的精准度。

4 实验结果分析

4.1 实验环境与模型超参数

收敛性分析

本次实验所使用的实验环境和模型的参数设置如表所示。

表 4-1 实验环境

环境配置	名称	信息
硬件配置	GPU	NVIDIA GeForce GTX 4090
	CPU	Intel Core i7-12700KF
	内存	64G
	显存	24G
软件环境	操作系统	Windows10
	Python	3.9.7
	Pytorch	2.0.0
	CUDA	11.8
	cuDNN	8.7.0.0

表 4-2 超参数设置

名称	数值
训练图片分辨率（image_size）	640×640×3
迭代运行次数 epochs	300
批大小（batch_size）	32
优化器（optimizer）	SGD
初始学习率（lr0）	0.01
周期学习率（lrf）	0.01
学习率动量（momentum）	0.937
权重衰减系数（weight_decay）	0.0005

4.3 数据集和评价指标

本文因为以无人机航拍图像作为背景，因此选用 VisDrone2019 数据集来进行训练，该训练集相比常用的 COCO 和 VOC 数据集，图片上的目标密度大，数量多，尺寸小，并且由于是无人机航拍图像的缘故，图片内的物体更容易失真。

VisDrone 数据集总共分为 10 类（不包括背景），分别是 pedestrian，prople，bicycle，car，van，truck，tricycle，awning-tricycle，bus，motor。

下图为 VisDrone 数据集所有类别的标签数量，以及通过 k-means 方法聚类得到的先验框集合。

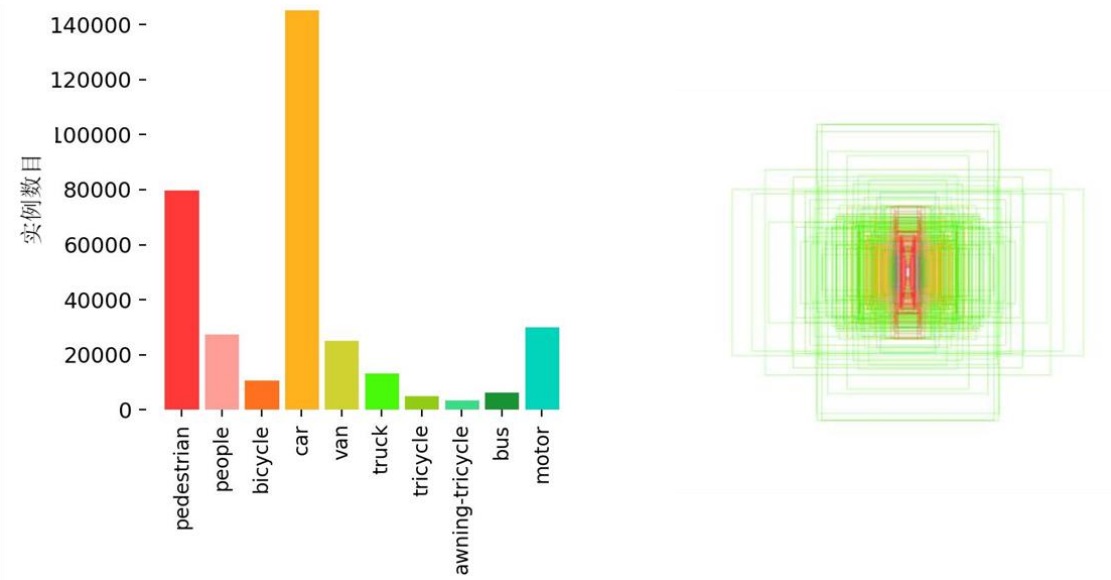


图 4-1 数据集标签数据直方图和先验框集合

目标检测评价指标如下表，如果目标检测的类别正确，并且检测框与标定的检测框重叠度高于一个阈值的检测结果为 TP（True Positives）。未被模型检测到或者检测框和标定重叠度低于阈值的结果为 FP（False Positives）。

表 4-3 被检测指标定义

预测	正相关类	负相关类（无关类）
被模型检测到	TP	FP
未被模型检测到	FN	TN

查全率 R（Recall），即正相关类输入数据被模型所正确检测到的比重。式如下：

$$Recall = \frac{TP}{TP+FN} \tag{4-1}$$

查准率 P（Precision）检测结果为正相关类在预测结果正确的样本中的占比，公式如下：

$$Precision = \frac{TP}{TP+FP} \tag{4-2}$$

F1 值，含义是正确预测的所有例子除以总数：

$$F1 = \frac{2(P\times R)}{P+R} \tag{4-3}$$

FLOPs（floating point operations 浮点运算数）：用于衡量模型复杂度和计算量。

4.4 检测算法结果分析

本文将进行了 Ghost 轻量化，加入了 Transfomer 模块，并知识蒸馏后的模型与原始的 YOLOv5s 进行对比。表 4-4 为算法对比结果。

由表可得，模型进行轻量化之后，模型大小减少到了原本的 54.2%，mAP50 也降低了 6.4%。当额外加入了 Transformer 模块，增加特征提取层，损失函数修改为 SIoU，并进行知识蒸馏后，mAP50 提升了 13.1%。这说明 Ghost 模块通过廉价的线性变换来生成

冗余特征图，有效的减少了模型的大小，而轻量化之后所施加的提升模型精度的手段起到了较为明显的效果，并且模型的参数量并没有明显增加。

表 4-4 三种模型算法对比

算法	Weights（M）	mAP50
YOLOv5s	13.7	34.2%
YOLOv5s+Ghost	7.43	29.8%
YOLOv5+Gh+Tr+KD	8.7	38.8%

YOLOv5s+Ghost+Tr+KD 模型的具体表现如下表 4-5。可以发现检测精准度较高的分类为汽车，而精准度最低的为人力三轮车和自行车，这与航拍图像的特点有关系，自行车在俯视角特征不明显，体积小，目标检测网络难以学习，而人力三轮车则是在目标小，难以识别的同时，数据集样本较少，特征和普通三轮车高度相似，容易和普通三轮车混淆。因此识别准确率较低。如图 4-2 黄色框，只有在部分高度较低的无人机视角中的被拍到侧面的自行车被模型良好的识别了出来，如图 4-2 蓝色框。而汽车则无论在侧视角还是俯视角都具有比较多的特征，因此检测精准度较高。

表 4-5 YOLOv5s+Ghost+Tr+KD 模型对于不同类别的检测性能

Class	Images	Instances	P	R	mAP50	mAP50-95
Pedestrian	548	8844	0.514	0.426	0.445	0.188
People	548	5125	0.48	0.356	0.345	0.125
Bicycle	548	1287	0.312	0.148	0.139	0.0521
Car	548	14064	0.667	0.799	0.804	0.54
Van	548	1975	0.515	0.418	0.423	0.29
Truck	548	750	0.555	0.326	0.354	0.217
Tricycle	548	1045	0.472	0.258	0.251	0.133
Bus	548	251	0.659	0.494	0.545	0.362
Motor	548	4886	0.522	0.451	0.437	0.178
Awning-tricycle	548	532	0.265	0.138	0.134	0.0836
All	548	38759	0.496	0.381	0.388	0.217

图 4-2 YOLOv5s+Ghost+Tr+KD 模型低空检测图

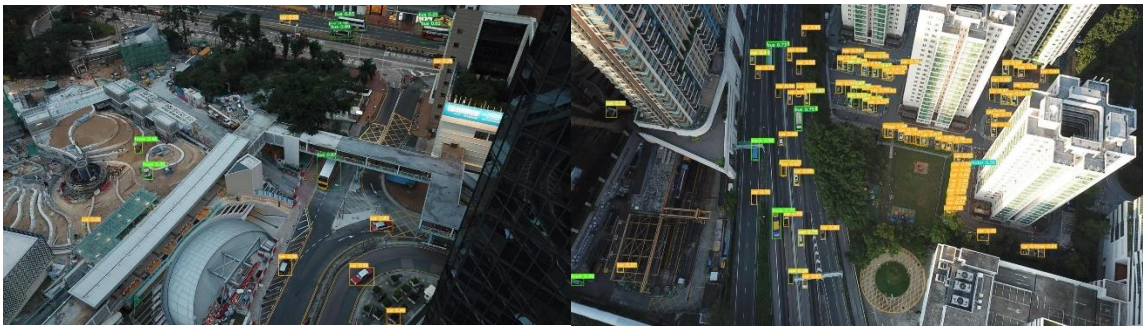
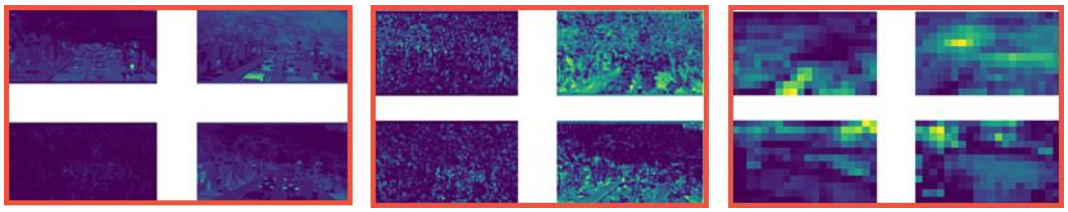


图 4-3 YOLOv5s+Ghost+Tr+KD 模型高空检测图

模型在不同层所输出的特征图如图 4-4 所示，分别为下采样 1/2，1/8，1/32 时的特征图。



（a） （b） （c）

图 4-4 在不同采样深度中网络提取的特征图（a）1/2 深度；（b）1/8 深度；（c）1/32 深度

如图 4-5 为模型在 VisDrone 数据集中验证集所生成的混淆矩阵，可以发现混淆矩阵图下方一条颜色较深的区域，这说明有很多目标被模型所漏检。且可以发现行人，一般人，和自行车一组类别，汽车，货车，卡车一组类别，一共两组类别容易混淆，第一组是因为行人和一般人具有很多相同的特征，而许多自行车上往往有人遮挡，这导致了模型学习到了错误的自行车特征。第二组主要是因为三者具有相似的特征，导致目标置信度的普遍下降。还有一点是由于 VisDrone 数据集中各种类别在数量上及其不平均，这导

致了训练结束之后类别与类别之间的识别准确率相差巨大。

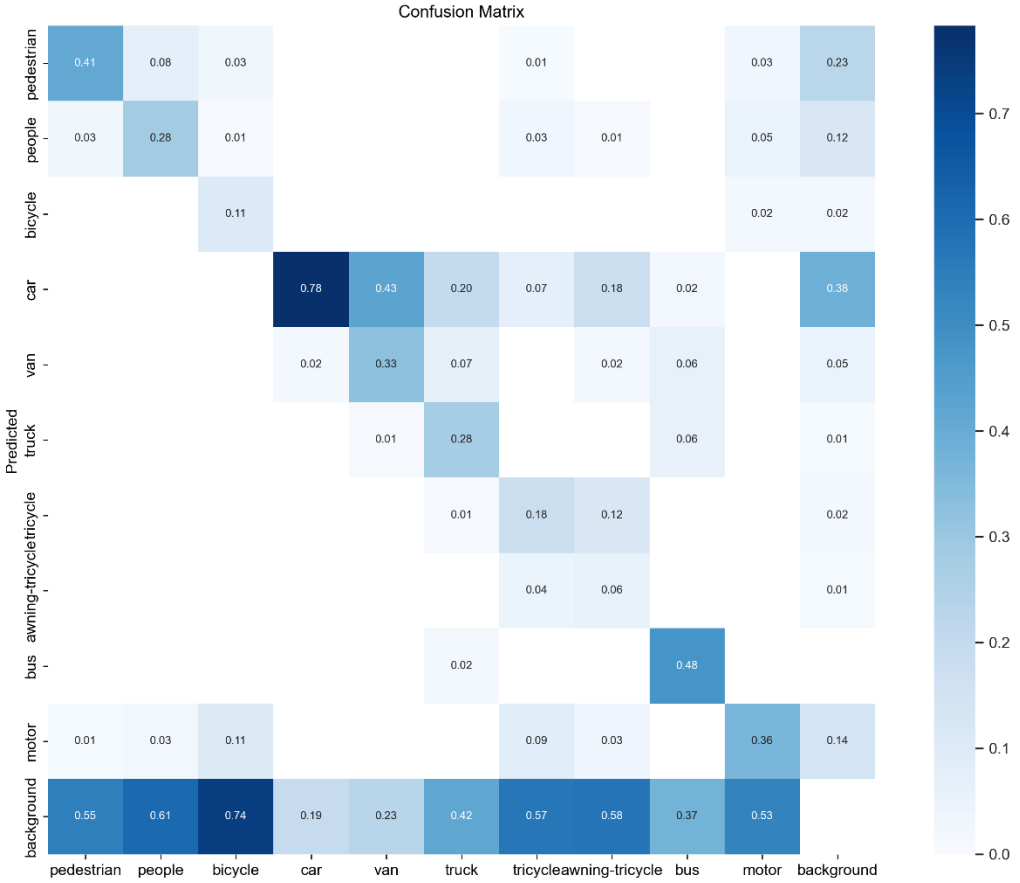


图 4-5 YOLOv5+Ghost+Tr+KD 模型混淆矩阵

4.5 消融实验

在 VisDrone 数据集上的消融实验结果如表 4-5 所示：可见 Transformer 模块可以提高模型的召回率(Recall)，而知识蒸馏（KD），则更能提高模型的精确度，但是召回率反而有所下降。通过对比表格中后三行可以看出 Transformer 模块和知识蒸馏能够一定程度上提升模型的 mAP 值，然而其效果并不明显，两者都部署后准确度提升 2%。然后在模型中再次添加一个额外的特征提取层并更换损失函数后，可以发现准确度有了大幅度提升，甚至超过了原本没有轻量化的网络模型识别准确率，同时可以发现知识蒸馏对模型准确率的提升幅度更加明显，达到了 2.4%，而注意力模块在准确率上仅提升 0.5%。

表 4-5 使用不同策略组合的模型性能对比表

方法					P(%)	R(%)	mAP50 (%)	mAP50:95 (%)
YOLOv5s	检测层更新	Ghost	Tr	KD				
√					46.0	34.9	34.3	19.0
√		√			39.7	29.9	27.9	14.7
√		√	√	√	41.6	31.3	30.0	15.5
√	√	√			45.4	36.8	36.4	20.0
√	√	√		√	47.8	38.0	38.2	21.3
√	√	√	√		46.8	37.1	36.9	20.5
√	√	√	√	√	48.6	38.1	38.8	21.7

4.6 本章小结

本章首先讲明了实验环境，训练用数据集和性能评价指标，然后将第三章所提出的对于 YOLOv5s 模型的三个改进策略进行了实验，实验结果表明通过卷积替换的方式可以有效地令模型大小减小到原来的 54.2%，而添加 Transformer 模块和进行知识蒸馏则对于轻量化之后的模型精确度有所提升，在添加了额外的特征提取层并更换损失函数之后，原本加入的准确率提升手段效果也得到了激活，准确率大幅提升，其中 mAP50 提升了 10.9%，mAP50:95 提升了 7.0%。

5 结论与展望

5.1 工作总结

为了满足将目标检测模型搭载到无人机这种嵌入式平台上，需要将原本模型体积相对较小的 YOLO 模型进一步轻量化，否则将难以拥有足够的检测效率来达到实时检测的目的，并且在模型轻量化的同时，还要保证模型的准确率尽可能高，还需要注意无人机航拍图像中含有大量的小目标，所以还要兼顾到小目标检测的问题。所以最终提出了四个方案来分别优化模型的参数量和模型的准确率。方案如下：

（1）将基准模型结构中原本的普通卷积换为计算量更少的 Ghost 卷积。Ghost 卷积相比普通卷积，使用了一些廉价的线性变换操作替代了原本使用深度卷积来提取冗余特征图的方式，从而大大减少了模型的参数量和计算量。

（2）在模型的瓶颈层中加入 Transformer 模块来引入注意力机制，Transformer 的编码器模块将目标区域的权重提高，但又没有抹去目标与背景之间的关联性，注意力机制提高了网络对于目标区域的关注度，并帮助模型滤除背景干扰。

（3）使用知识蒸馏的方式将 YOLOv5l 的知识迁移到改进的模型上，令其具有更高的准确度和泛化能力。

（4）在基准的模型中增加额外的特征提取层，并且替换损失函数，令模型准确率有了大幅度提升。

轻量化的模型在模型大小和计算量上均减少了接近一半，后续在准确度方面的优化在 VisDrone 数据集上检测结果仅得到了较大的提升，但是模型大小和计算量也有所提升，所以改进方案还需要进一步完善和优化。

5.2 研究展望

为了进一步有效提升算法性能，后续的研究首先是可以尝试通过更换模型的主干网络来进行模型的轻量化，比如 MobilNet，GhostNet，ShuffleNet 等，这一类已经经过验证的轻量化主干网络可能会让轻量化模型有更高的准确性以及更快的模型推断速度。

其次是模型居高不下的漏检率需要降低，这是由于数据集中有大量小目标，密集目标，模糊目标和被遮挡的目标所导致的，并且在增加了检测层之后仍然可能会有目标过小而导致的信息不足问题。提升算法本身对于上述情况的识别准确度或者是使用更有效地数据增强手段可能可以从根本上解决问题。

最后是数据集本身各种标签的数量分布比较极端，考虑在数据扩充时去平衡数据集中各个类别的标签数量可能会让模型具有更好的性能。

参考文献

[1] Dasiopoulou S, Mezaris V, Kompatsiaris I, et al. Knowledge-assisted semantic video object detection[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2005, 15(10): 1210-1224.

[2] Castrillón M, Déniz O, Hernández D, et al. A comparison of face and facial feature detectors based on the Viola–Jones general object detection framework[J]. Machine Vision and Applications, 2011, 22: 481-494.

[3] Lindeberg T. Scale invariant feature transform[J]. 2012.

[4] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C].2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). Ieee, 2005, 1: 886-893.

[5] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.

[6] Girshick R. Fast r-cnn[C].Proceedings of the IEEE international conference on computer vision. 2015: 1440-1448.

[7] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28.

[8] Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector[C].Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14. Springer International Publishing, 2016: 21-37.

[9] Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 779-788.

[10] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C].Proceedings of the IEEE international conference on computer vision. 2017: 2980-2988.

[11] Dai J, Qi H, Xiong Y, et al. Deformable convolutional networks[C].Proceedings of the IEEE international conference on computer vision. 2017: 764-773.

[12] 王艳,杨丰蔚,翟兴等.基于深度学习的显微图像计算机辅助诊断[J].计算机与现代化,2023,No.331(03):54-59.

[13] 龙洁花,赵春江,林森,郭文忠,文朝武,张宇.改进 Mask R-CNN 的温室环境下不同成熟度番茄果实分割方法[J].农业工程学报,2021,37(18):100-108.

[14] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[15] Simony M, Milzy S, Amendey K, et al. Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds[C].Proceedings of the European Conference on Computer Vision (ECCV) Workshops. 2018: 0-0.

[16] Zhu X, Lyu S, Wang X, et al. TPH-YOLOv5: Improved YOLOv5 based on transformer prediction head for object detection on drone-captured scenarios[C].Proceedings of the IEEE/CVF international conference on computer vision. 2021: 2778-2788

[17] Jocher G, Nishimura K, Mineeva T, et al. yolov5[J]. Code repository https://github. com/ultralytics/yolov5, 2020: 9.

[18] Du D, Zhu P, Wen L, et al. VisDrone-DET2019: The vision meets drone object detection in image challenge results[C].Proceedings of the IEEE/CVF international conference on computer vision workshops. 2019: 0-0.

[19] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.Gevorgyan Z. SIoU loss: More powerful learning for bounding box regression[J]. arXiv preprint arXiv:2205.12740, 2022.

[20] He K, Gkioxari G, Dollár P, et al. Mask r-cnn[C].Proceedings of the IEEE international conference on computer vision. 2017: 2961-2969.

[21] Gkioxari G, Malik J, Johnson J. Mesh r-cnn[C].Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019: 9785-9795.

[22] Jiang P, Ergu D, Liu F, et al. A Review of Yolo algorithm developments[J]. Procedia Computer Science, 2022, 199: 1066-1073.

[23] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 7263-7271

[24] Redmon J, Farhadi A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018

[25] Woo S, Park J, Lee J Y, et al. Cbam: Convolutional block attention module[C].Proceedings of the European conference on computer vision (ECCV). 2018: 3-19.

[26] Hu J, Shen L, Sun G. Squeeze-and-excitation networks[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 7132-7141.

[27] Zhu X, Cheng D, Zhang Z, et al. An empirical study of spatial attention mechanisms in deep networks[C].Proceedings of the IEEE/CVF international conference on computer vision. 2019: 6688-6697.

[28] Liu S, Qi L, Qin H, et al. Path aggregation network for instance segmentation[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 8759-8768.

[29] Tan M, Pang R, Le Q V. Efficientdet: Scalable and efficient object detection[C].Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020: 10781-10790.

[30] Bodla N, Singh B, Chellappa R, et al. Soft-NMS--improving object detection with one line of code[C].Proceedings of the IEEE international conference on computer vision. 2017: 5561-5569.

[31] Elfwing S, Uchibe E, Doya K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning[J]. Neural Networks, 2018, 107: 3-11

[32] Elfwing S, Uchibe E, Doya K. Expected energy-based restricted Boltzmann machine for classification[J]. Neural networks, 2015, 64: 29-38.

[33] Wang C Y, Liao H Y M, Wu Y H, et al. CSPNet: A new backbone that can enhance learning capability of CNN[C].Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020: 390-391.

[34] He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 37(9): 1904-1916.

[35] Rezatofighi H, Tsoi N, Gwak J Y, et al. Generalized intersection over union: A metric and a loss for bounding box regression[C].Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 658-666.

[36] Zheng Z, Wang P, Liu W, et al. Distance-IoU loss: Faster and better learning for bounding box regression[C].Proceedings of the AAAI conference on artificial intelligence. 2020, 34(07): 12993-13000.

[37] Zheng Z, Wang P, Ren D, et al. Enhancing geometric factors in model learning and inference for object detection and instance segmentation[J]. IEEE Transactions on Cybernetics, 2021, 52(8): 8574-8586.

[38] Han K， Wang Y， Tian Q , et al. GhostNet: More Features From Cheap Operations[C]. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2020.

[39] Chollet F. Xception: Deep learning with depthwise separable convolutions[C].Proceedings of the IEEE conference on computer vision and pattern recognition. 2017: 1251-1258.

[40] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[J]. arXiv preprint arXiv:1503.02531, 2015.k

[41] Mnih V, Heess N, Graves A. Recurrent models of visual attention[J]. Advances in neural information processing systems, 2014, 27.

[42] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[43] Yu J, Jiang Y, Wang Z, et al. Unitbox: An advanced object detection network[C].Proceedings of the 24th ACM international conference on Multimedia. 2016: 516-520.

[44] Xue Z, Lin H, Wang F. A small target forest fire detection model based on YOLOv5 improvement[J]. Forests, 2022, 13(8): 1332.

[45] 黎学飞, 童晶, 陈正鸣, 包勇, 倪佳佳. 基于改进 YOLOv5 的小目标检测

[J]. 计 算 机 系 统 应 用 ， 2022, 31(12): 242-250.http://www.c-s-a.org.cn/1003-3254/8835.html

[46] Huang H, Sun D, Wang R, et al. Ship target detection based on improved YOLO network[J]. Mathematical Problems in Engineering, 2020, 2020.

致谢

历时约摸一个月，从无从下手的实验数据和大量参考资料到人生中第一篇毕业论文，这是作为一名本科生一定要上交的最后一项作业，也是四年学习经历的沉淀和心血的浓缩。因而在此必须要对于我大学生涯中引导和帮助过我的人表达感谢。

首先是要感谢我的导师宋冰。在本科学习期间，宋老师作为专业课老师对我学习工作的认可让我重拾了学习的动力和信心，并且也耐心地指导了我的毕业设计。还有我的学长，在论文撰写期间也给予了我很大的帮助

其次需要感谢大学生活中家人对我的支持。感谢父母在我迷茫时给予我的支持、建议和让我在最后独立做决定的认可，认真倾听我的看法和事不顺心的抱怨。感谢家里长辈给我的宝贵的建议。

还要感谢我的室友，他们陪伴我度过了大学四年生活。能够遇到他们我深感荣幸。

卫宇枫

2023 年 5 月 10 日