电子科技大学（深圳）高等研究院

GPU 并行编程

课程设计报告

题目	《基于 TensorRT 的行人检测算法研究》		
专业	控制科学与工程		
学号	202122280534		
姓名	陈玉熙	签名	陈玉熙
指导教师	卢国明		
完成日期	2022.01.04		

基于 TensorRT 的行人检测算法研究

陈玉熙

摘 要 行人检测技术是利用计算机视觉技术判断图像或者视频序列中是否存在行人并给予准确定位，该技术与行人跟踪、行人识别等技术相结合，可应用于智能视频监控、人体行为分析、智能交通、车辆辅助驾驶系统和智能机器人等领域。而 TensorRT 是 NVIDIA 公司推出的高速推理引擎加速，内部使用 CUDA C 进行编程。使用 TensorRT 对深度学习网络结构进行优化和加速，可以使输出的网络具有低延迟，高吞吐率的优点。通过 TensorRT 对部署在嵌入式设备的行人检测模型进行加速，在轻量化模型的同时可以获得较好的识别效率，具有显著的优点。

关键词 计算机视觉，目标检测，行人检测，CUDA，TensorRT

Research of Pedestrian Detection Algorithm Based on TensorRT

Yuxi Chen

Abstract By using computer vision technology, pedestrian detection technology can detect whether there are pedestrians in an image or video sequence and give pedestrian accurate locations. Combined with pedestrian tracking, pedestrian recognition and other technologies, pedestrian detection technology can be applied to intelligent video monitoring, human behavior analysis, intelligent transportation, vehicle-assisted driving systems, intelligent robots, and other fields. On the other hand, TensorRT is a high-speed inference engine acceleration from NVIDIA, programmed internally using CUDA C. Optimization and acceleration of deep learning network structures using TensorRT allows the output network with the advantages of low latency and high throughput rate. The acceleration of pedestrian detection models deployed in embedded devices by TensorRT has significant advantages in lightweighting the models while obtaining better recognition efficiency.

Key words computer vision; target detection; pedestrian detection; CUDA; TensorRT

1 需求背景与意义

行人检测技术是利用计算机视觉技术判断图像或者视频序列中是否存在行人并给予准确定位，该技术与行人跟踪和行人识别等技术相结合，可应用于智能视频监控、人体行为分析、智能交通、车辆辅助驾驶系统和智能机器人等领域。

近年来得益于数据集的极大丰富和计算机计算能力的提高，深度学习技术获得了高速发展。深度学习技术通过构建复杂的神经网络来解释数据（如图像和文本）。因此，相对于传统的机器学习方法，深度学习技术能自动地从海量数据中学习数据的知识表示和隐藏关系，并基于这些表示和关系完成模式识别等任务。目前基于深度学习的图像分类算法在 ImageNets[1] 数据集上的分类准确度已经超过了人工分类；基于深度学习的目标检测算法在 COCO[2] 数据集上的平均准确率达到了 0.51[3]；基于深度学习的行人检测算法在 COCOPerson 数据集[4]、Caltech 数据集和 CityPerson 数据集[5] 等行人数据集上的检测准确率越来越高。

2 国内外研究概况

行人检测在计算机视觉领域中一直都是研究的热点和难点，不管是学术界或工业界都进行了大量的工作，从而也促进了行人检测技术的发展和应用[6–8]。行人检测需要找出图片或视频帧中的行人，并标注出其位置，和人脸检测类似，也是目标检测的典型问题，但由于人体的姿态多样，过于复杂，不同的人体外观差异巨大，并且人体的附着物以及其他物体的遮挡背景变化复杂等问题也非常常见，因此准确的检测出行人目标具有非常大的难度。

传统行人检测方法主要从运动目标和人体特征的角度考虑。基于运动目标的方法思想较为简单，其假设摄像机静止不动，利用背景建模算法提取出运动的前景目标，然后再用分类器对提取出的运动目标进行分类，判断其是否是行人目标，这种方法在特定环境下识别效果获取很好，但检测图像中运

动的物体更多情况下不仅只包含行人目标，在真实环境下，由于场景变化的复杂性，检测效果就不够理想[9–11]。

2012 年，由 Geoffrey Hinton 的学生 Alex Krizhevsky 设计的 AlexNet[12] 神经网络在当年的 ImageNet Large Scale Visual Recognition Challenge[13] 比赛中获得冠军，自此，深度学习技术得到了广泛的关注，并获得了巨大的发展，基于深度学习学到的特征具有很强的层次表达能力和很好的鲁棒性，可以更好的解决一些视觉问题。使用深度学习方法来处理行人检测时，一般来说会将其归类为目标检测的问题。

2014 年，由加州大学伯克利分校的 Ross Girshick 等提出的 R-CNN[14] 目标检测框架首次将卷积神经网络 (CNN) 应用到目标检测中，取得了当时目标检测领域最好的精度效果。R-CNN 的主要思想是，利用选择性搜索[15] 对图像中可能存在目标的区域提出候选框，然后使用 CNN 网络提取出候选框区域的特征，最后使用线性分类器判断是否存在某类物体并使用线性回归器对候选框的位置作出修正。

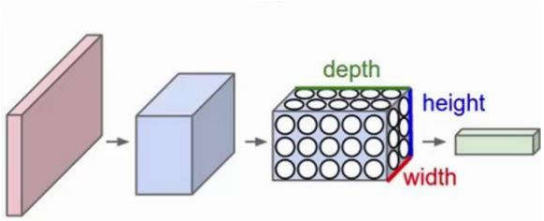


图 1　CNN 特征提取

R-CNN 在当时虽然取得了非常好的精度效果，但其缺点也非常明显，对选择性搜索提出的每一个候选框都会独立运行 CNN 获取特征，增大了计算量，也增加了冗余的计算。针对这个缺点，由 Kaiming He 等提出了 SPP-Net[16]。R-CNN 之所以需要对每个候选框独立做卷积计算提取特征，是因为卷积后一般还需要对卷积后的特征图做进一步的池化操作并将其输出固定到同样的长度才可以进行之后的分类和回归计算，而每个候选框本身的大小都不可能是相同的，因此 SPP-Net 提出一种新的空间金字塔池化层 (Spatial Pyramid Pooling, SPP)，这样设计的新的池化层，可以将不同长度的特征图池化为相同长度的特征向量，因而只需要对图像做一次卷积计算即可，不再需要对每个候选框进行单独计算，减少了计算量，从而使整体的检测速度获得了提升。

受 SPP-Net 中的空间金字塔池化层的启发，Ross Girshick 又提出了 Fast R-CNN[17]，在 SPP-Net 中需要对一个特征图池化为不同尺度的特征，构成了空间金字塔结构特征，但 Fast R-CNN 采用了空间池化的思想，只需要对最后的特征图做单层的空间池化，称为 ROI Pooling 层。Fast R-CNN 对比与 R-CNN 还将候选框分类和候选框回归合并成为多任务 (multi-task) 模型，并使用 multi-task loss 来训练网络，实现了端到端 (end-to-end) 的训练方式，使得训练难度大大降低。

Fast R-CNN 虽然已经取得了非常好的精度和速度，但是其仍然存在瓶颈，问题就在于 Fast R-CNN 仍然与 R-CNN 一样采用的是选择性搜索来提出候选框，这种方法通常无法获取质量较高的候选框而且非常耗时，影响了模型整体的速度和精度。为了解决这个瓶颈问题，Shaoqing Ren 等提出了 Faster R-CNN[18]，使用区域建议网络 (Region Proposal Network, RPN) 替代选择性搜索来生成候选框。Faster R-CNN 中将 RPN 放在最后一个卷积层之后，通过训练 RPN 可以直接得到候选区域，且 RPN 也是一个卷积网络，这样就进一步减少了计算量提升了检测速度而且得到的候选框更加精确，为后续的分类器也从一定程度上提升了精度。

R-CNN 系列将回归与分类分离开，需要生成候选区域，为双阶段的目标检测算法，单阶段目标检测算法不需要生成候选区域，以 SSD，YOLO 为代表。Liu[19] 等提出 SSD 目标检测算法，该算法优势是速度比较快，不需要进行感兴趣区域的生成，使用 6 个不同尺度的特征图进行预测，提高了检测能力。周腾[20] 等对 SSD 算法进行改进，使用 Focal Loss[21] 减少了网络正负样本不平衡状况，并且使用 DenseNet[22] 替代原始 VGG[23] 骨干网络，提高了特征提取的能力，提高了网络识别的精度。

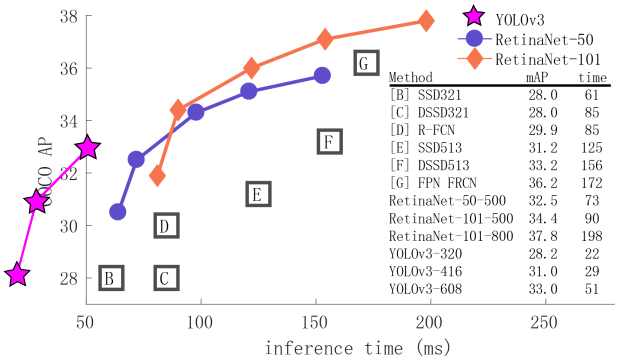


图 2　YoloV3 与其它网络对比

Redmon[24] 提出 YOLO 目标检测算法，YOLO 算法将目标检测当做回归问题，同时回归出目标的位置和类别，YOLOv2[25]，YOLOv3[26]，YOLOv4[27]，YOLOv5[28] 均是 YOLO 的改进。图2中，YOLOv3 由于采用了 DarkNet-53 作为骨干网络，并且在预测时采用三个尺度分别预测不同大小的目标，在目标检测任务获得了较高的准确率，并且保持了较好的速度。

表1为不同目标检测算法使用 TITAN V 推理计算，在 COCO 目标检测数据集上的速度及精度对比，比较其检测速度和精度。可以看出 YOLOv3 在速度和精度上能够实现较好多的平衡。就目前的研究而言，深度学习在目标检测，目标识别等方面具有超越传统方法的精确度，能够检测到更多的信息，而且可以完成多目标检测在同一个网络中的实现，但目前仍然存在计算量大，部署较为困难，而针对这些问题提出的轻量级深度学习网络存在着训练困难，精度较低，难以完成复杂数据集的检测需求。所以本文以 YOLOv3 为基础进行模型压缩研究，提高运行速度。

表 1　目标检测模型在 COCO[2] 上性能对比

网络模型	检测速度 (FPS)	检测精度 (mAP)
Faster R-CNN	7	59.1
YOLOv2	67	44.0
SSD	46	50.4
YOLOv3	45	57.9

3　技术路线

车辆行人目标检测算法的目的是在图片中定位出车辆和行人的位置及种类置信度信息，一般分为基于单阶段和双阶段的算法。相对于单阶段检测器，双阶段检测器将定位和分类离开，需要生成区域建议框来进行定位，消耗了大量的时间。基于检测器网络推理速度和准确度考虑，本节对单阶段的 YOLOv3 检测算法做速度和精度方面的改进。在下一章对此算法进行层和通道的剪枝，从而提高网络的运行速度，在保持较高精度的基础上仍然能够有较好的推理速度表现。

3.1　骨干网络

车辆行人目标检测网络是通过骨干网络（backbone）来进行图片特征的提取，骨干网络一般为卷积神经网络。YOLOv3 的骨干网络为如图3所示的 DarkNet-53，相比于 DarkNet-19 大幅增加了网络的深度。

	网络类型	卷积参数		输入特征图大小
	Convolutional	32	3×3	416×416
下采样	Convolutional	64	3×3/2	208×208
1×	Convolutional	32	1×1	
	Convolutional	64	3×3	
	Residual			208×208
下采样	Convolutional	128	3×3/2	104×104
2×	Convolutional	64	1×1	
	Convolutional	128	3×3	
	Residual			104×104
下采样	Convolutional	256	3×3/2	52×52
8×	Convolutional	128	1×1	
	Convolutional	256	3×3	
	Residual			52×52
下采样	Convolutional	512	3×3/2	26×26
8×	Convolutional	256	1×1	
	Convolutional	512	3×3	
	Residual			26×26
下采样	Convolutional	1024	3×3/2	13×13
4×	Convolutional	512	1×1	
	Convolutional	1024	3×3	
	Residual			13×13

图 3　DarkNet-53 网络结构

DarkNet-53 采用了 23 处残差结构来增加网络的深度，从而提高了网络提取特征的能力，网络中大部分采用 1*1 和 3*3 的卷积结构减少计算量，没有池化层，是一个全卷积网络，网络的下采样（特征图减小为原来的二分之一）也全部由卷积来完成，当网络下采样时，通道数变为原来的二倍，有利于减少网络中信息的损失，并且增加信息特征的维度，更能提取到网络的抽象特征，使得精确度提升。

3.2　YOLO 后处理过程

YOLOv3 是由三个不同尺度的特征图来直接预测目标的种类和位置，得到边界框之后又通过置信度过滤和非极大值抑制(Non-Maximum Suppression, NMS）去除同一目标的检测到的重复的框。

由于最后用于预测的特征图是通过不断的下采样得到的，所以一个格子的感受野远大于格子本身，每个格子基于三个锚框（Anchor）预测三个种类和边界框，所以特征图的通道数为 3*（5+C），5 为边界框位置信息 t_x 、t_y 、t_w 、t_h 和有物体的概率 p_0 ，C 为要预测的种类个数，在本文车辆行人目标检测算法中，类别为两类，故类别概率为 p_1 、p_2 ，特征图的通道数为 3*（5+2），特征图输出为

13*13*21。

由于边界框的位置及种类信息是逐格来进行处理的，目标物体在特征图中实际位置的中心点落在格子内，中心点位置用相对于格子左上角的坐标(c_x, c_y)，偏移来表示，目标宽高用相对于预设锚框的偏移来表示。

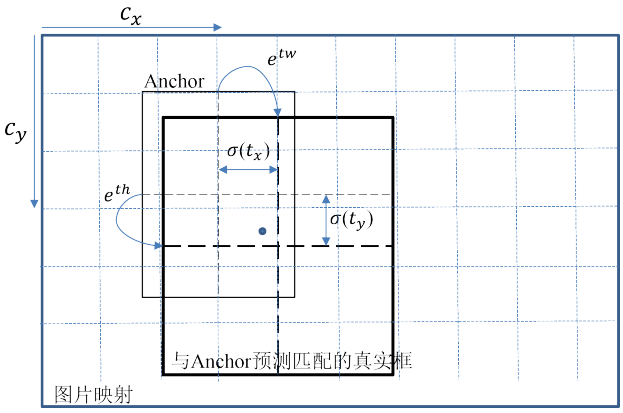


图 4 Anchor 预测

所以在三个不同尺度上总共有（13*13+26*26+52*52）*3 即 10647 个边界框信息，从 10647 个边界框中得到我们需要的边界框就需要进行置信度阈值的过滤和非极大值抑制来完成。使用置信度阈值过滤掉低置信度的边界框，置信度阈值为本格子中存在物体的概率乘以目标分类的概率，通过设置一个合适的置信度阈值筛选掉低置信度的预测框，并将筛选后的结果输入到非极大值抑制函数。

每个格子周围的格子都可能预测到原图中的同一个目标,所以通过计算重叠格子的交并比(IOU)来删除重叠度高的边界框，从而得到我们需要预测的目标，置信度阈值和非极大值抑制的阈值都是可调的，通过调节不同的阈值，得到不同的预测结果，可以衡量模型在不同工况下的表现。

4 总体结构

本文旨在对行人检测算法进行研究，并通过剪枝、量化等模型压缩算法来减少参数量、计算量获得车辆行人检测轻量级模型，对于车辆行人检测轻量级模型进行层融合、低精度推理等，最终提高网络的运行速度。分为以下几个部分：

（1）选择合适的行人训练数据集，并且从数据集中提取本文所需的行人类别数据，对同类型的数据归类及模型训练数据集格式转化，并将数据集分为训练集和测试集，训练集用于行人检测网络的训练，测试集用于评估行人检测网络及轻量级网络模型的性能。

（2）本文以基于 Yolov3 的目标检测算法进行行人的检测，以训练集数据完成对行人检测网络的训练，针对数据集行人目标数量不平衡情况，对损失函数等进行修改以提高检测的精度，并以平均准确率以及实际检测效果进行评估。

（3）针对行人检测模型的冗余问题，对模型进行稀疏化训练，然后进行通道剪枝和层剪枝等优化方式，获得轻量级网络模型，剪枝完成后导入剪枝后的网络结构和权重进行重新训练，使得模型的性能恢复，对比剪枝前后网络的大小、参数量、计算量（Floating Point Operations，FLOPs）、运算速度、准确度等指标，评判剪枝后模型的性能。

（4）将轻量级模型通过 Jetson TX2 平台进行加速推理实验，使用 TensorRT 优化网络，首先将 DarkNet 格式模型权重和网络结构导出为开放式神经网络交换（Open Neural Network Exchange，ONNX）格式，然后再将 ONNX 模型转化为 TensorRT 的推理引擎，使用半精度浮点数（Half-Precision Floating-Point Format，FP16）的低精度推理，进行行人的检测，并衡量运行的速度及实际检测效果。

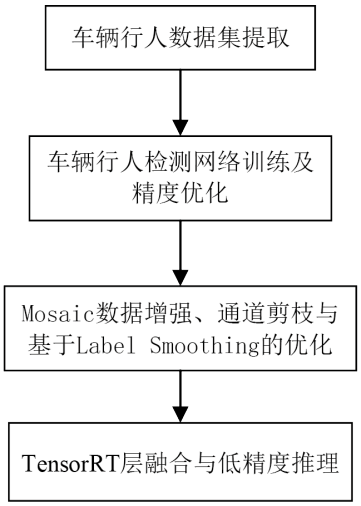


图 5 总体结构

5 实施计划

5.1 Mosaic 数据增强

Mosaic 数据增强方法是一种常见的数据增强，由 Alexey Bochkovskiy 等人[27]提出，其主要思想就是将四张图片进行随机裁剪，再拼接到一张图上作为训练数据，这样做的好处是丰富了图片的背景，并且四张图片拼接在一起变相地提高了 batch size，

在进行 batch normalization 的时候也会计算四张图片，所以对本身 batch size 不是很依赖，单块 GPU 就可以训练 Yolov3，训练的过程如图6所示

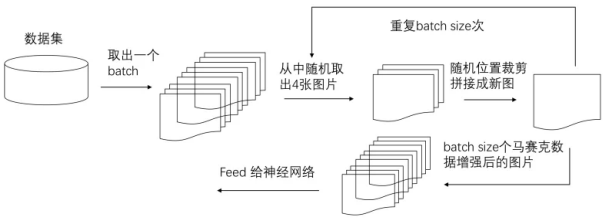


图 6　Mosaic 数据增强流程

相较于 Mixup[29]、Cutout[30]、CutMix[31]，Mosaic 数据增强在训练过程中具有以下优点：

1. 在训练过程中不会出现非信息像素，从而能够提高训练效率；

2. 保留了 regional dropout 的优势，能够关注目标的 non-discriminative parts；

3. 通过要求模型从局部视图识别对象，对 cut 区域中添加其他样本的信息，能够进一步增强模型的定位能力；

4. 不会有图像混合后不自然的情形，能够提升模型分类的表现；

5. 训练和推理代价保持不变。

而在本次实验中采用 Mosaic 方法的增强版——Mosaic-9，即对 9 张图片随机裁剪、随机缩放、随机排列组合成一张图片，其细节如图7所示。Mosaic-9 数据增强利用了 9 张图片，对 9 张图片进行拼接，每一张图片都有其对应的框，将 9 张图片拼接之后就获得一张新的图片，同时也获得这张图片对应的框，然后将这样一张新的图片传入到神经网络当中去学习，相当于一次性传入 9 张图片进行学习，这极大丰富了检测物体的背景，且在标准化计算的时候就会计算 9 张图片的数据。操作完成之后然后再将原始图片按照 9 块相对位置进行摆放。完成 9 张图片的摆放之后，利用矩阵的方式将 9 张图片它固定的区域截取下来，然后将它们拼接起来，拼接成一张新的图片，新的图片上含有框等一系列的内容。拼接完成之后得到的新的一张图片，在拼接的时候部分图会被相邻近的图覆盖掉了，拼接的时候很有可能也会把另外的图中的框给覆盖掉，这些问题都会在最后的对框进行处理：当图片的框（或者图片本身）超出两张图片之间的边缘（即设置的分割线）的时候，就需要把这个超出分割线的部分框或者图片的部分）处理掉，进行边缘处理。

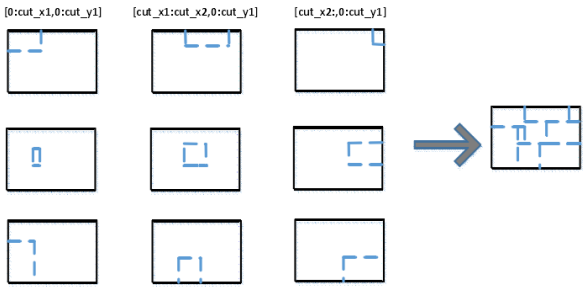


图 7　Mosaic-9 数据增加细节

5.2　MobileNet

MobileNet 是 Google 提出的一种用于移动和嵌入式视觉应用的称为的高效模型。MobileNet 基于一种流线型的架构，该架构使用深度可分离卷积 (Depthwise separable convolution) 来构建轻量级的深度神经网络。深度可分卷积是一种分解卷积的形式，它将标准卷积分解为深度卷积和一个被称为点卷积的 1×1 卷积。其不仅可以降低模型计算复杂度，而且可以大大降低模型大小[32]。

MobileNet V2 是在 2018 年提出，对 V1 的卷积单元进行了改进，主要引入了线性瓶颈 (Linear bottleneck) 和反向残差 (Inverted residuals)[33]：

1. 线性瓶颈即去掉卷积单元中最后一个 ReLU 函数，不使用 ReLU 激活，而用线性变换来代替原本的非线性激活变换。这样做是为了避免 ReLU 对低维度运算时对特征的破坏，造成信息的丢失，使得卷积层参数为空；

2. 反向残差即先升维到高维空间进行卷积操作来提取特征，随后再进行降维。由于深度卷积本身没有改变通道的能力，如果输入通道很少，深度卷积只能在低维度上工作，于是使用点卷积升维，在一个更高维的空间中进行卷积操作来提取特征，最后再次使用点卷积进行降维，以增加非线性通道转换的能力。

MobileNet V3 结构上相较于 V2 而言，引入了 SE 模块、修改了尾部结构、修改了通道数、改用了 h-swish 激活函数[34]：

1. 引入 SE 模块，即通过显式地建模网络卷积特征通道之间的相互依赖关系，来提高网络所产生表示的质量。

2. 修改尾部结构，即将平均池化层前的 1×1 卷积放置层后，因为 1×1 卷积会占据大量的运算时间，放置在池化层后可以减少计算量。

3. 修改通道数，即将第一个卷积核的通道数由 32 修改为 16。

4. 为提高网络准确性，可以使用 swish 非线性激活函数替换 ReLU 激活函数，其公式为：

$$\text{swish}(x) = x \cdot \sigma(x) \tag{1}$$

但由公式可知，计算 swish 需要计算 sigmoid 函数，这会提高计算成本。而 ReLU6 函数在许多软硬件框架中都已实现，易于量化部署，即使以 16 位浮点数或 8 位整型低精度运算时性能也较好，且计算推理速度快。于是使用 h-swish 激活函数来近似 swish 函数，其公式如下：

$$h-\text{swish}[x] = \frac{x \cdot (\text{ReLU6}(x+3))}{6} \tag{2}$$

经过试验验证，如图8所示，h-swish 激活函数并不会降低网络精度，即使用该近似能在保持精度的情况下加快速度。

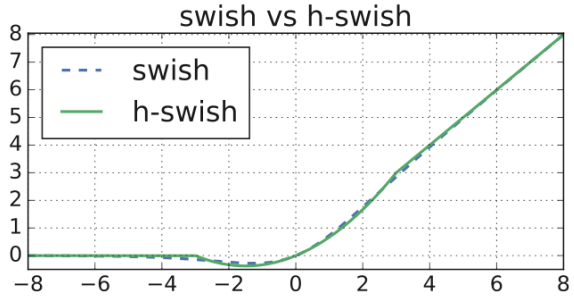


图 8 swish 与 h-swish 激活函数对比

MobileNet V3 是目前最新于是效果最好的版本，同时其推出 Large 版本和 Small 版本，适用于不同算力资源的情况。为实现 YOLOv3 网络的轻量化、小型化，最终本实验采用 MobileNet V3 Small 取代 YOLOv3 的 Backbone 特征提取网络来进行特征提取。

5.3 通道剪枝

深度卷积神经网络 (CNN) 的部署在许多现实世界的应用很大程度上受到其高计算成本的阻碍，这是因为 CNN 优异的性能表现通常来源于上百万可训练的参数，而且在推理期间 CNN 的中间激活值和响应存储空间甚至需要比存储模型参数的空间还要大，以及因为在高分辨率图片上卷积操作可能会出现计算密集从而使计算时间很长。

通道剪枝是通过在网络中以一种简单但有效的方式强制信道级稀疏性来实现的，直接适用于现代 CNN 架构，在训练过程中引入了最小的开销，并且生成的模型不需要特殊的软硬件加速器。通道剪枝可以减小模型大小、减少运行时的内存占用、在不影响精度的同时降低计算操作数，使得其广泛应用在实际场景中[35]。

目前的剪枝算法分为结构化枝和非结构化剪枝。非结构化剪枝的方法需要对稀疏连接的网络进行量化和编码才能减少模型的实际存储空间，而且需要采用专业的硬件设备和计算方式才能实现模型推理加速。采用结构化剪枝作为卷积神经网络的轻量化方法，通道剪枝以模型重构的方式筛选神经网络中存在的一些冗余连接，这些结构对于模型性能的贡献很小，去掉这部分神经元能够有效降低模型复杂度，同时几乎不会对网络的精度产生影响，甚至还能改善网络的综合性能。

本实验在神经网络中的 BN （batch normalization) 层引入可学习的参数 γ 和 β 加快网络的训练和收敛速度，通过平移和缩放对通道数据进行归一化处理，在迭代训练中学习网络的特征分布，公式如下所示：

$$\hat{z} = \frac{z_{\text{in}} - \mu_B}{\sqrt{\sigma_B^2 + \varepsilon}}, z_{\text{out}} = \gamma\hat{z} + \beta \tag{3}$$

模型通道剪枝示意图如图9所示：

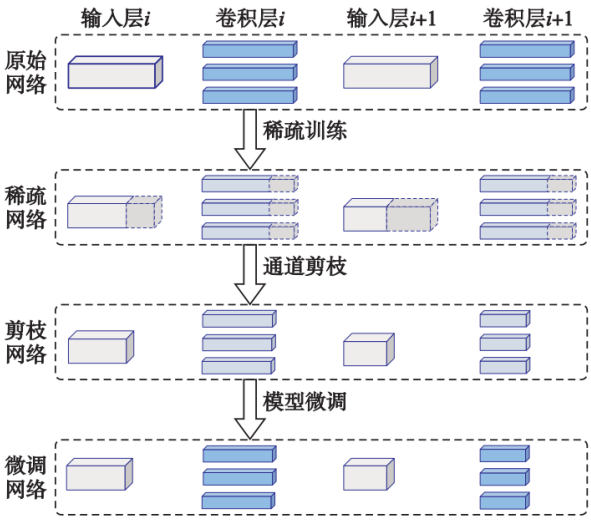


图 9 模型通道剪枝示意图

5.4 Label Smoothing

Label Smoothing[36] 标签平滑最早应用于分类算法中，后面引入目标检测算法中。目标检测算法分为分类与回归两个分支，其主要作用于分类分支，属于正则化方法中的一种。本质上，标签平滑将帮助模型围绕错误的标签数据进行训练，从而提

高其健壮性和性能。标签平滑可以降低模型的可信度，并防止模型下降到过拟合所出现的损失的深度裂缝里，其公式为

$$q_i' = (1-\varepsilon)q_i + \frac{\varepsilon}{K} \tag{4}$$

其中 q_i 表示真实标签，ε 是一个非常小的常数，K 代表分类的类别数。经过 Label Smoothing 后能过减少模型过度依赖标签的问题，有效改善标签准确性不高的情况，故在 Prediction 层中引入 Label Smoothing 标签平滑方法。

5.5 TensorRT 优化

NVIDIA TensorRT 是基于 Nvidia CUDA 编程模型的 SDK，主要用于高性能深度学习推理[37]。具体优化结构如图10所示，TensorRT 提供 api 和解析器来从所有主要的深度学习框架中导入经过训练的模型，经过权重与激活精度校准、层与张量融合、内核自动调整、动态张量显存和多流执行，然后生成可部署在各种环境的优化运行时引擎。[38]

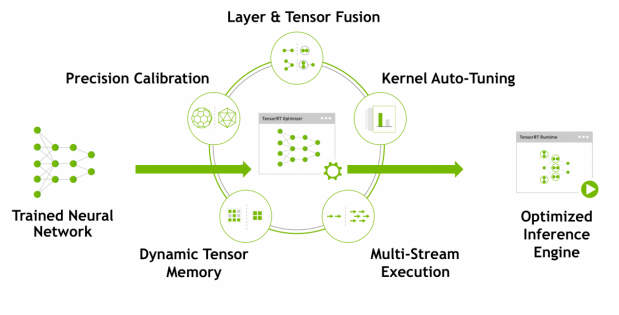


图 10 TensorRT 优化原理

TensorRT 解析网络计算图，通过网络层及张量的合并来达到加速的效果，对计算流图的优化没有改变底层的计算，只是对计算图进行了重构，使得计算可以高效运行。图12 以 Inception 网络为例说明网络层及张量合并。

图12（a）为原始网络结构图，当深度学习框架推理时，每一层会调用多个函数，每个函数均需要在 GPU 上运行，主机（host）与 GPU（device）之间频繁进行内存传输，多个 CUDA kernel 函数被启动，kernel 函数计算比启动和读取消耗的时间短，所以内存会限制 GPU 资源的利用。TensorRT 网络和张量合并减少了 kernel 函数启动，减少层间内存输入输出。卷积层（Conv 层），偏移层或 BN 层和激活层（Relu 层）被整合成一个“CBR”kernel。TensorRT 还可以将相同输入和相同尺寸的滤波器整合，将图12（b）中几个相同输入的“1*1 CBR”kernel 整合为图12（c）中一个“1*1 CBR”kernel。同时 TensorRT 还可以去掉 Concat 层，通过预先分配出缓存区，将结果跨步写入缓冲，Concat 层是将两个输出矩阵直接拼接到一起，TensorRT 可以实现将每个矩阵直接连接到缓冲区，替代拼接操作。所以可以直接连接到网络下一层的输入（Next input）。图12（d）中的左侧“1*1 CBR”分路与右侧的 Maxpool 分路也可以通过 GPU 并行进行计算，加快深度网络推理速度。

在我们的项目中，为了使 yolo 训练的模型进一步轻量化，我们将模型送入 TensorRT 中优化产生 Engine 引擎，然后再应用在 GPU 推理中，优化流程如图11所示。在最后的 GPU 推理阶段，优化后的引擎被反序列化解析，当推理请求发出时，输入数据从 CPU 复制到 GPU，推理完成后再以异步方式返回结果至 CPU。

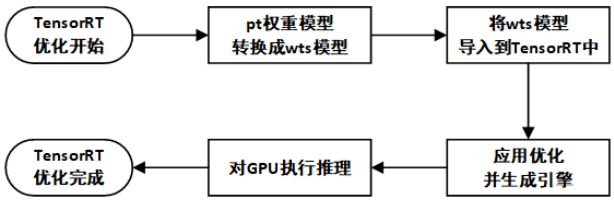
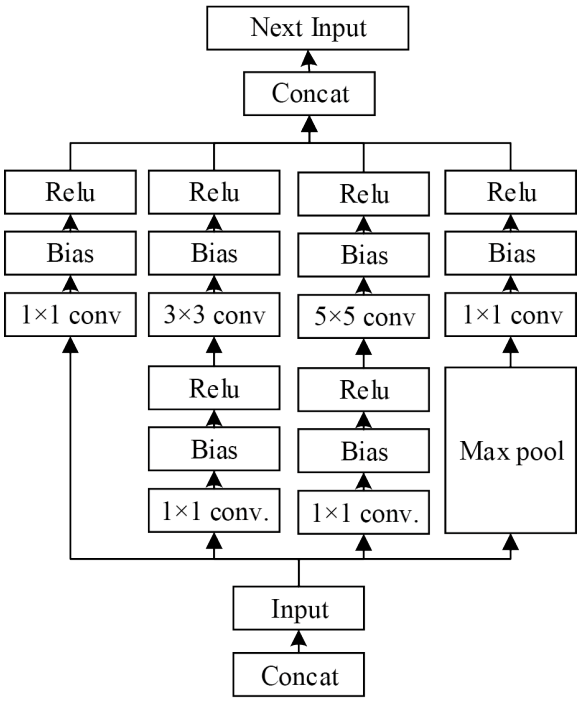
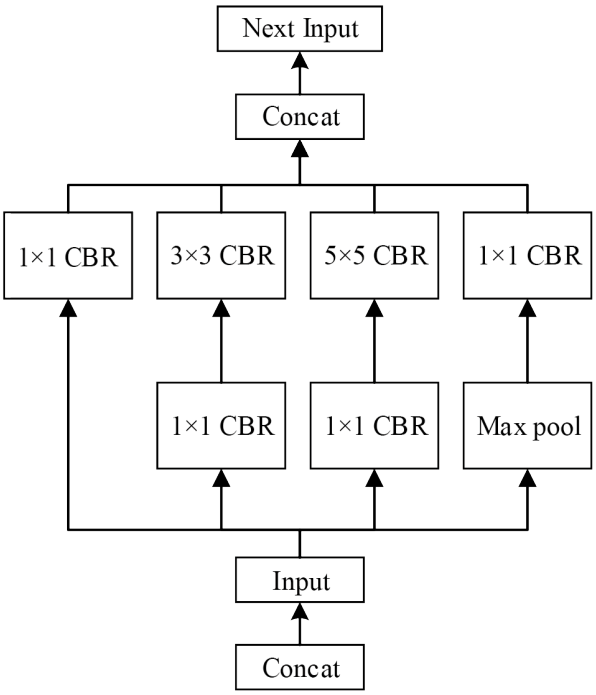


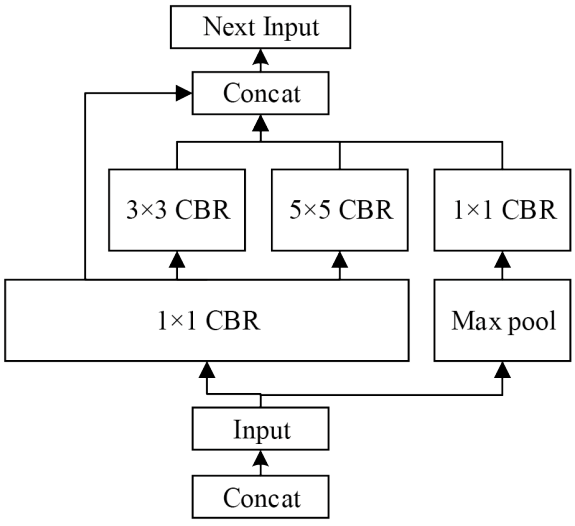
图 11 TensorRT 处理模型流程图



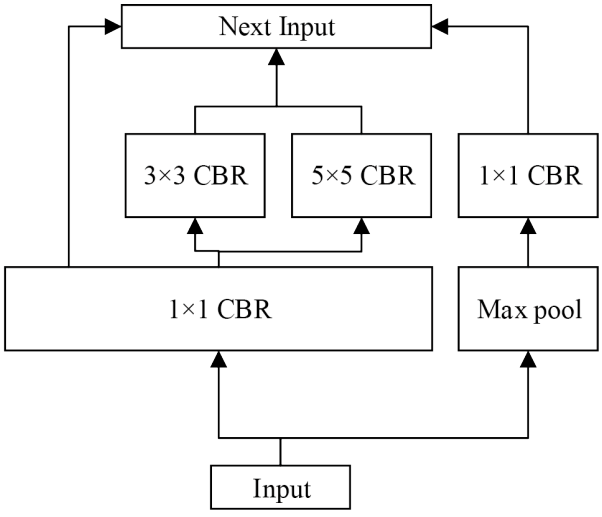
(a)原始网络结构



(b)层合并后网络结构



(c)相同模块合并后结构



(d)最终网络结构

图 12　Inception 网络层及张量合并图

参考文献

[1] DENG J, DONG W, SOCHER R, et al. Imagenet: A large-scale hierarchical image database[C]//2009 IEEE conference on computer vision and pattern recognition. [S.l.]: Ieee, 2009: 248-255.

[2] LIN T Y, MAIRE M, BELONGIE S, et al. Microsoft coco: Common objects in context[C]//European conference on computer vision. [S.l.]: Springer, 2014: 740-755.

[3] ZOPH B, CUBUK E D, GHIASI G, et al. Learning data augmentation strategies for object detection[C]//European Conference on Computer Vision. [S.l.]: Springer, 2020: 566-583.

[4] ESS A, LEIBE B, VAN GOOL L. Depth and appearance for mobile scene analysis[C]//2007 IEEE 11th international conference on computer vision. [S.l.]: IEEE, 2007: 1-8.

[5] ZHANG S, BENENSON R, SCHIELE B. Citypersons: A diverse dataset for pedestrian detection[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2017: 3213-3221.

[6] 许言午, 曹先彬, 乔红. 行人检测系统研究新进展及关键技术展望[J]. 电子学报, 2008, 36(5):962.

[7] 苏松志. 行人检测若干关键技术研究[D]. [出版地不详]: 万方数据资源系统, 2011.

[8] VIOLA P, JONES M J, SNOW D. Detecting pedestrians using patterns of motion and appearance[J]. International Journal of Computer Vision, 2005, 63(2):153-161.

[9] DOLLAR P, WOJEK C, SCHIELE B, et al. Pedestrian detection: An evaluation of the state of the art[J]. IEEE transactions on pattern analysis and machine intelligence, 2011, 34(4):743-761.

[10] GERONIMO D, LOPEZ A M, SAPPA A D, et al. Survey of pedestrian detection for advanced driver assistance systems[J]. IEEE transactions on pattern analysis and machine intelligence, 2009, 32(7):1239-1258.

[11] 贾慧星, 章毓晋. 车辆辅助驾驶系统中基于计算机视觉的行人检测研究综述[D]. [出版地不详]: 自动化学报, 2007.

[12] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks[J]. Advances in neural information processing systems, 2012, 25:1097-1105.

[13] RUSSAKOVSKY O, DENG J, SU H, et al. Imagenet large scale visual recognition challenge[J]. International journal of computer vision, 2015, 115(3):211-252.

[14] 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. [S.l.: s.n.], 2014: 580-587.

[15] UIJLINGS J R, VAN DE SANDE K E, GEVERS T, et al. Selective search for object recognition[J]. International journal of computer vision, 2013, 104(2):154-171.

[16] 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.

[17] GIRSHICK R. Fast r-cnn[C]//Proceedings of the IEEE international conference on computer vision. [S.l.: s.n.], 2015: 1440-1448.

[18] 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, 28:91-99.

[19] LIU W, ANGUELOV D, ERHAN D, et al. Ssd: Single shot multibox detector[C]//European conference on computer vision. [S.l.]: Springer, 2016: 21-37.

[20] 周腾, 兰时勇. 交通场景下的行人和车辆实时检测算法[J]. 现代计算机, 2019(21):50-55.

[21] 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. [S.l.: s.n.], 2017: 2980-2988.

[22] HUANG G, LIU Z, VAN DER MAATEN L, et al. Densely connected convolutional networks[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2017: 4700-4708.

[23] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[24] 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. [S.l.: s.n.], 2016: 779-788.

[25] REDMON J, FARHADI A. Yolo9000: better, faster, stronger[C]// Proceedings of the IEEE conference on computer vision and pattern recognition. [S.l.: s.n.], 2017: 7263-7271.

[26] REDMON J, FARHADI A. Yolov3: An incremental improvement[J]. arXiv preprint arXiv:1804.02767, 2018.

[27] BOCHKOVSKIY A, WANG C Y, LIAO H Y M. Yolov4: Optimal speed and accuracy of object detection[J]. arXiv preprint arXiv:2004.10934, 2020.

[28] JOCHER G. TPC-W yolov5[Z]. [S.l.: s.n.], 2020.

[29] ZHANG H, CISSE M, DAUPHIN Y N, et al. mixup: Beyond empirical risk minimization[Z]. [S.l.: s.n.], 2018.

[30] DEVRIES T, TAYLOR G W. Improved regularization of convolutional neural networks with cutout[Z]. [S.l.: s.n.], 2017.

[31] YUN S, HAN D, OH S J, et al. Cutmix: Regularization strategy to train strong classifiers with localizable features[Z]. [S.l.: s.n.], 2019.

[32] HOWARD A G, ZHU M, CHEN B, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications[Z]. [S.l.: s.n.], 2017.

[33] SANDLER M, HOWARD A, ZHU M, et al. Mobilenetv2: Inverted residuals and linear bottlenecks[Z]. [S.l.: s.n.], 2019.

[34] HOWARD A, SANDLER M, CHU G, et al. Searching for mobilenetv3 [Z]. [S.l.: s.n.], 2019.

[35] LIU Z, LI J, SHEN Z, et al. Learning efficient convolutional networks through network slimming[Z]. [S.l.: s.n.], 2017.

[36] DAI J, LI Y, HE K, et al. R-fcn: Object detection via region-based fully

convolutional networks[Z]. [S.l.: s.n.], 2016.

[37] NVIDIA. NVIDIA TensorRT 可编程推理加速器[Z]. [出版地不详: 出版者不详], 2021.

[38] ABBASIAN H, PARK J, SHARMA S, et al. Speeding up deep learning inference using tensorrt[Z]. [S.l.: s.n.], 2020.