第 59 卷 第 16 期/2022 年 8 月/激光与光电子学进展 研究论文

激光与光电子学进展 先进成像

面向小目标检测的轻量化YOLOv3算法

张官荣[1]，陈相[1]，赵玉[1*]，王建军[2]，易国彪[3]

[1]空军工程大学航空工程学院，陕西 西安 710038；

[2]西北工业大学电子信息学院，陕西 西安 710129；

[3]中国人民解放军95696部队，重庆 405200

摘要 为了提高面向遥感图像目标检测的YOLOv3-CS算法的检测速度，提出了一种基于Batcg Normalization（BN）层γ参数的自适应稀疏因子调整算法。以γ作为通道的重要性判断依据对YOLOv3-CS进行剪枝，得到YOLOv3-CSP目标检测模型。实验结果表明，所提剪枝方法在mean Average Precision（mAP）损失仅为0.22%的情况下，使YOLOv3-CS的模型大小压缩了95.92%，检测速度提高了173%。所提YOLOv3-CSP可以应用于检测精度和实时性要求较高的场合。

关键词 图像处理；YOLOv3；稀疏训练；模型剪枝

中图分类号 TP391.4 **文献标志码** A **DOI**：10.3788/LOP202259.1610008

Lightweight YOLOv3 Algorithm for Small Object Detection

Zhang Guanrong[1], Chen Xiang[1], Zhao Yu[1*], Wang Jianjun[2], Yi Guobiao[3]

[1]*Aeronautics Engineering College, Air Force Engineering University, Xi'an* 710038, *Shaanxi, China*;

[2]*School of Electronics and Information, Northwestern Polytechnical University, Xi'an* 710129, *Shaanxi, China*;

[3]*Unit 95696 of the Chinese People's Liberation Army*, 405200, *Chongqing, China*

Abstract To improve the detection speed of the YOLOv3-CS algorithm for remote sensing image target detection, an adaptive sparse factor adjustment algorithm based on the γ parameter of the Batch Normalization (BN) layer is proposed. YOLOv3-CS was pruned to obtain YOLOv3-CSP using γ as the basis for determining the channel importance. The experimental results show that the proposed pruning method reduces the model size by 95.92%, while increasing the detection speed by 173%, when the mean Average Precision (mAP) loss of YOLOv3-CS is 0.22%. The YOLOv3-CSP can be applied to certain occasions requiring high detection accuracy and real-time performance.

Key words image processing; YOLOv3; sparse training; model pruning

1 引 言

目标检测作为计算机视觉领域的重要技术，近年来发展迅速。例如以R-CNN[1]系列算法为代表的检测框架成为了基于卷积神经网络（CNN）目标检测算法研究的热点。自此，基于深度神经网络的目标检测算法经过飞速发展，逐渐出现了两类目标检测算法：双阶段检测算法（two-stage detection）和单阶段检测算法（one-stage detection）。双阶段目标检测算法以R-CNN为代表，有SPPNet[2]、Fast R-CNN[3]、Faster R-CNN[4]等；单阶段目标检测算法以You Only Look Once（YOLO）[5]为代表，有YOLO9000[6]、YOLOv3[7]、YOLOv4[8]、SSD[9]和RetinaNet[10]等。双阶段目标检测算法的检测精度虽然较高，但是难以适用于实时性要求较高的场合；单阶段目标检测算法的检测精度虽然较低，但是其frame per second（FPS）较高，可以适用于实时性要求较高的场合。

在众多的目标检测算法中，YOLOv3既有较高的精测精度又有较快的FPS，并且具有结构简单的优点，被广泛应用。YOLOv3在保留了YOLO和YOLO9000的图像栅格化、目标中心栅格负责预测目标等特点的同时，引入了多尺度特征融合的思想，既

收稿日期：2021-03-29；**修回日期**：2021-05-19；**录用日期**：2021-07-13

基金项目：陕西省自然科学基金（2021JM-225）

通信作者：*5325975@qq.com

保证了大目标的检测精度，又提高了小目标的检测精度。但是对于以小目标为主的遥感图像目标检测，YOLOv3的检测精度还有待进一步提高。文献[11]提出的YOLOv3-CS与YOLOv3相比，对遥感图像小目标的检测有较大优势，但是推理速度仍然难以满足高实时性的要求，因此需要压缩模型来提高性能。压缩模型的方法可分为剪枝[1213]、量化[1415]、低秩分解[1617]和知识蒸馏[1819]，其中剪枝是目前模型压缩中应用最多的方法。量化是用更少的位数表示权重参数，只针对参数本身，对模型的计算量并没有改变；低秩分解是使用小卷积核的线性组合来表示大卷积核，实现上较为复杂；知识蒸馏是用一个简单的网络拟合一个复杂的网络并作用于剪枝后的模型，可以有效提高剪枝模型的精度。模型剪枝通过一种有效的评判准则来判断CNN中参数的重要性，然后将不重要的连接或通道移除从而减少模型的冗余参数。剪枝按照粒度分为非结构化剪枝和结构化剪枝。非结构化剪枝对权重进行剪枝，需要特殊的硬件设备和计算库的支持。与非结构化剪枝不同，结构化剪枝更具有通用性，可以在通用硬件设备和计算库中轻松实现，因此实际应用中多使用结构化剪枝。结构化剪枝又可以分为层剪枝、卷积核剪枝和通道剪枝。结构化剪枝有3种思路：1）基于重要性因子进行剪枝[20]；2）利用重建误差来指导剪枝[21]；3）优化目标的变化来衡量通道的敏感性[22]。

结构化剪枝以卷积核、卷积通道或层为剪枝对象，剪枝后的卷积结构仍然完整，可以在通用计算库和硬件设备上运行。本文根据文献[11]提出的YOLOv3-CS目标检测算法模型结构的特殊性，利用基于重要性因子的结构化剪枝方法对YOLOv3-CS进行压缩，以降低其模型大小，提高推理速度。

2　YOLOv3-CS目标检测模型

YOLOv3-CS的结构如图1所示，是在YOLOv3的基础上改进而来的。首先根据不同尺度特征图的重要性，对YOLOv3的backbone——Darknet-53进行了改进来增强对小目标特征的提取能力；其次引入Receptive Field Block（RFB）结构增大浅层特征图的感受野来提升小目标检测精度；最后优化了anchor boxes及其分配原则。与YOLOv3相比，YOLOv3-CS在遥感图像目标检测中不论是mean Average Precision（mAP）还是F1系数都有较大优势，而且所需存储空间更低。

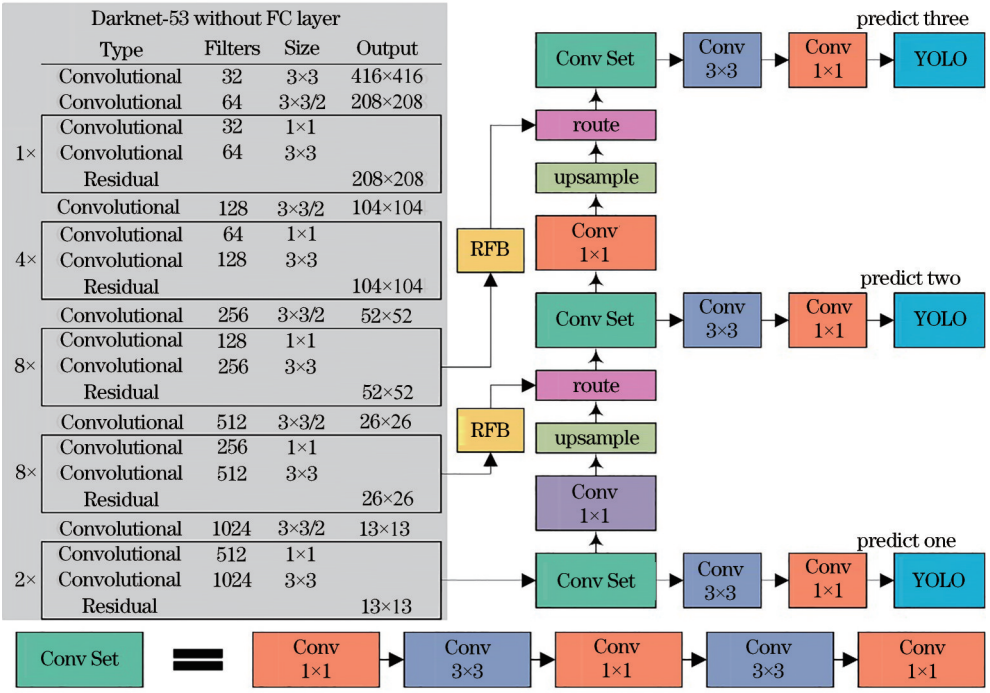

图1　YOLOv3-CS结构图

Fig. 1　Structure diagram of YOLOv3-CS

3　基于Batch Normalization（BN）层缩放因子的剪枝准则

文献[23]提出的Network slimming法利用网络中BN层的缩放系数γ判断卷积层输出通道的重要性，在损失函数中直接添加缩放系数γ的正则项进行稀疏训练，即

$$L=L(W)+\lambda\sum_{\gamma\in\Gamma}R_{g}(\gamma),\tag{1}$$

其中：$L(W)$为模型损失函数；$R_{g}(\gamma)$为稀疏正则项；Γ为γ的集合；λ为稀疏因子。

前向传播结束后，γ参数为

$$\gamma^{*}=\arg\min_{\gamma}\left[L(W)+\lambda\sum_{\gamma\in\Gamma}R_{g}(\gamma)\right]。\tag{2}$$

通过随机梯度下降(SGD)优化式(2),并求梯度有

$$\nabla\gamma^{*}=\nabla_{\gamma}\lambda\sum_{\gamma\in\Gamma}R_{g}(\gamma)。\tag{3}$$

设学习率为η,则参数γ更新为

$$\gamma'=\gamma-\eta\nabla\gamma^{*}=\gamma-\eta\lambda\nabla_{\gamma}\sum_{\gamma\in\Gamma}R_{g}(\gamma)。\tag{4}$$

因此在更新γ参数时,只需要在γ参数中添加正则项的偏导即可,对于整个网络来说,增加的计算量可以忽略不计。实际的稀疏因子为$\eta\lambda$,因此在稀疏训练的过程中,学习率和稀疏因子配合调整才能达到较好的稀疏效果。稀疏训练后,网络中部分γ参数接近0,然后将接近0的γ对应的通道移除。

式(1)中的正则项通常为L1正则,即式(1)可以改写为

$$L=L(W)+\lambda\sum_{\gamma\in\Gamma}|\gamma|。\tag{5}$$

4　基于剪枝的YOLOv3-CS

YOLOv3-CS的最小单元是DarknetConv2D+BN+Leaky ReLU(DBL)结构,则YOLOv3-CS的通道剪枝对象为网络中每层的DBL单元。通道剪枝原理如图2所示,第j层的C_{j2}和C_{j4}通道对应的缩放系数γ接近于0,对网络的贡献忽略不计,可以将其移除,经过剪枝后,第j层的通道数变为$n-2$,将获得一个更为紧凑的网络。

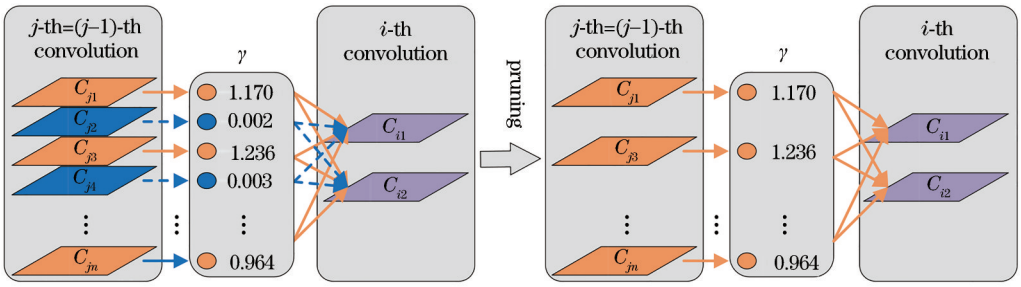

图2　通道剪枝示意图

Fig. 2　Schematic diagram of channel pruning

YOLOv3-CS的backbone属于多分支网络,其backbone包含5种不同的Residual block结构,分别对应5种不同尺度的特征,每个Residual block都由1×1和3×3两种尺度的卷积核构成。Residual block顺序连接。由于shortcut的存在,Residual block的输入通道数与其内部3×3卷积核个数必须相等,才能保证加法运算,因此在剪枝时,Residual block内部1×1的卷积核可以任意剪枝,不受约束,但是3×3的卷积核剪枝后的数量需要与Residual block的输入通道数保持一致。Residual block的连接根据输入的不同分为两种,如图3所示,[图3(a)]中,Residual block的输入为下采样层的输出,因此在剪枝后,Residual block内部3×3的卷积核的个数需要与下采样层的输出通道数保持一致;[图3(b)]中,Residual block的输入为与之相连的前一个Residual block的输出,因此剪枝后,Residual block内部3×3卷积核的个数需要与前一个Residual block的输出通道数保持一致。总的来讲,剪枝后,下采样之后所有的Residual block内部3×3的卷积核个数必须与下采样层的卷积核个数保持一致。YOLOv3-CS的backbone之外的卷积层没有shortcut层,可以不受约束地进行剪枝。另外,检测头连接的卷积层和RFB结构的前一个卷积层不参与剪枝。

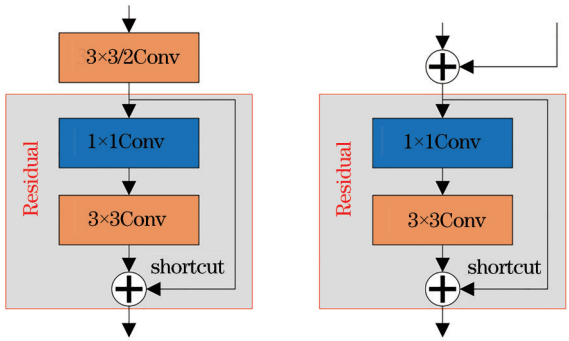

图3　Residual block结构图。(a)输入为下采样的输出;(b)输入为Residual的输出

Fig. 3　Structure diagrams of Residual block. (a) Input is output of downsample; (b) input is output of Residual

剪枝流程如图4所示,在对网络进行剪枝前需要对网络进行稀疏训练,剪枝后对网络进行重训练进行微调。为了能够使模型在精度不损失的情况下获得较高的稀疏程度,本实验组提出了自适应调整稀疏因子的算法(算法1)。首先将γ参数划分为K组,然后设定稀疏因子λ,第k组的初始稀疏因子$\lambda_{k}=\lambda$,再根据每一

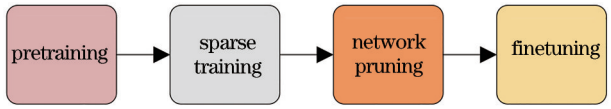

图4　剪枝流程图

Fig. 4　Flow chart of pruning

组的稀疏情况自适应调整各组的稀疏因子λ_k。

Algorithm 1 Adaptive adjustment of sparsity factor:λ

set:the group number of parameter γ is K, the sparsity factor is λ, the number of iterations is epochs
Begin
　While epoch < epochs do
　　While $k < K$ do
　　If 80% of a parameter in Group K is 0
　　The sparsity factor of this group is
　　$\lambda_k = \lambda \times 0.01$
　　End
　　End
　End
End

5　实　　验

5.1　数据集

采用遥感图像数据集RSOD进行训练和测试。RSOD数据集由武汉大学于2015年发布，用于遥感图像目标检测。数据集包含从谷歌Earth和Tianditu下载的976张图像，图像的空间分辨率在0.3 m～3 m之间，包含飞机（aircraft）、油桶（oiltank）、立交桥（overpass）和操场（playground）4类目标。图5为样例图片[11]。在RSOD数据集中，aircraft和oiltank类别中基本为小目标，并且分布集中，overpass类别中包含少量的小目标，playground类别中目标分布较为分散，总之数据集中大部分目标为小目标（目标的尺寸是原图的十分之一）。

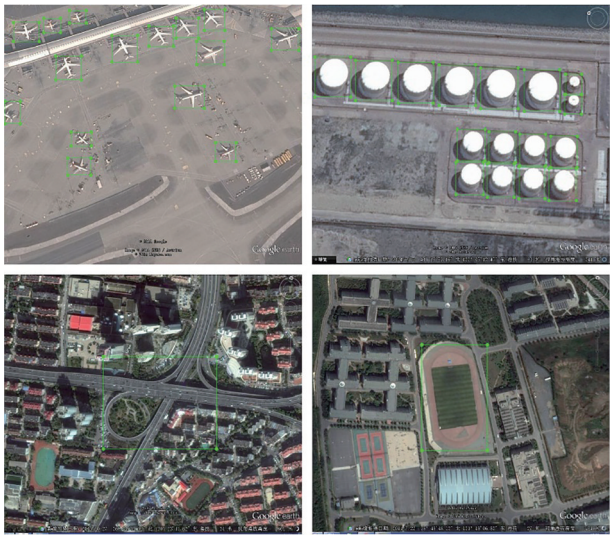

图5　RSOD数据集样例

Fig. 5　Sample images of RSOD dataset

5.2　稀疏训练

图6为YOLOv3-CS模型在数据集RSOD上的γ参数分布图，其中横轴（value）为γ参数的值，纵轴（Layer）为backbone的层序号。从图6可以看出，每一层的γ参数基本在1附近且分布不同。

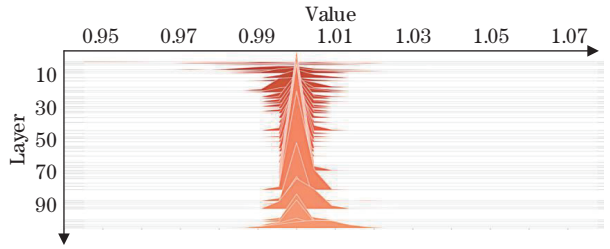

图6　YOLOv3-CS在数据集RSOD上的γ参数分布图

Fig. 6　γ parameter distribution map of YOLOv3-CS on RSOD dataset

根据式(5)对基础训练得到的最优模型进行稀疏训练，模型的稀疏力度主要由稀疏因子λ控制。文献[24]采用全局恒定稀疏因子λ的方式进行稀疏训练，图7为训练后，mAP的变化曲线和模型的稀疏效果图。从图中可以看出：当$\lambda=0.001$时，mAP几乎没有下降，但是γ参数被控制在0.85左右，仍然难以分离出不重要的γ参数，模型的稀疏效果不佳；当$\lambda=0.007$时，模型获得较好的稀疏效果，但是mAP下降严重，在后期难以恢复。因此较大的λ能够很快将模型中的γ参数控制在0附近，但是会造成严重的精度损失，并且难以恢复；较小的λ往往得不到较好的稀疏效果。所以采用全局恒定稀疏因子的方式进行稀疏训练很难确定合适的稀疏因子使模型在mAP保持不变的情况下有较好的稀疏效果。

本实验组通过算法1进行稀疏训练，设定初始稀疏因子$\lambda=0.006$，使得模型达到了较高稀疏率的同时保持了原有模型的mAP。图8为稀疏训练后的mAP变化曲线和稀疏效果图。从图中可以看出，稀疏训练使得80%左右的γ参数接近0，稀疏后模型精度为0.906，相比基础训练的YOLOv3-CS，mAP反而提高了0.003，这是由于RSOD数据集的训练样本较少，模型参数较多，使得原始模型训练产生过拟合的现象，而稀疏训练起到了Dropout的作用。

5.3　剪枝

5.3.1　通道剪枝

本小节在稀疏训练的基础上对模型进行通道剪枝实验。对所提通道剪枝方法、文献[25]和文献[24]中的通道剪枝方法进行了对比，图9为随着通道剪枝率的增加，剪枝模型mAP的变化曲线。从图中可以看出：当剪枝率不超过60%时，与稀疏训练后没有进行剪枝的模型相比，所提通道剪枝方法、文献[24]和文献[25]通道剪枝方法得到的模型mAP基本没有损失；当剪枝率大于60%时，文献[24]和文献[25]的通道剪枝方法得到的模型mAP有严重衰减；而所提剪枝方法在剪枝率大于85%时，mAP才开始衰减，当剪枝率大于95%时，mAP衰减至0。当剪枝率在85%和95%之间时，所提

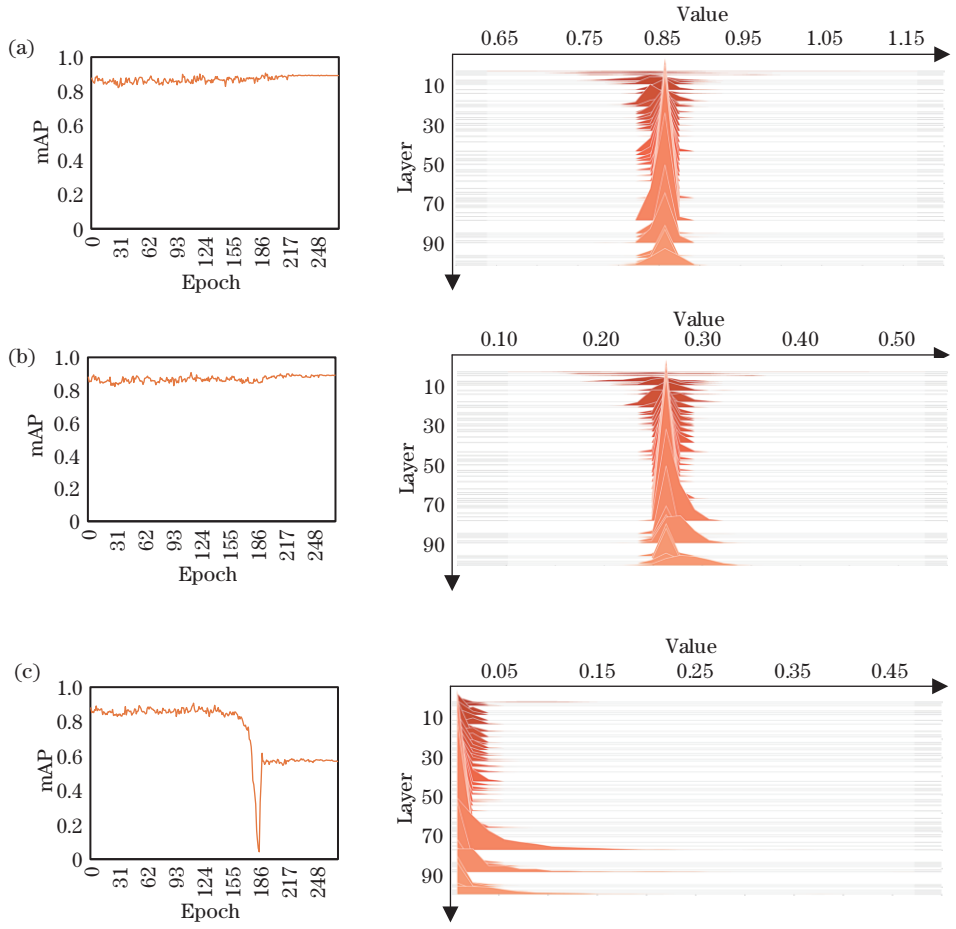

图 7　恒定 λ 稀疏训练后 mAP 的变化曲线和稀疏效果图。(a) λ= 0.001; (b) λ= 0.005; (c) λ= 0.007

Fig. 7　Change curve of mAP and sparse rendering after constant γ sparse training. (a) λ= 0.001; (b) λ= 0.005; (c) λ= 0.007

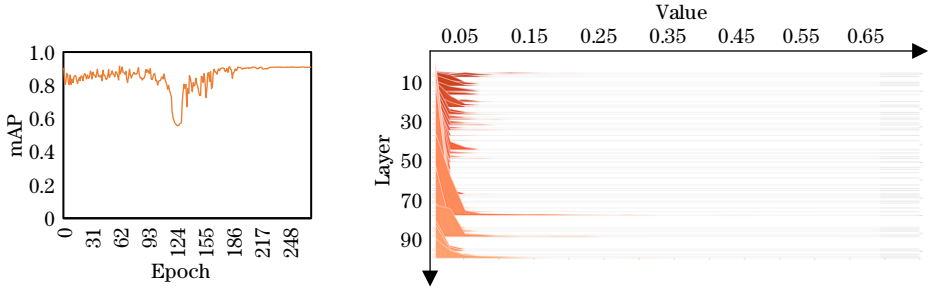

图 8　算法 1 稀疏训练后的 mAP 变化曲线和稀疏效果图

Fig. 8　Change curve of mAP and sparse rendering after algorithm 1 sparse training

剪枝方法得到模型的 mAP 大于文献[24]和文献[25]的通道剪枝方法得到模型的 mAP。因此所提剪枝方法优于文献[24]和文献[25]的通道剪枝方法。

图 10 为随着通道剪枝率的增加，模型 FPS 的变化曲线。从图中可以看出：当剪枝率小于 90% 时，由所提通道剪枝方法得到的模型的 FPS 优于文献[24]的通道剪枝方法得到模型的 FPS，而文献[25]的通道剪枝方法得到模型的 FPS 优于所提通道剪枝方法得到模型的 FPS；当剪枝率大于 90% 时，所提剪枝方法得到模型的 FPS 优于文献[24]和文献[25]剪枝方法得到模型的 FPS。即所提通道剪枝方法得到剪枝模型的

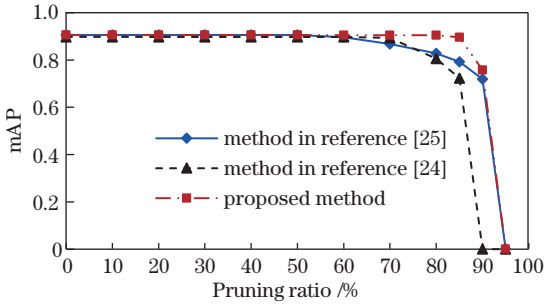

图 9　通道剪枝后模型 mAP 的变化曲线图

Fig. 9　Change curve of model mAP after channel pruning

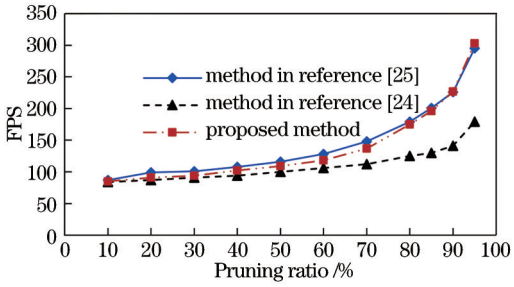

图 10　通道剪枝后模型 FPS 变化曲线图

Fig. 10　Change curve of model FPS after channel pruning

FPS 总是大于文献[24]通道剪枝方法得到的模型 FPS，这是由于所提通道剪枝方法移除的通道数总是大于等于文献[24]剪枝方法移除的通道数。稀疏训练后每一层 γ 参数的分布不尽相同(图 8)，有些层的 γ 整体较小，有些层的 γ 整体较大，文献[24]的通道剪枝方法采用全局阈值进行剪枝，当剪枝率大于 60% 时，会出现网络中某一层的所有 γ 参数都小于剪枝阈值的情况，为了不使该层的通道数变为 0，会保留 γ 最大的通道，因此实际的通道剪枝率总是小于等于预设剪枝率；而所提通道剪枝方法为每一层单独计算剪枝阈值，所以通道剪枝率等于预设剪枝率。

图 11 为模型尺寸的变化曲线，随着剪枝率的增加，模型大小变化趋势与 FPS 的变化趋势相同，因此模型的存储空间与参数量息息相关。当剪枝率小于 80% 时，所提通道剪枝方法优于文献[24]的通道剪枝方法，文献[25]的通道剪枝方法优于所提通道剪枝方法；当剪枝率大于 80% 时，所提通道剪枝方法与文献[25]的通道剪枝方法得到的模型有相同的大小。

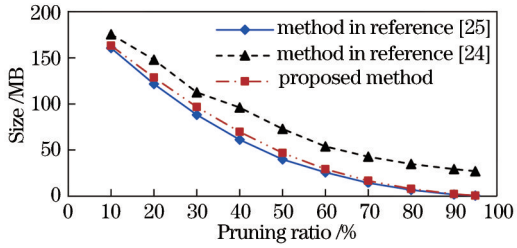

图 11　通道剪枝后模型尺寸变化曲线图

Fig. 11　Change curve of model Size after channel pruning

通过分析剪枝模型的 mAP、FPS 和模型大小的变化，最后确定通道剪枝率为 85%，剪枝结果如表 1 所示。剪枝后模型的 mAP 降低了 0.11%，参数量降低 94.77%，模型尺寸降低 94.99%，FPS 增加 88.46%，浮点运算量(GFLOPS)降低 78.7%，F1 分数提高 0.92%。对剪枝后的模型进行微调后，通道剪枝后模型的 mAP 基本可以恢复到原始模型的 mAP。

表 1　通道剪枝结果

Table 1　Channel pruning results

Name	Size /MB	Number of parameters /MB	GFLOPS	mAP@0.5	F1	FPS
YOLOv3-CS	215.5	51.2	65.6	0.903	0.899	78
YOLOv3-CS (sparse train)	215.5	51.2	65.6	0.906	0.876	78
YOLOv3-CS-CP (pruning rate)	10.8	2.68	14	0.896	0.873	147
YOLOv3-CS-CP (fine-tuned after layer pruned)	10.8	2.68	14	0.902	0.875	147

对网络进行通道剪枝即移除网络中某层的卷积通道，也就是移除该层的输出特征，图 12 为通道剪枝(85% 剪枝率)前后的通道数量对比图。[图 12(a)]为通道剪枝前后 backbone 的通道数对比，深层卷积层(特征图大小为 13×13 和 26×26)移除的通道数大于浅层卷积层(特征图大小为 52×52 和 104×104)移除的通道数；[图 12(b)]为通道剪枝前后 backbone 之外层的通道数对比，同样大小为 13×13 和 26×26 的特征图对应的卷积层剪枝比例较高。因此，在小目标检测中，backbone 中浅层特征比深层特征重要，这与文献[11]的结论一致。

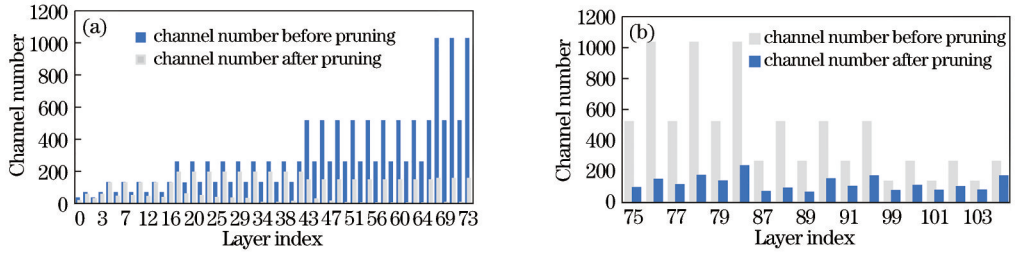

图 12　通道剪枝前后通道数对比。(a) 通道剪枝前后 backbone 的通道数对比；(b) 通道剪枝前后 backbone 之外层的通道数对比

Fig. 12　Comparison of channel numbers before and after channel pruning. (a) Channel number comparison of backbone before and after channel pruning; (b) channel number comparison before and after channel pruning besides backbone

5.3.2　层剪枝

层剪枝同样在稀疏训练的基础上进行。图 13 为随着移除 Residual block 个数的增加，剪枝模型的 mAP 和 FPS 的变化曲线。从图中可以看出：当移除 Residual block 个数不大于 6 时，剪枝模型能保持原来模型的 mAP；当移除 Residual block 个数大于 6 时，mAP 有 24% 的损失，微调后，mAP 可以恢复；当移除 Residual block 个数大于 14 时，层剪枝模型的 mAP 损失严重，微调后也不能恢复到原始模型的 mAP。因此，为了保证模型精度，层剪枝移除的 Residual block 个数不能大于 14。

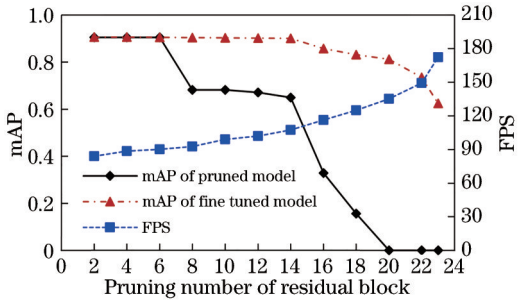

图 13　层剪枝结果图

Fig. 13　Layer pruning result graph

层剪枝与通道剪枝相比，FPS 增益较低，整个 YOLOv3-CS 有 23 个可剪枝 Residual block，也就是 69 个卷积层，将其全部移除，FPS 才能达到 172，通道剪枝率为 80% 时，FPS 达到了 175。

所提层剪枝最终确定移除 14 个 Residual block，表 2 为层剪枝结果。当移除网络中 14 个 Residual block 后，相比原始模型，模型尺寸减小了 46.31%，参数量降低 43.54%，FPS 增加了 49.88%，mAP 损失了 28.13%，通过重训练微调后，模型的 mAP 可以恢复到 0.902，损失基本可以忽略。

表 2　层剪枝结果

Table. 2　Layer pruning results

Name	Size /MB	Number of parameters /MB	GFLOPS	mAP@0.5	F1	FPS
YOLOv3-CS-LP (layer pruned)	115.7	28.91	40.7	0.649	0.575	108
YOLOv3-CS-LP (fine-tuned after layer pruned)	115.7	28.91	40.7	0.902	0.855	108

5.3.3　混合剪枝

通道剪枝和层剪枝实验分别分析了模型精度随着剪枝率的变化，得到了最优的剪枝率。本小节根据最优通道剪枝率和最优层剪枝率，采用先通道剪枝、再层剪枝的方式对 YOLOv3-CS 进行混合剪枝，混合剪枝得到的剪枝模型命名为 YOLOv3-CSP。在混合剪枝中，设定通道剪枝率为 85%，层剪枝移除 Residual block 个数为 14。表 3 中比较了所提剪枝方法、文献[25]和文献[24]剪枝方法对 YOLOv3-CS 的剪枝结果。从表中可以看出，所提剪枝方法对 YOLOv3-CS 剪枝后得到的 YOLOv3-CSP 模型与未剪枝前的模型相比，mAP 损失 0.22%，F1 系数增加 1.11%，而模型大小降低了 95.92%，参数量降低 94.51%，GFLOPS 降低了 82.32%，FPS 增加 173%。YOLOv3-CSP 在模型大小、模型精度和 FPS 上均优于文献[25]和文献[24]的剪枝方法得到的模型。由于 YOLOv3-CS 在 YOLOv3 的基础上增加了两个 104×104 特征对应的 Residual block，而通道剪枝对较浅卷积层的通道剪枝较多，对较深卷积层的通道剪枝较少，所以计算量相对较高。

表 3　所提剪枝方法、文献[25]和文献[24]剪枝方法对 YOLOv3-CS 的剪枝比较

Table 3　Comparison of proposed pruning method, literature [25], and literature [24] pruning methods on YOLOv3-CS

Name	Size /MB	Number of parameters /MB	GFLOPS	mAP@0.5	F1	FPS
YOLOv3-CS	215.5	51.23	65.6	0.903	0.899	78
YOLOv3-CS (70% pruning rate)[25]	12.7	2.97	8.7	0.896	0.875	111
YOLOv3-CS (90% pruning rate)[24]	28.3	6.67	8.2	0.899	0.885	122
YOLOv3-CSP	8.8	2.18	11.6	0.901	0.889	213

表 4 比较了所提剪枝方法（通道剪枝率为 85%，Residual block 剪枝个数为 14）、文献[25]和文献[24]剪枝方法对 YOLOv3 的剪枝结果。从表中可以看出，由所提剪枝方法得到的模型不论是模型大小、检测精

表 4　所提剪枝方法、文献[25]和文献[24]剪枝方法对 YOLOv3 的剪枝比较

Table 4　Comparison of proposed pruning method, literature [25], and literature [24] pruning methods on YOLOv3

Name	Size /MB	Number of parameters /MB	GFLOPS	mAP@0.5	F1	FPS
YOLOv3	246.5	61.63	65.7	0.848	0.825	85
YOLOv3 (70% pruning rate)[25]	14.5	3.58	8.8	0.834	0.789	121
YOLOv3 (90% pruning rate)[24]	31.2	8.01	8.3	0.842	0.819	141
YOLOv3 (our)	10.1	2.62	11.6	0.846	0.824	224

度，还是FPS，均优于文献[25]和文献[24]方法剪枝得到的模型。在mAP损失0.24%的情况下，所提剪枝方法可以对YOLOv3的模型大小实现95.90%压缩，且FPS提高了169.51%。

表5比较了一些轻量化的YOLO模型的性能。与采用轻量化backbone（MobileNetV3[26]和GhostNet[27]）的YOLOv3以及SlimYOLOv3-SPP3-90[24]相比，YOLOv3-CSP不论是模型大小、模型精度，还是FPS，都有较大优势，因此可以在更多的应用场景中使用。与YOLOv3-tiny和YOLOv4-tiny相比，YOLOv3-CSP的FPS虽然较低，但是mAP比YOLOv4-tiny高3.68%，模型大小比YOLOv3-tiny低73.57%；由于GFLOPS比YOLOv3-tiny高一半以上，导致YOLOv3-CSP的FPS低于YOLOv3-tiny。与YOLO-fastest相比，虽然模型大小和FPS没有优势，但是YOLOv3-CSP的mAP高31.15%；与YOLO-fastest-xl相比，YOLOv3-CSP的检测速度高4.41%，mAP高25.14%。总之，YOLOv3-CSP在FPS和检测精度上做到了较好的平衡，可以应用于检测精度和FPS要求较高的场景中。

表5 轻量化模型比较

Table 5 Lightweight model comparison

Name	Size /MB	Number of parameters /MB	GFLOPS	mAP@0.5	F1	FPS
MobileNet-YOLOv3	95.6	23.82	14.5	0.867	0.836	116
GhostNet-YOLOv3	94.2	23.51	14.1	0.861	0.828	133
SlimYOLOv3-SPP3-90[24]	32.4	8.02	9.97	0.892	0.871	133
YOLOv3-tiny	33.3	8.67	5.5	0.843	0.858	322
YOLOv4-tiny	24.4	6.07	13.2	0.869	0.861	243
YOLO-fastest	1.4	0.29	0.8	0.687	0.660	313
YOLO-fastest-xl	3.6	0.84	2.3	0.720	0.637	204
YOLOv3-CSP	8.8	2.18	11.6	0.901	0.889	213

6 结 论

对面向遥感图像小目标检测的YOLOv3-CS进行通道剪枝和层剪枝，实现了模型的压缩。在遥感图像数据集RSOD上的实验结果表明，所提YOLOv3-CSP目标检测模型的性能优于YOLOv3和YOLO-fastest，YOLOv3-CSP在检测精度和FPS上取得了较好的平衡。但是YOLOv3-CSP的浮点运算量仍然较高，这也是推理速度不及YOLOv3-Tiny的主要原因。YOLOv3-tiny和YOLOv4-tiny是针对通用数据集进行设计的通用模型，而YOLOv3-CSP是通过剪枝的方式，在RSOD数据集上得到的在RSOD上有效的模型，其结构有很大的差异，因此导致计算量的差异。进一步的研究可以采用YOLOv4-Tiny的设计方法对剪枝后的模型进行重新设计。对于剪枝后模型精度有损失的问题，可以采用知识蒸馏的方法进一步提高剪枝模型的精度。

参 考 文 献

[1] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//2014 IEEE Conference on Computer Vision and Pattern Recognition, June 23-28, 2014, Columbus, OH, USA. New York: IEEE Press, 2014: 580-587.

[2] He K M, Zhang X Y, Ren S Q, 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.

[3] Girshick R. Fast R-CNN[C]//2015 IEEE International Conference on Computer Vision, December 7-13, 2015, Santiago, Chile. New York: IEEE Press, 2015: 1440-1448.

[4] Ren S Q, He K M, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137-1149.

[5] Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection[C]//2016 IEEE Conference on Computer Vision and Pattern Recognition, June 27-30, 2016, Las Vegas, NV, USA. New York: IEEE Press, 2016: 779-788.

[6] Redmon J, Farhadi A. YOLO9000: better, faster, stronger[C]//2017 IEEE Conference on Computer Vision and Pattern Recognition, July 21-26, 2017, Honolulu, HI, USA. New York: IEEE Press, 2017: 6517-6525.

[7] Redmon J, Farhadi A. YOLOv3: an incremental improvement[EB/OL]. (2018-04-08)[2021-05-06]. https://arxiv.org/abs/1804.02767.

[8] Bochkovskiy A, Wang C Y, Liao H Y M. YOLOv4: optimal speed and accuracy of object detection[EB/OL]. (2020-04-23)[2021-02-03].

[9] Liu W, Anguelov D, Erhan D, et al. SSD: single shot MultiBox detector[M]//Leibe B, Matas J, Sebe N, et al. Computer vision-ECCV 2016. Lecture notes in computer science. Cham: Springer, 2016, 9905: 21-37.

[10] Lin T Y, Goyal P, Girshick R, et al. Focal loss for dense object detection[C]//2017 IEEE International Conference on Computer Vision, October 22-29, 2017, Venice, Italy. New York: IEEE Press, 2017: 2999-3007.

[11] Wang J J, Wei J, Mei S H, et al. An improved YOLOv3 for small object detection in remote sensing images[J]. Computer Engineering and Applications, 2021, 57(20): 133-141.

[12] LeCun Y, Denker J S, Solla S A. Optimal brain damage [J]. Advances in Neural Information Processing Systems, 1989, 2: 598-605.

[13] Li H, Kadav A, Durdanovic I, et al. Pruning filters for efficient ConvNets[EB/OL]. (2016-08-31) [2021-06-04]. https://arxiv.org/abs/1608.08710.

[14] Gong Y C, Liu L, Yang M, et al. Compressing deep convolutional networks using vector quantization[EB/OL]. (2014-12-18) [2021-03-06]. https://arxiv. org/abs/1412.6115.

[15] Dettmers T. 8-bit approximations for parallelism in deep learning[EB/OL]. (2015-11-14) [2021-03-04]. https://arxiv.org/abs/1511.04561.

[16] Lebedev V, Ganin Y, Rakhuba M, et al. Speeding-up convolutional neural networks using fine-tuned CP-decomposition[EB/OL]. (2014-12-19)[2021-03-04]. https://arxiv.org/abs/1412.6553.

[17] Oseledets I V. Tensor-train decomposition[J]. SIAM Journal on Scientific Computing, 2011, 33(5): 2295-2317.

[18] Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network[EB/OL]. (2015-03-09)[2021-03-05]. https://arxiv.org/abs/1503.02531.

[19] Zagoruyko S, Komodakis N. Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer[EB/OL]. (2016-12-12)[2021-04-02]. https://arxiv.org/abs/1612.03928.

[20] Molchanov P, Mallya A, Tyree S, et al. Importance estimation for neural network pruning[C]//2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 15-20, 2019, Long Beach, CA, USA. New York: IEEE Press, 2019: 11256-11264.

[21] He Y H, Zhang X Y, Sun J. Channel pruning for accelerating very deep neural networks[C]//2017 IEEE International Conference on Computer Vision, October 22-29, 2017, Venice, Italy. New York: IEEE Press, 2017: 1398-1406.

[22] Lee N, Ajanthan T, Torr P H S. SNIP: single-shot network pruning based on connection sensitivity[EB/OL]. (2018-10-04) [2021-06-04]. https://arxiv. org/abs/1810.02340.

[23] Liu Z, Li J G, Shen Z Q, et al. Learning efficient convolutional networks through network slimming[C]// 2017 IEEE International Conference on Computer Vision, October 22-29, 2017, Venice, Italy. New York: IEEE Press, 2017: 2755-2763.

[24] Zhang P Y, Zhong Y X, Li X Q. SlimYOLOv3: narrower, faster and better for real-time UAV applications [C]//2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), October 27-28, 2019, Seoul, Korea (South). New York: IEEE Press, 2019: 37-45.

[25] He Y, Liu P, Wang Z W, et al. Filter pruning via geometric Median for deep convolutional neural networks acceleration[C]//2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 15-20, 2019, Long Beach, CA, USA. New York: IEEE Press, 2019: 4335-4344.

[26] Howard A, Sandler M, Chen B, et al. Searching for MobileNetV3[C]//2019 IEEE/CVF International Conference on Computer Vision (ICCV), October 27-November 2, 2019, Seoul, Korea (South). New York: IEEE Press, 2019: 1314-1324.

[27] Han K, Wang Y H, Tian Q, et al. GhostNet: more features from cheap operations[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 13-19, 2020, Seattle, WA, USA. New York: IEEE Press, 2020: 1577-1586.