文章编号：2096-1618(2020)05-0531-006

一种改进的 YOLOv3-Tiny 目标检测算法

杨 铭，文 斌
(成都信息工程大学通信工程学院，四川 成都 610225)

摘要：YOLOv3-Tiny 作为 YOLOv3 目标检测算法的简化版本，拥有检测速度快、体积小、易于在边缘设备上部署等优点，同时也存在着识别精度低，定位不准的问题。由此在该算法的基础上进行改进，首先，对网络结构进行改进，在保证实时性的同时设计一个新的主干网络，提高网络的特征提取能力；其次改进目标损失函数和特征融合的策略，使用 IOU 损失函数代替原先边框位置损失函数，提高定位精度。实验结果表明，改进后的 YOLOv3-Tiny 算法，在保证实时性的情况下明显优于原算法。

关 键 词：深度学习；目标检测；YOLOv3-Tiny；IOU
中图分类号：TP301.6 **文献标志码**：A
doi：10.16836/j.cnki.jcuit.2020.05.009

0 引言

目标检测作为计算机视觉的一个重要的研究方向，其主要目的是对目标图片中可变数量的目标进行分类和定位。目前，仍有许多有待解决的问题，其中包括被检测目标的种类与数量的不平衡；目标尺度大小不一；光照、遮挡、噪声等外在环境的干扰。目标检测算法主要分为传统的目标检测算法和基于深度学习的目标检测算法。

传统的目标检测算法：人脸检测 Viola-Jones 算法[1]、基于梯度直方图(histograms of oriented gradients)[2]和支持向量机(support vector machine)[3]相结合的目标检测算法、DMP(deformable parts model)[4]算法。这些传统的目标检测算法都遵循了手动设计特征，并结合了滑动窗口的形式来对目标进行检测和定位。然而，手动设计的特征无法应对多变复杂的场景，利用滑窗的形式提取目标框进行判断，需要耗费大量的时间。导致传统的目标检测算法泛化能力弱，实时性差。

基于深度学习的目标检测算法主要有 Two-stage 算法和 One-stage 算法两个分支。Two-stage 算法主要思想是先通过特征提取网络提取特征，生成一系列的候选区域(region proposal)，然后在候选区域分类和回归，其中具有代表性的算法有 R-CNN、Fast R-CNN、Faster R-CNN 等。One-stage 算法主要思想是通过一个 CNN(convolutional neural networks)网络直接对目标进行分类和回归，其中代表性的算法有 YOLO[5]、SSD[6]等。相比于传统的目标检测算法，基于深度学习的目标检测算法使用卷积神经网络提取目标特征替代传统算法手动设计特征的形式，具有更好的泛化能力，同时算法的实时性得到了很大的提升。相比于 Two-stage 算法，One-stage 算法实时性高，精度相对较低。但随着近年来不断发展，One-stage 算法在兼顾实时性的同时，检测精度有了很大的提升，更容易满足工业应用的需求，是当前研究的热点。

在实际应用中由于模型体积过于庞大，而且需要高性能的 GPU 支持才能达到目标的实时检测，这就导致了目标检测算法在一些边缘设备部署困难，YOLOv3-Tiny 算法是 YOLOv3[7]算法的简化实现，由于其体积小、实时性高、容易部署，因此非常适合在一些边缘设备、性能较弱的嵌入式平台上使用。但同时存在检测精度较低，定位不准等问题。为此，文中对其进行改进，使其能在保证实时性的前提下，提高目标的检测精度。

1 传统的 YOLOv3-Tiny 算法

1.1 网络结构

YOLOv3-Tiny 网络总共有 24 个网络层，包括 13 个卷积层、6 个池化层、2 个路由层(route)、1 个上采样层和 2 个输出层，使用了特征金字塔网络[8](feature pyramid networks)关联了 2 个不同尺度的特征信息。主干网络采用 7 个卷积层与 6 个池化层构成。主干网络的卷积层由一个 3×3 大小的二维卷积层、BN 层(batch normalization layer)和带泄露线性整流函数(leaky rectified linear unit)依次串联构成。池化层主要用于调整特征图的大小，网络中前 5 个池化层步长为 2，每进行一次池化，特征图大小变为原来的 1/2，最后一个池化层步长为 1，池化后特征图大小不变。若输入网络图片大小为 416×416，网络经过 5 次下采样，输出的特征图大小为 13×13。路由层主要用于改变网

收稿日期：2020-06-27

络输出的路径和关联不同特征层的信息,形成特征金字塔结构。在网络中输入一张 416×416 图片经过主干网络和 3 个卷积层后,输出为网络第一个 13×13 大小的输出层。该输出层经历的网络层数较深,拥有较大的感受野,适合对大目标检测。而后通过路由层、卷积层和上采样层操作,将 13×13 的特征层与第二个尺度 26×26 的特征层进行融合。卷积操作后,输出为网络第二个 26×26 大小的输出层。该输出层经历的网络层数较浅,拥有较小的感受野,适合对小目标检测。将两个输出层的信息进行合并,形成最终的网络输出层。网络结构如图 1 所示。

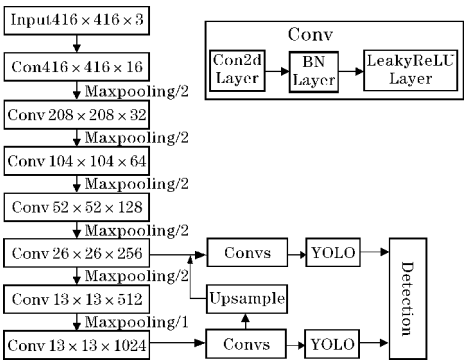


图 1 YOLOv3-Tiny 网络结构

1.2 目标损失函数

在 YOLOv3 算法中,网络将一张图片细分为 $n\times n$ 个网格,每个网格会预测 3 个边界框(bounding box)信息。这些边界框的信息就是网络损失函数所优化的目标。在 YOLOv3-Tiny 中,损失函数主要组成为边界框位置信息损失、边界框置信度得分损失以及相应类别概率得分损失,公式如下:

$$L_{xy}=\lambda_{\text{coord}}\sum_{i=0}^{s\times s}\sum_{j=0}^{m}I_{ij}^{\text{obj}}(2-w_i\times h_i)[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2] \quad (1)$$

$$L_{wh}=\lambda_{\text{coord}}\sum_{i=0}^{s\times s}\sum_{j=0}^{m}I_{ij}^{\text{obj}}(2-w_i\times h_i)[(w_i-\hat{w}_i)^2+(h_i-\hat{h}_i)^2] \quad (2)$$

$$L_{\text{conf}}=-\sum_{i=0}^{s\times s}\sum_{j=0}^{m}I_{ij}^{\text{obj}}[\hat{c}_i\log(c_i)+(1-\hat{c}_i)\log(1-c_i)]-\lambda_{\text{noobj}}\sum_{i=0}^{s\times s}\sum_{j=0}^{m}I_{ij}^{\text{noobj}}[\hat{c}_i\log(c_i)+(1-\hat{c}_i)\log(1-c_i)] \quad (3)$$

$$L_{\text{cls}}=-\sum_{i=0}^{s\times s}I_{ij}^{\text{obj}}\sum_{c\cdot\text{class}}[\hat{p}_i(c)\log(p_i(c))+(1-\hat{p}_i(c))\log(1-p_i(c))] \quad (4)$$

$$Loss=L_{xy}+L_{\text{wh}}+L_{\text{conf}}+L_{\text{cls}} \quad (5)$$

式中 λ 为加权系数,用于平衡各类损失比重。$s\times s$ 为对应特征图的网格数量,m 为每个网格预设的锚边框(anchor)[9]的数量,I_{ij}^{obj} 表示若在第 i 个网格,第 j 个框处有目标该值为 1;反之为 0。I_{ij}^{noobj} 与之相反,表示若在第 i 个网格,第 j 个框处没有目标该值为 1;反之为 0。每个边界框的信息分别为边界框的中心点所在网格相对于该网格左上角的偏移 t_x,t_y,边界框的宽高相对于锚边框的放缩比列 t_w,t_h,衡量当前目标框的置信度得分 P_{obj},以及当前框内各个类别的可能的概率得分 P_n,n 为目标类别的个数。边界框的信息组成如图 2 所示。

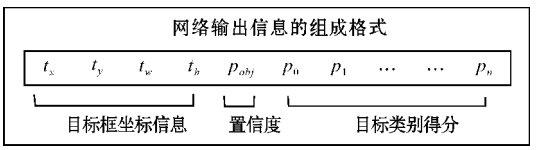


图 2 网络预测边界框信息构成

边界框实际坐标求取公式如下:

$$b_x=\sigma(t_x)+c_x \quad (6)$$

$$b_y=\sigma(t_y)+c_y \quad (7)$$

$$b_w=p_w e^{t}_{w} \quad (8)$$

$$b_h=p_h e^{t}_{h} \quad (9)$$

$$\sigma(z)=\frac{1}{1+e^{-z}} \quad (10)$$

式中:b_x,b_y 为边界框中心点坐标,b_w,b_h 为边界框的宽和高,c_x,c_y 为目标框中心所属网格的左上角坐标,$\sigma(t_x)$,$\sigma(t_y)$ 为中心坐标相对于所属网格左上角的偏移距离,p_w,p_h 为预设锚边框的宽和高。在训练中,在 YOLOv3 算法中会将网格中预设的锚边框与标签中的目标框进行匹配,选取与目标框最大 IOU[10](intersection over union)的框作为正样本。同时,设定一个阈值来过滤 IOU 大于阈值的锚边框,不计入损失函数的计算,低于阈值的框作为负样本计算。

传统的 YOLOv3-Tiny 算法将目标检测看作一个多任务优化的问题,其中目标框的位置信息、置信度、相应的类别概率是优化的对象。从网络输入一张图片,直接输出特征图上每个网格预测的边界框的信息,形成一个端到端的网络。然而,分类与定位的任务相互独立,边界框的位置信息都是独立预测,不能表征整体边框位置信息,导致目标定位不准的问题。在训练阶段,由于阈值的筛选作用,会导致网络可供学习的样本减少,只会保留与标签中最大 IOU 锚边框。如果标签的目标框的大小介于两个预设的锚边框之间,IOU 相差不多,在优化阶段网络可能存在更容易优化 IOU 偏小的框。同时 YOLOv3-Tiny 主干网络较浅,存在特征提取能力较弱,目标感受野不足等问题[11]。针对上述问题,在基于 YOLOv3-Tiny 算法的基础上,从加强对边框位置的学习,调整匹配策略与网络结构这几个方面对 YOLOv3-Tiny 进行改进。

2 改进的 YOLOv3-Tiny 算法

2.1 特征提取模块的改进

在传统 YOLOv3-Tiny 算法中,主干网络采用 3×3 大小的卷积核用于特征提取,每进行一次特征提取就进行

一次池化操作。这样设计虽然网络结构体积小巧、实时性高,但特征能力提取能力弱、信息丢失较大。增加网络的深度与卷积核的数量能提高特征提取能力,但是会带来更多时间上的开销,而且模型的体积也会随之增加。由此,在借鉴了一些轻量化网络架构的设计思想[12],同时兼顾速度与精度的前提下,采用深度可分离卷积模块[13]作为主干网络的主要特征提取模块,该模块能在保证一定精度的条件下可大幅减少网络参数的计算。深度可分离卷积的主要思想分为两步:首先,对输入特征图进行通道分组卷积;然后,对所得结果进行逐点卷积。相对于一个标准的3×3的卷积核模块,采用同样大小的深度可分离卷积核模块参数约为标准卷积的1/3。为此,设计两种不同的模块用于主干网络不同层之间的特征提取,卷积模块如图 3 所示。

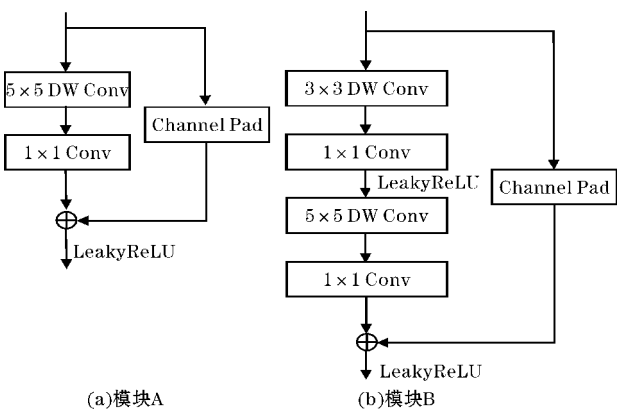


图 3　卷积模块

图 3 中,卷积模块 A 主要采用一个 5×5 大小的深度可分离卷积替代传统 YOLOv3-Tiny 算法普通的 3×3 卷积作为主要的特征提取模块。相比于 3×3 的卷积核,5×5 大小的深度可分离卷积拥有更大的感受野以及更少的参数计算,1×1 的卷积用作特征图通道数调整。同时,为了加强相邻特征层之间的联系,将低层特征进行通道补齐后与高层特征进行叠加,减少信息丢失。卷积模块 B 主要用于高层深度特征提取,采用3×3与5×5不同大小的卷积核进行组合,拥有更强的深层特征提取能力与较少的参数计算。文中以这两种模块为基础构建主干网络。

2.2　新的特征融合方式

在传统的 YOLOv3-Tiny 算法中,使用一个两层的特征金字塔结构融合了两个不同尺度的特征,高层的特征经过卷积与上采样直接与低层的特征在通道上进行堆叠,这样做主要是将低层的特征看作高层信息上采样后的一个残差[14]。不过,在低层的特征仅有较少信息情况下,通过堆叠的方式不足以恢复原有信息。针对这种情况,采用了一种高级语义嵌入(SEB)[15]的方式替代堆叠特征融合方式。在高级语义嵌入中,主要是将高层信息经过卷积和上采样后与低层信息进行逐元素相乘,而不是普通的特征相加或者是特征通道堆叠,其具有更强的特征表达能力。高级语义嵌入模块如图 4 所示。

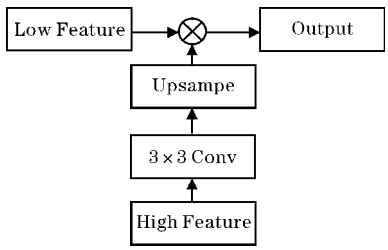


图 4　高级语义嵌入模块(SEB)

2.3　损失函数改进

在 YOLOv3-Tiny 损失函数中,边界框的位置信息是独立预测的。坐标之间没有明确的关联性,不足以代表整个边界框的位置好坏。为了增加位置信息的准确性,可以用 IOU 代替边框位置信息损失函数,IOU 信息代表了边界框与目标框之间的面积的交并比。IOU 值越大,代表与真实目标的面积重叠越多,定位效果更好,能更好地从整体反映边界框的定位精度。IOU 计算公式为

$$\mathrm{IOU}=\frac{S_{\mathrm{A}\cap\mathrm{B}}}{S_{\mathrm{A}\cup\mathrm{B}}} \tag{11}$$

其中 $S_{\mathrm{A}\cap\mathrm{B}}$代表预测的边界框与真实的目标框 B 相交的面积,$S_{\mathrm{A}\cup\mathrm{B}}$表示预测的边框 A 的面积加上目标框 B 的面积减去两个框相交的面积,也称之为并集。可见预测的目标框与真实框越接近,IOU 的值越大,接近于 1;反之,IOU 的值越小,接近于 0。但同时也会存在一些问题,当 IOU 的值为 0 时,表示两个框没有重叠,无法表征两个框的距离。若将 IOU 作为损失函数,网络在进行反向传播的时候,梯度为 0,网络无法被训练;其次,IOU 仅用面积来度量定位的精度,而忽略了两个框之间的对齐方式,也就是说拥有相同 IOU,两个目标框有着不同的位置关系,定位的效果也各不相同。针对上述的问题,选用了 GIOU[16](Generalized Intersection over Union)作为目标损失函数能够更好地优化边框的位置信息。GIOU 的主要思想是在两个框 A、B 中寻找到一个包含 A,B 的封闭区间 C,然后计算 C 里面没有覆盖 A 和 B 的面积与 C 面积的比值,最后用 A 与 B 的 IOU 减去这个比值,将 GIOU 作为损失函数公式为

$$\mathrm{GIOU}=\mathrm{IOU}-\frac{|\mathrm{C}\backslash(\mathrm{A}\cup\mathrm{B})|}{|\mathrm{C}|} \tag{12}$$

$$L_{\mathrm{GIOU}}=1-\mathrm{GIOU} \tag{13}$$

可以看出,IOU 为 0 时,GIOU 不为 0。GIOU 考虑了两个框的位置距离信息使得在两个框没有交叠的情况下也能进行优化,在一定程度上克服了 IOU 的缺点。将 GIOU 损失函数替代原先的边界框位置信息损失函数,最终改进的目标损失函数的公式为

$$\text{Loss}=L_{\text{GLOU}}+L_{\text{conf}}+L_{\text{cls}} \tag{14}$$

在锚边框的匹配中,YOLOv3-Tiny 预设了 3 个锚边框,并设定了一个 IOU 阈值筛选。其中,对于大于阈值而被过滤掉的锚边框也有着与目标框不错的 IOU,相较于 IOU 最大的锚边框可能存在更容易被优化的情况。由此,对锚边框的匹配策略进行调整,不再将大于阈值的锚边框过滤,而是将其按 GIou 值对置信度加权作为正样本进行学习,有着更好的筛选能力。加权后的置信代表了当前边界框的 IOU 大小,随着网络的不断学习,那些容易学习且具有不错的 IOU 的边界框的置信度会不断提升,而那些开始 IOU 值很大,但是网络不容易优化的边界框的置信度会减小,那些开始 IOU 值很大也容易被优化的边界框不会受到影响,在后面阶段通过置信度过滤能得到更好的边界框。策略的调整在一定程度上提升了网络学习的样本数。置信度加权的公式为

$$P_{\text{obj}}=1-\alpha(1-\text{GIOU}_{\text{pos}}) \tag{15}$$

式中,P_{obj}代表了每个正样本边界框的置信度,α 是一个大于等于 0 的加权系数,GIOU_{pos}代表每个正样本的 GIOU 值,GIOU_{pos}越接近 1,P_{obj}越接近 1;GIOU_{pos}越接近 0,P_{obj}越接近 0。

2.4 改进后的 YOLOv3-Tiny

在新的主干网络设计中参照了之前网络设计架构,采用卷积和池化的组合方式,网络层数和卷积核的数量有所增加。新的主干网络总共有 15 层,包括 9 个卷积层和 6 个池化层。第一层网络仍然是普通 3×3 卷积核模块,后面的低层的网络采用卷积模块 A 进行特征提取,高层网络使用卷积模块 A 和卷积模块 B 的嵌套堆叠加强在相应尺度的特征提取能力,特征融合阶段采用高级语义嵌入(SEB)方式进行了特征融合,改进后的网络结构如图 5 所示。

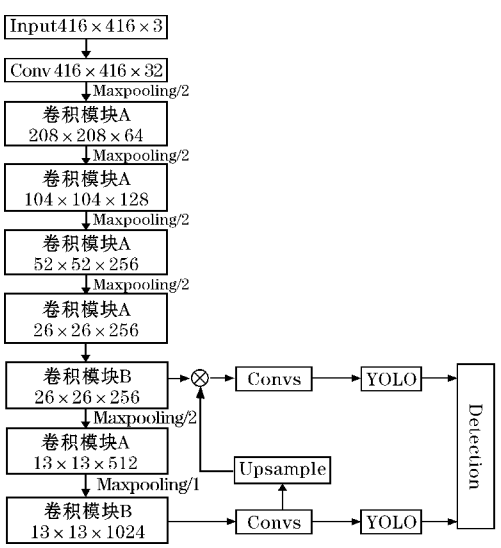


图 5 改进后的 YOLOv3-Tiny 网络结构

3 实验结果分析

3.1 实验环境准备

为验证改进后的算法是否有性能提升,在基于 Linux 操作系统下,用 Pytorch 框架搭建了传统的 YOLOv3-Tiny 算法与改进后的 YOLOv3-Tiny 算法,以进行对比实验。实验平台显卡使用 GTX1080Ti,CPU 型号为 E5-2678 v3,内存为16 GB。数据集使用的是公开的 Pascal VOC2007 和 Pascal VOC2012 数据集。测试集采用 Pascal VOC2007 测试集和 Pascal VOC2012 测试集,其余数据划分为训练集。模型采用均值平均精度与 F1 调和平均数作为评价标准。

3.2 实验过程

文中分别使用了 YOLOv3-Tiny 算法与改进的 YOLOv3-Tiny 算法先后在 VOC 数据集上进行训练,在训练阶段使用了图像增强[17],随机多尺度的策略来扩充数据集,初始学习率设定为0.001,锚边框的筛选阈值设定为0.25,IOU 加权系数为1.2,权重衰减为 0.0005。在训练集上迭代了 285 个 Epochs,训练时长约为12 h。训练结束后对比两个网络的损失值变化情况,结果如图 6 所示。

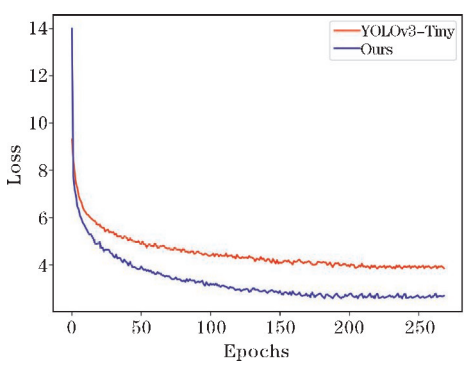


图 6 损失曲线图

由图 6 可知,虽然改进后的的 YOLOv3-Tiny 算法初始误差较大,但是随着迭代次数的增加,误差迅速减少,在网络最终收敛时误差值明显低于未改进的 YOLOv3-Tiny 算法。从损失值可以看出,改进后的模型具有更好的学习能力。

3.3 实验结果与分析

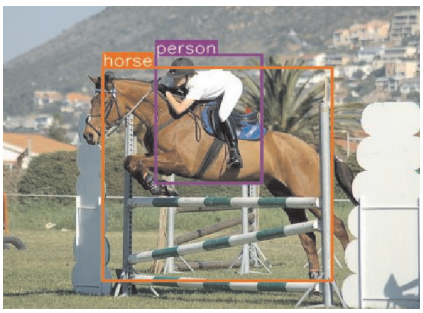
为验证网络的泛化能力,将训练完的最优模型保存,将测试集的图片统一调整到 416×416 大小并对测试结果进行对比分析,在 VOC2007 测试集上进行测试,结果如表 1 所示。

表 1 VOC2007 测试集结果

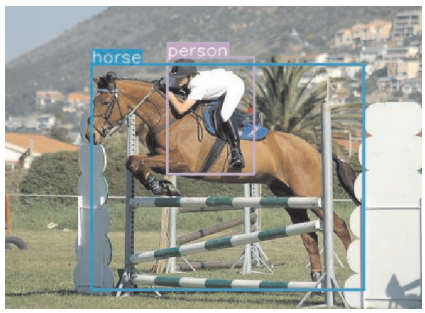
算法	mAP/%	F1/%	模型大小/M
YOLOv3-Tiny	59.7	57.1	33.25
改进的 YOLOv3-Tiny	64.8	62.7	20.8

由表 1 可以知道,改进后的 YOLOv3-Tiny 算法 mAP 为 64.8%,相比于传统的 YOLOv3-Tiny 算法提高了5.1%,F1 提高了5.6%,模型的检测性能得到了极大的提升。同时,改进后的 YOLOv3-Tiny 算法模型大小为20.8 M,比原模型减小了12.45 M,体积更小巧。

在测试集上,实际的检测效果如图 7 所示。



(a) YOLOv3-Tiny 检测结果



(b) 改进的 YOLOv3-Tiny 检测结果

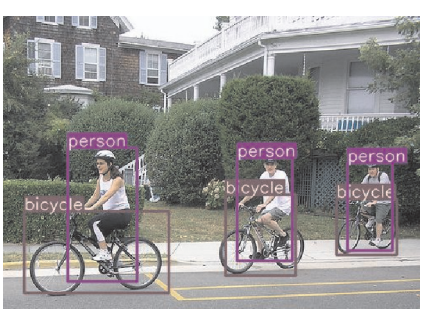
图 7　VOC2007 检测结果对比图

由于 VOC2012 测试集并未公开,只能提交检测结果上传至官方服务器计算,以 mAP 作为评价标准,检测结果如表 2 所示。

表 2　VOC2012 测试集结果

算法	mAP/%	模型大小/M
YOLOv3-Tiny	54.8	33.25
改进的 YOLOv3-Tiny	59.4	20.8

由表 2 可以知,改进后的 YOLOv3-Tiny 算法 mAP 为 59.4%,相比于传统的 YOLOv3-Tiny 算法提高了 4.6%。实际的检测效果如图 8 所示。



(a) YOLOv3-Tiny 检测结果

(b) 改进的 YOLOv3-Tiny 检测结果

图 8　VOC2012 检测结果对比图

网络的实时性也是网络性能评价的重要标准,为检测网络的实时性,将原网络和改进后的网络分别在笔记本电脑上做推理速度的检测。选用的笔记本电脑 CPU 型号为 i5-6300HQ,显卡为 GTX960M,内存为 8GB。在 VOC2007 测试集里选取 400 张图片进行网络推理速度的测试,结果如表 3 所示。

表 3　推理速度结果

算法	平均每帧用时/ms
YOLOv3-Tiny	14.3
改进的 YOLOv3-Tiny	17.5

由表 3 可知,改进后的 YOLOv3-Tiny 算法仍然是一个实时性很强的算法,相比于原算法平均每帧仅多耗时3.2 ms。主要原因是网络有一定程度的加深,新设计的卷积模块通过特征通道填充关联了低层与高层的特征信息,有额外的时间开销。

4　结束语

文中对 YOLOv3-Tiny 算法进行了改进。基于原先的主干网络设计了一个更加轻量、高效的特征提取网络,使用高层语义嵌入代替了原先特征金字塔特征融合方式,减少了特征信息的丢失,调整了原有在训练阶段锚边框的匹配策略增加网络学习样本,使用新的 GIOU 损失函数替代了原先的边界框位置损失函数,加强了网络对边界框位置学习,提高了定位的精度。

在性能对比实验中,改进后的 YOLOv3-Tiny 算法在 VOC2007 测试集中比原算法 mAP 提升5.1%,F1 分数提高5.6%,在 VOC2012 测试集中 mAP 提升4.6%,模型大小减小了12.45 M,检测速度仅有轻微的增加,适用于一些硬件设备要求不高,需要实时检测的应用场景。

参考文献:

[1] Viola P A, Jones M J. Rapid Object Detection using a Boosted Cascade of Simple Features[C]. Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. IEEE, 2001.

[2] Dalal N, Triggs B. Histograms of oriented gradients for human detection[C]. Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. IEEE, 2005:886-893.

[3] Saunders C, Stitson MO, Weston J, et al. Support Vector Machine[J]. Computer Science, 2002, 1(4):1-28.

[4] Pedro F, Felzenszwalb, Ross B, et al. Object detection with discriminatively trained part-based models[J]. IEEE transactions on pattern analysis and machine intelligence, 2010, 32(9):1627-45.

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

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

[7] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[J]. arXiv e-prints, 2018.

[8] Lin T Y, Dollár, Piotr, Girshick R, et al. Feature Pyramid Networks for Object Detection[J]. arXiv e-prints, 2016.

[9] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, 39(6):1137-1149.

[10] Jiang B, Luo R, Mao J, et al. Acquisition of Localization Confidence for Accurate Object Detection[J]. arXiv e-prints, 2018.

[11] 李昕, 赵猛, 董修武, 等. 基于改进 YOLOV3 算法的遥感图像油罐检测[J]. 中国科技论文, 2020(3):267-273.

[12] Bazarevsky V, Kartynnik Y, Vakunov A, et al. BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs[J]. arXiv e-prints, 2019.

[13] Chollet F. Xception: Deep Learning with Depthwise Separable Convolutions[J]. arXiv e-prints, 2016.

[14] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation[J]. arXiv e-prints, 2015.

[15] Zhang Z, Zhang X, Peng C, et al. ExFuse: Enhancing Feature Fusion for Semantic Segmentation[J]. arXiv e-prints, 2018.

[16] Rezatofighi H, Tsoi N, Gwak J Y, et al. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression[C]. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, 2020.

[17] 孙晓凯, 倪卿元, 陈文强. 图像增强方法在深度学习图像识别场景应用中的可行性研究[J]. 电信科学, 2020, 36(S1):172-179.

An Improved YOLOv3-Tiny Target Detection Algorithm

YANG Ming, WEN Bin

(College of Communication Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract: As a simplified version of YOLOv3 target detection algorithm, YOLOv3-Tiny has the advantages of fast detection speed, small size, easy to deploy on edge devices and so on. At the same time, it also has the problems of low recognition accuracy and inaccurate positioning. In this paper, the algorithm is improved. First of all, the network structure is improved. A new backbone network is designed on the premise of instantaneity, which improves the feature extraction ability of the network. Secondly, target loss function and border matching strategy are improved, and the IOU loss function is used to replace the original frame position loss function to improve the positioning accuracy. The experimental results show that the improved YOLOV3-Tiny algorithm is better than the original algorithm when the instantaneity is guaranteed.

Keywords: deep learning; target detection; YOLOv3-Tiny; IOU