doi: 10.7690/bgzdh.2022.03.002

基于嵌入式平台与优化 YOLOv3 的航拍目标检测方法

郭智超，徐君明，刘爱东
（海军航空大学，山东 烟台 264001）

摘要：针对部署在嵌入式平台的目标检测模型在检测航拍目标时存在的检测速率低、耗时高、存储容量低的问题，提出一种基于优化 YOLOv3 算法的航拍目标检测方法。通过模型剪枝极大地减少了模型参数量，使用二分 K-means 对传统的锚框聚类算法进行优化改进，引入 CIOU 损失函数加强边界框回归效果，再经 TensorRT 对模型优化加速后将该检测模型部署到 JetsonTX2 平台上。选取大量不同类别不同环境的航拍图像制作数据集进行实验对比。结果表明：优化后的算法在检验不同航拍图像目标时平均精度可达到 83.9%，对每张图片的检测速度从 2.8 FPS 提升至 14.7 FPS，满足精确性和实时性要求。

关键词：目标检测；YOLOv3 算法；神经网络；深度学习；JetsonTX2 平台
中图分类号：TP301.6　　**文献标志码：**A

Aerial Target Detection Method Based on Embedded Platform and Optimized YOLOv3

Guo Zhichao, Xu Junming, Liu Aidong
(*Naval Aviation University*, *Yantai* 264001, *China*)

Abstract: Aiming at the problems of low detection rate, high time consumption and low storage capacity of target detection model deployed on embedded platform when detecting aerial targets, proposes an aerial target detection method based on optimized YOLOv3 algorithm. The number of model parameters is greatly reduced by model pruning. The traditional anchor box clustering algorithm is optimized and improved by using binary K-means. The CIOU loss function is introduced to enhance the effect of bounding box regression. After the model is optimized and accelerated by TensorRT, the detection model is deployed on JetsonTX2 platform. By selecting a large number of aerial images of different types and different environments to make data sets, the experimental results show that the average accuracy of the optimized algorithm can reach 83. 9% when detecting targets in different aerial images, and the detection speed of each image is improved from 2. 8 FPS to 14. 7 FPS, which meets the requirements of accuracy and real-time.

Keywords: target detection; YOLOv3 algorithm; neural network; deep learning; JetsonTX2 platform

0　引言

由于小型无人机具有成本低、安全风险系数小、机动性强等优点，小型无人机智能化无论对于军事情报侦查、交通疏导还是治安防范等领域都有十分重要的意义，将目标检测技术应用在无人机航拍领域也得到越来越多的重视[1]。

在传统的航拍目标检测方法中，最常用的是通过建立数学模型来实现，包括帧间差分法、背景减法、光流场法等[2]。近些年来随着深度学习技术的不断成熟，研究者越来越多倾向于应用基于深度学习的目标检测算法来增强检测精确性与实时性。基于深度学习的检测方法主要分为 2 类：1) 基于候选区域的方法，Ren S Q 提出的 R-CNN，在此基础上又相继提出了性能更优的 Fast R-CNN 和 Faster R-CNN[3]，微软研究院的何凯明团队分别在 2016 年提出了 R-FCN 模型和 2017 年在 Faster R-CNN 的基础上提出的 Mask R-CNN 模型[4]这些算法虽然在检测精度上达到了较高水平，但在实时性上却无法满足实际需求；2) 基于回归的检测算法，如 YOLO 系列的 YOLO[5]和 YOLOv2 这 2 种算法检测速率极快，但检测精度较差，SSD 算法在检测小目标时存在严重不足；直至 2018 年 Redmon 等[6]提出的 YOLOv3 算法无论在精确性、鲁棒性还是实时性上均达到理想水平。

但在实际航拍应用中，由于无人机上嵌入式设备算力有限、容量较小以及航拍目标多样性与航拍环境复杂性[7]，且传统 YOLOv3 检测模型参数量大，将其部署到无人机上将面临实时性差、加载时间长等问题。一些研究者尝试将轻量级 Tiny-YOLOv3 部署到嵌入式设备上[8-9]，尽管其参数量小，相比原

收稿日期：2021-11-03；**修回日期：**2021-12-28
基金项目：国家自然科学基金项目(51605487)
作者简介：郭智超(1997—)，男，山西人，硕士，从事计算机视觉研究。E-mail: 2868706140@qq.com。

YOLOv3 提升了检测速率，但同时大幅削减了检测精度，无法满足实际需求。严开忠等[10]提出一种将 YOLOv3 骨干网络中的传统卷积全部替换为深度可分离卷积的改进方式，减少了计算量与参数量，满足了实时性要求，检测精度相比于 Tiny-YOLOv3 有所提高，但在面临背景环境复杂的航拍图像时仍无法满足精准度需求。

笔者为了解决无人机在航拍目标检测中所面临的问题，使目标检测技术更好地部署在无人机航拍应用领域，提出一种全新的 YOLOv3 改进模型。其创新点如下：使用综合考虑重叠面积、中心点距离、长宽比的 CIOU 评价指标，减小了模型边界框的损失；通过网络剪枝微调减小模型参数量和模型大小[11]；采用二分 K-means 算法代替 K-means 算法提高聚类效果，降低聚类代价函数。将优化后的模型使用 TensorRT 引擎对模型进行重构与优化，部署至 TX2 平台上，由最终测试结果可知，该优化模型满足航拍目标检测的实际需求。

1　YOLOv3 目标检测模型

1.1　YOLOv3 网络结构

YOLOv3 算法采用全卷积神经网络并引入残差网络，其网络深度为 107 层，其中有 75 层为卷积层。如图 1 所示，YOLOv3 以 Darknet-53 为基础网络，主要由卷积层与残差网络组成，卷积层由 1*1 和 3*3 大小的卷积块组成。同时 YOLOv3 在 Darknet-53 的基础上添加了 YOLO 层，用于多尺度预测，其输出 3 个不同的尺度分别为(52, 52)、(26, 26)、(13, 13)，但其通道数均相同。YOLOv3 网络结构中借鉴了 FPN 特征网络的思想，将不同层次的特征图进行了特征融合，使得检测器能更好地检测不同尺度大小的目标。通过添加上采样层和连接层，融合了 3 个不同尺度的特征，在多个尺度的融合特征图上分别独立做检，(13, 13)、(26, 26)、(52, 52)尺度输出分别用于检测大中小尺度目标。

1.2　YOLOv3 检测原理

YOLOv3 采用多尺度特征预测，输出不同尺度的特征图，共聚类出 9 种尺寸的预设边界框，根据尺寸依次分为不同的组，在检测目标时，给一个特定的特征图分配一个组[12]。YOLOv3 在预测图片上采用的是端对端的检测，将整个图片分为 S×S 个区域，如果一个物体的中心落在某个区域上，则对应的网络会对它进行检测。对于每个网络，都有一个预测框，对于一个 416×416 的输入图像，总共有 13×13×3+26×26×3+52×52×3=10 647 个预测，每次预测时有 4 个坐标参数和 1 个置信度参数。每个大小边界框输出特征图的通道数包括位置信息、置信度和类别预测，所以其输出张量的维度为 N×N×3×(5+M)，3 表示每个网格预测的边界框数量；5 表示 4 个坐标信息和 1 个目标性得分；M 为类别数目。

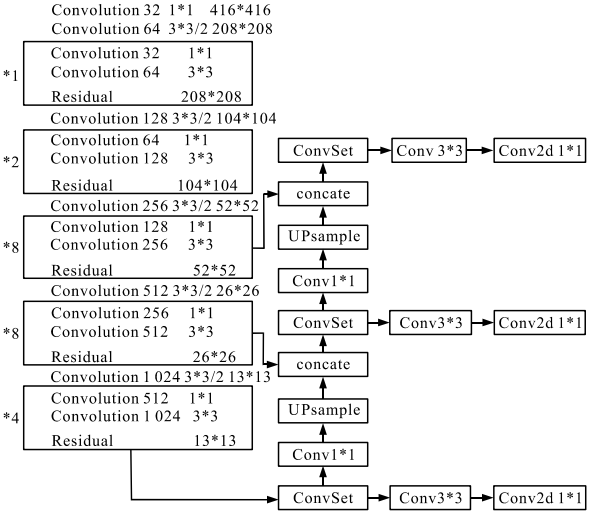


图 1　YOLOv3 网络结构

YOLOv3 通过 NMS 算法去除重复的预测框，通过 logistic 回归对 9 个预设边界框包围的部分进行一个目标性评分，即置信度，通过计算预设框与真实框的交并比(intersection over union，IOU)而得。设定相应的阈值，选取置信度最高的预设框作为最终预测框，将置信度高于阈值而不是最大的预设框忽略处理。置信度计算公式如下：

$$\mathrm{IOU}=\mathrm{anchor}\cap\mathrm{groundtruth}/\mathrm{anchor}\cup\mathrm{groundtruth}\ 。\quad(1)$$

YOLOv3 检测原理如图 2 所示。

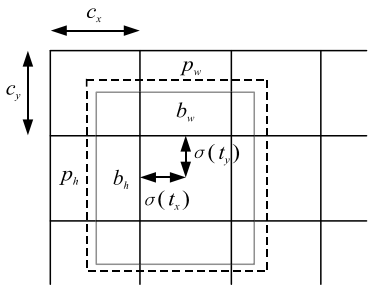


图 2　YOLOv3 检测原理

图中：虚线框表示预设边界框；实线框表示通过网络预测的偏移量计算得到的预测边界框；c_x 和 c_y 分别为中心点所在网格相对于图像左上角偏移量 p_w 和 p_h 为预设边界框在特征图上的宽和高；(t_x, t_y, t_w, t_h)分别为网络预测的边界框中心偏移量(t_x, t_y)和

宽高缩放比(t_w, t_h)；($\sigma(t_x)$, $\sigma(t_y)$)为对应于网格的相对中心点坐标；(b_x, b_y, b_w, b_h)为最终预测的目标边界框的位置信息，其运算公式如下：

$$b_x=\sigma(t_x)+c_x,\ b_y=\sigma(t_y)+c_y,\ b_w=p_we^{t_w},\ b_h=p_he^{t_h}。\quad(2)$$

2 模型优化方案

2.1 网络结构剪枝

原版 YOLOv3 训练所得的模型参数量为 260 M 左右，虽然当前无人机上使用的 JetsonTX2 是一款性能较强的嵌入式开发版，但与 PC 机的 GPU 相比性能上还存在着很大差距。将原版 YOLOv3 部署至 TX2 上执行目标检测时会出现检测速率慢、加载时间长、占用空间过高等问题。为解决以上问题，笔者采用模型剪枝压缩的优化方式减小 YOLOv3 参数量，从而满足其在嵌入式设备 TX2 的实时检测。

模型剪枝分为稀疏化训练、模型剪枝和微调剪枝后的模型 3 步，并循环执行。在 YOLOv3 剪枝的思路中，利用 BN 层缩放因子 γ 作为重要性参数，即 γ 越小，其对应的通道越不重要，可对其进行裁剪操作。剪枝方法如图 3 所示。

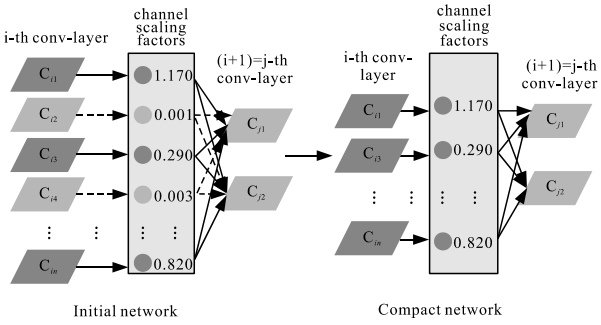


图 3 YOLOv3 模型剪枝

YOLOv3 网络中，每一个卷积层后都添加了 BN 层对通道进行批归一化操作：

$$y_i=\left((x_i-\mu_B)\Big/\sqrt{\sigma_B^2+\varepsilon}\right)\cdot\gamma+\beta；\quad(3)$$

$$\mu_B=\frac{1}{m}\sum_{i=1}^{m}x_i；\quad(4)$$

$$\sigma_B{}^2=\frac{1}{m}\sum_{i=1}^{m}(x_i-\mu_B)^2。\quad(5)$$

式中：γ 和 β 为 BN 层的缩放因子；x_i 和 y_i 为 BN 层的输入与输出。

剪枝过程需要对每一个通道都引入一个缩放因子 γ，然后与通道的输出相乘，接着联合训练网络权重和这些缩放因子。

其中在稀疏化训练中需引入一个关于 γ 的 L_1 正则项，以此来压缩 γ 的值，使其逐渐趋于 0。对于 γ=0 的通道直接裁剪，对于剩下的通道制定相应的裁剪比例，剪去其通道和相应的卷积核，微调剪枝后的网络。该方法的目标函数定义如下：

$$L=\sum_{(x,y)}l(f(x,W),y)+\lambda\sum_{\gamma\in\Gamma}g(\gamma)。\quad(6)$$

2.2 锚框聚类方法优化

在目标检测中，为了让预测框与真实框取得更好的 IOU，更好地对目标进行定位，需要提前标定锚点框对目标进行聚类。传统 YOLOv3 算法使用 K-means 算法对训练集中的检测目标进行聚类，主要通过计算每一个对象与聚类中心欧氏距离来衡量相似度，将对象点归到最相似的类中，接着重新计算每个类的聚类中心，重复过程直至不再变化。由于 YOLOv3 检测 3 个不同尺度目标，需将聚类结果按照大小顺序各自分配到不同的特征图上作为先验框。

聚类结果通过聚类中心和对象之间的聚类距离来评价，聚类距离越小，聚类效果越好。其距离公式如下：

$$d(\text{box}_i,\text{centroid}_j)=1-\left(\frac{\text{box}_i\cap\text{centroid}_j}{\text{box}_i\cup\text{centroid}_j}\right)。\quad(7)$$

式中：box_i 为第 i 个样本的预测框；centroid_j 为第 j 个聚类中心；d 为表示二者之间的距离。

K-means 方法简单且应用广泛，但也存在诸多问题。例如：类中 k 值需要提前给定，而很多情况下 k 值是难以估计的；其次初始聚类中心的选择对聚类结果有着重大的影响，若选择不好可能对最终结果产生重大影响。由于无人机航拍视角的特殊性和复杂性，K-means 方法已无法满足正常需求。笔者将采用二分 K-means 方法进行改进[13]，通过逐步分裂的方式选择初始聚类中心，最大限度降低聚类代价函数，避免了 K-means 局部最优解的问题。具体步骤为：

1) 将所有点作为一个簇，然后将该簇一分为二。

2) 选择满足条件的可以分解的簇，选择误差平方和 SSE 最大的簇进行划分。因为该值越小表示数据点越接近于它们的质心，聚类效果就越好；SSE 越大，表示该簇聚类越不好，越有可能是多个簇被当成一个簇了，所以首先需对这个簇进行划分。SSE 公式如下：

$$\text{SSE}=\sum_{i=1}^{n}w_i(y_i-\hat{y}_i)^2。\quad(8)$$

式中：w_i 为权重值；$\hat{y}_i$ 为该簇所有点的平均值。

3) 使用 K-means 算法将可分裂的簇分为 2 簇。

4) 一直重复步骤 2)和 3)直至簇的总个数达到 K。

针对笔者自建航拍数据集，将二分 K-means 聚类算法中的簇即 k 值设置为 9，实验所得锚框的尺度分别为(18, 33)、(19, 107)、(26, 61)、(39, 24)、(45, 99)、(54, 51)、(75, 94)、(99, 25)、(137, 53)，最终锚框聚类精度可提升至 82.7%。

2.3 边界损失函数优化

在原始的 YOLOv3 中，利用 MSE 作为损失函数来进行目标框的回归，利用 MSE 评价指标有时并不能将不同质量的预测结果区分开来。更重要的是，MSE 损失函数对目标的尺度相当敏感。虽然采用对目标的长宽开根号的方式降低尺度对回归准确性的影响，但还是无法根治。

由于 IOU 计算的是交并比，更能体现回归框的质量，在目标尺度上有较好的鲁棒性，所以常被用来代替 MSE 作为损失函数进行回归[14]。公式如下：

$$L_{\mathrm{IOU}} = 1 - \left|A \cap B\right| \Big/ \left|A \cup B\right| 。 \tag{9}$$

式中：A 为预测框；B 为真实框。

虽然 IOU 相较于 MSE 的结果有所提升，但当检测框与真实框没有重合且梯度为 0 时，无法优化，预测框和真实框无法反映重合度大小；因此，在此基础上不断优化，相继提出了 GIOU 和 DIOU，其公式分别如下：

$$L_{\mathrm{GIOU}} = 1 - \mathrm{IOU} + \left(\boldsymbol{C} - B \cup B^{\mathrm{gt}}\right) \Big/ \left|\boldsymbol{C}\right| ; \tag{10}$$

$$L_{\mathrm{DIOU}} = 1 - \mathrm{IOU} + \rho^2\left(b, b^{\mathrm{gt}}\right) \Big/ c^2 。 \tag{11}$$

式中：$\boldsymbol{C}$ 为预测框和真实框的最小外接矩形；b, b^{gt} 分别为预测框和真实框的中心点；ρ 为计算 2 个中心点间的欧氏距离；c 为能够同时包含预测框和真实框最小闭包区域的对角线距离。

GIOU 缓解了非重叠情况下梯度消失的问题，但由于其严重依赖 IOU，对于水平和垂直方向的盒子收敛速度较慢，导致其非常不稳定。相比之下，DIOU 能够直接最小化预测框和真实框的中心点距离，加速收敛，但没有考虑边界框回归的重要因素纵横比。

CIOU 在 DIOU 的惩罚项的基础上加了一个影响因子 αv，该因子将边界框宽高比的尺度信息考虑了进去[15]。完整的 CIOU 损失函数定义如下：

$$L_{\mathrm{CIOU}} = 1 - \mathrm{IOU} + \rho^2(b, b^{\mathrm{gt}}) \big/ c^2 + \alpha v ; \tag{12}$$

$$\alpha = v \big/ ((1 - \mathrm{IOU}) + v) ; \tag{13}$$

$$v = 4(\arctan(\omega^{\mathrm{gt}} \big/ h^{\mathrm{gt}}) - \arctan(\omega / h))^2 \big/ \pi 。 \tag{14}$$

式中：α 为用作 trade-off 的参数；v 为用来衡量长宽比一致性的参数，主要衡量候选框和目标框的一致性。

为了达到更好的回归效果，本次课题在 YOLOv3 检测模型中使用 CIOU 回归损失函数代替 YOLOv3 的 MSE 损失函数。

2.4 TensorRT 加速

TensorRT 是 NVIDIA 推出的一款基于 CUDA 和 cudnn 的神经网络推断加速引擎，可为深度学习应用提供低延迟、高吞吐率的部署推理，主要对训练好的模型进行加速[16]。相比于一般的深度学习框架，TensorRT 可在 CPU 或者 GPU 模式下提供 10X 乃至 100X 的加速，极大提高了深度学习模型在边缘设备上的推断速度，可用于对超大规模数据中心、嵌入式平台或自动驾驶平台进行推理加速。

在计算资源并不丰富的嵌入式设备上，TensorRT 之所以能加速神经网络的推断主要得益于：TensorRT 支持 INT8 和 FP16 的计算，通过减少计算量和保持精度之间达到一个理想的 trade-off，达到加速推断的目的。网络结构的横向合并与纵向合并。横向合并可以把卷积、偏置和激活层合并成一个 CBR 结构，只占用一个 CUDA 核心，纵向合并可以把结构相同，但权值不同的层合并成一个更宽的层，也只占用一个 CUDA 核心，合并之后模型的层次变得更少，占用的 CUDA 核心数更少；因此，经加速后整个模型结构会更小、更快、更高效。同时，TensorRT 可以通过解析网络模型将网络中无用的输出层消除以减小计算。图 4 为 TensorRT 优化机理。

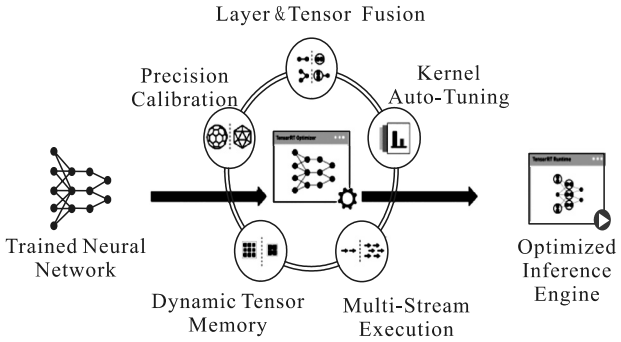


图 4　TensorRT 优化机理

主要对训练好的 YOLOv3 模型的网络结构进行重构与优化，从而达到加速推断的目的。

3 实验设计

3.1 实验运行环境

实验硬件配置为 intel 第九代酷睿 i7-9700 处理

器，32 GB RAM 的服务器，外配有 2080 Ti 显卡，主要用来对优化后的 YOLOv3 模型进行训练。同时，使用 JetsonTX2 作为测试移动端，配有 HMP Dual Denver 2/2 MB L2+Quad ARMA57/2 MB L2 处理器，NVIDIA Pascal™显卡，8 GB RAM，59.7 GB/s 的显存带宽。

该实验软件环境包括 windows10 64 位操作系统、PyCharm 编译软件、python3.6.3 编程语言、pytorch 深度学习框架。同时为了实现算法模型，还需用到 CUDA10.0、cuDNN7.4.1、OpenCV3.0 等软件包。

3.2 实验数据

3.2.1 数据集介绍

该航拍数据集共有 1 200 张，包括训练集、测试集、验证集，按照 4:1:1 比例进行划分数据集从 DOTA－V1.0 的图像挑选而来，包括谷歌卫星图像和部分中国卫星图，包括码头、飞机、网球场 3 类目标。为提高模型的训练效果并加以验证，在训练集中除了正样本外还加入部分负样本，选取多种复杂环境下的图像加以检测。使用 labelimg 工具将挑选的图片手动标注并以 VOC 格式存储，使用 xml 工具存储检测目标的类别信息和位置信息。如图 5 所示。

(a) 飞机 (b) 网球场

(c) 码头

图 5 数据集

3.2.2 数据增强

在神经网络训练的过程中，为了提高样本数量和质量，增强模型的泛化能力和鲁棒性，需要对数据集的样本进行数据增强。本文中数据增强采用如下方法：

1) 尺度变换：图像可以被放大或缩小；

2) 图像剪裁：随机从图像中选择一部分，然后这部分图像裁剪出来；

3) 噪声扰动：对图像的每个像素 RGB 进行随机扰动，常用的噪声模式是椒盐噪声和高斯噪声；

4) 对比度变换：在图像的 HSV 颜色空间，改变饱和度 S 和 V 亮度分量，保持色调 H 不变对每个像素的 S 和 V 分量进行指数运算(指数因子在 0.25 到 4 之间)增加光照变化；

5) mosaic 数据增强：将 4 张图片进行随机裁剪，再拼接到一张图上作为训练数据。不仅丰富了图片的背景，还变相提高了 batch_size。

3.3 评价指标

本网络结构的性能通过 *mAP* 指标、检测速度进行综合评价，其中 *mAP* 指标综合由精度(precision)、召回率(recall)和平均值组成。检测速率由每秒检测的图像帧数(FPS)来表示。

$$\text{precision}=TP/(TP+FP); \tag{15}$$

$$\text{recall}=TP/(TP+FN); \tag{16}$$

$$\text{Accuracy}=(TP+TN)/(TP+FP+TN+FN)。 \tag{17}$$

式中：*TP* 预测为正，实际为正；*FP* 预测为正，实际为负；*TN* 预测为负，实际为负；*FN* 预测为负，实际为正。

$$mAP=\sum_{k=1}^{N}P(k)\Delta r(k)\Big/m; \tag{18}$$

$$\text{FPS}=\text{frameNum}/\text{elapsedTime}。 \tag{19}$$

式中：frameNum 为检测图片总数；elapsedTime 为检测总时间。

3.4 参数设置

本实验训练模型参数选择如表 1 所示，按照设定的参数值经过大量迭代训练，最终使得检测模型达到较优效果。

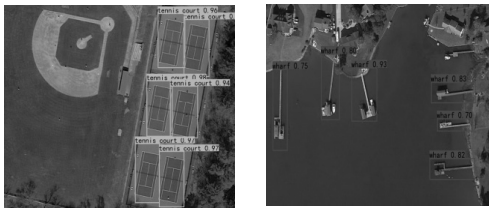
表 1 模型训练参数设置

初始化参数名称	参数值	备注
Batch Size	16	批处理大小
Epoch	2 000	迭代次数
Learning Rate	0.001	学习率
Momentum	0.9	动量值
Weight Delay	0.000 5	权重衰减值
NMS	0.4	非极大抑制阈值
Match Threshold	0.5	置信度阈值

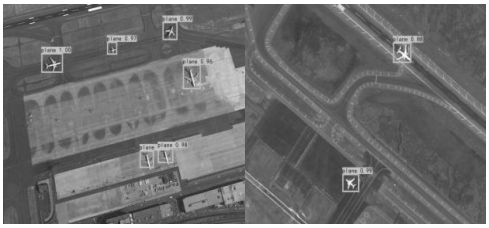
4 实验结果分析

4.1 实验效果

实验效果如图 6 所示。



(a) 网球场检测效果　　(b) 码头检测效果



(c) 飞机检测效果

图 6　实验效果

4.2　实验数据测试对比

在本次实验测试中，分别将模型在 PC 机上训练之后部署至 jetsonTX2 平台上，并进行了 3 组对比实验。分别为：K-means 与二分 K-means 在文中数据集上聚类精度的对比实验；原 YOLOv3 网络与剪枝微调后网络性能的对比实验；原 YOLOv3 检测模型与全面优化 YOLOv3 模型以及常用轻量级 Tiny-YOLOv3 在测试集上检测效果对比实验。

1) 分别使用 K-means 与二分 K-means 在本文中数据集上设置聚类中心数为 3、6、9、12、15、18 的实验，不同聚类中心数下聚类结果如图 7 所示。由实验结果可知，二分 K-means 相比 K-means 在聚类精度上平均提升了 2.3%。

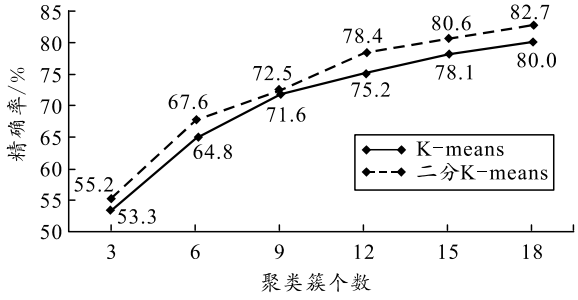


图 7　聚类结果对比

2) 将原 YOLOv3、Tiny-YOLOv3 与剪枝微调的 YOLOv3 经过 PC 机训练后所得模型部署在 JetsonTX2 上进行目标检测，将其性能进行全面对比，对比结果如表 2 所示。

表 2　模型剪枝前后性能对比

模型	参数量/M	模型大小/M	速率/FPS	Map 值/%
YOLOv3	61.5	246.4	2.8	85.4
Tiny-YOLOv3	8.8	33.4	13.1	63.0
剪枝微调 YOLOv3	12.7	48.2	6.3	78.8

由实验结果对比可知：剪枝 YOLOv3 参数量下降为原 YOLOv3 的 20.6%，模型大小下降为 YOLOv3 的 19.6%，Map 可达到 78.8%，检测速率可达 6.3 FPS，剪枝后的模型在精度保持较高的情况下，检测速率有了较大提升。

3) 将原 YOLOv3、Tiny-YOLOv3 与全面优化的 YOLOv3 经 PC 机训练后部署至 JetsonTX2 上进行测试对比，对比结果如表 3 所示。

表 3　模型整体优化前后测试对比

算法	Map/%	AP 值/%			速率/FPS
		飞机	码头	棒球场	
原 YOLOv3	85.4	89.2	84.6	82.4	2.8
Tiny-YOLOv3	63.0	65.0	62.7	61.3	13.1
优化 YOLOv3	83.9	84.7	85.1	81.9	14.7

由实验结果可知：全面改进后的 YOLOv3 检测精度 Map 可达 83.9%，检测速率可达 14.7 FPS，虽然相较于 YOLOv3 的 Map 值有 1.5%的下降，但检测速率达到了 YOLOv3 的 5.25 倍。该检测模型在嵌入式设备上同时满足了实时性和精确性的要求。

5　结论

笔者设计了一种面向航拍目标实时检测应用的神经网络优化方法，解决了原 YOLOv3 检测模型在嵌入式平台上部署时高精度要求与高实时性要求的矛盾。针对本次任务设计了一种改进的 YOLOv3 模型，同时进行了剪枝微调，使其能在较小模型规模与参数量的同时保持高检测精度。为了最大限度地提高运行速率，最后在 JetsonTX2 平台上使用 TensorRT 加速引擎对模型重构优化。从最后的实验结果可知：经过全面优化后的 YOLOv3 模型检测速率从 2.8 FPS 提升至 14.7 FPS，其平均检测精度保持在 81.9 %，基本没有太大影响。和原 YOLOv3 算法及 Tiny-YOLOv3 算法相比，模型无论在精准度、实时性还是运行效率上都达到了预期效果，能在嵌入式平台上对航拍图像进行检测识别的任务，但在模糊图像目标或小图像目标情况下的检测任务中还存在较大提升空间，下一步将尝试使用诸如 MobileNetv3 在内的高性能轻量级网络，同时使用全新的神经网络量化方法，进一步提升嵌入式平台上目标检测模型的效率。

参考文献：

[1] 张宁. 基于无人机平台的动态目标检测系统开发[D]. 杭州: 浙江大学, 2018.

（下转第 20 页）