doi:10.3772/j.issn.1002-0470.2019.10.001

基于 IOU 分析的稀疏视频检测技术研究①

刘 畅②[*,**,***] 王鹏钧[****] 张美玲[**] 田 霖[*,**] 周一青[*,**,***] 石晶林[*,**,***]

([*]移动计算与新型终端北京市重点实验室 北京 100090)

([**]中国科学院计算技术研究所 北京 100190)

([***]中国科学院大学 北京 100049)

([****]96901 部队 北京 100094)

摘 要 基于深度学习的目标检测技术发展迅速,检测性能不断提高。然而,在视频检测应用中,由于视频数据量较大且实时性约束严格,导致目标检测算法的计算资源消耗极高。针对视频检测算法的巨额计算资源消耗问题,本文提出了一种基于深度学习的目标检测算法和目标追踪算法自适应结合的稀疏视频检测方法,能够动态地基于目标区域交并比(IOU)分析,自适应地利用计算资源消耗较小的目标追踪算法替代目标检测算法进行视频分析,从而在保障视频检测准确率的前提下,大幅降低计算资源开销,并进一步提高了视频检测的鲁棒性。

关键词 视频检测;目标追踪;目标区域交并比(IOU)分析;计算资源;处理速度

0 引 言

随着无线通信技术的快速发展[1-3],视频业务的应用领域更加广泛,对智能视频处理技术的需求也日趋提高。目标检测算法是智能视频处理技术的基础,能够不断地在视频帧内和帧间扫描、搜寻运动目标或感兴趣目标,实现自然场景中对目标的定位和识别[4]。在视频检测领域,如何在保证准确率的前提下,降低视频处理中目标检测算法的计算资源开销、提高算法的处理速度一直以来都是研究的热点。

目前,针对视频处理的目标检测方案包括密集检测和稀疏检测 2 种。其中,密集检测是一种系列检测方案,即对视频中的每一帧进行检测,比如基于背景差法、方向梯度直方图(histogram of oriented gradient,HOG) + 支持向量机(support vector machine,SVM)方法[5],以及基于深度学习的快速区域卷积神经网络(fast region with convolutional neural network,fast RCNN)算法[6]、Faster RCNN 算法[7]、特征金字塔网络(feature pyramid networks,FPN)算法[8]、YOLOv3(you only look once)算法[9]等。这类算法需要对视频帧的全部像素进行独立处理,计算量非常大,同时需要很大的资源空间来支持。此外,该类算法只能检测特定的目标。

稀疏检测是一种部分检测方案,即通过检测视频中第 i 帧的目标,利用帧间临时的运动特征来追踪后续的 $i+n$ 帧中的对应目标。这种方式通常考虑目标在前一帧的位置和外观特征,避免目标检测可能会出现的漏检和误检问题。当追踪的置信度较低时,将重新启动检测器检测目标,因此在检测失败的情况下仍有可能追踪到目标。此外,追踪算法只处理估计位置附近的像素,即只做局部的搜索而非全局搜索,只在乎追踪目标的运动特征,不在乎追踪

① 北京市自然科学基金(L172049)资助项目。

② 男,1986 年生,博士生;研究方向:多媒体传输,网络资源分配,计算机视觉;联系人,E-mail:liuchang@ict.ac.cn

(收稿日期:2018-12-25)

的是什么,因此处理速度非常快。常见的追踪算法有基于生成模型的卡尔曼滤波[10]、粒子滤波[11]、核心相关滤波器(kernelized correlation filters, KCF)[12]、基于时空可靠性的相关滤波器(discriminative correlation filter tracker with channel and spatial reliability, CSRT)[13]、高效卷积追踪器(efficient convolution operators, ECO)[14]等方法。

目前稀疏视频检测的研究极少[15],且精度和实时性较低。针对这一问题,本文提出了一种基于目标区域交并比(intersection over union, IOU)分析的稀疏视频检测方法。该方法将检测准确率高但计算资源消耗较高的 YOLOv3[9]检测算法与传统的 CSRT 追踪算法相结合,基于目标区域的 IOU 分析,自适应调整视频帧的检测或追踪处理模式,能够在保障视频检测准确率的前提下,大幅降低计算开销并提高处理速度。

1 基于 IOU 分析的视频稀疏检测算法

在目标检测和目标追踪之间需要一种自适应判决机制来实现检测器和追踪器的交替使用,使得目标检测算法和目标追踪算法能够紧密融合。在视频中,目标区域的面积一般是不会发生陡变的,因此,可以基于检测器和追踪器所得到的目标区域面积进行 IOU 分析[16],从而实现检测算法和追踪算法的自适应调整。

基于目标检测与追踪结合的视频稀疏检测算法架构如图 1 所示。在初始化阶段,通过对同一目标计算检测(detection)和追踪(track)得到的感兴趣区域(region of interest, ROI),并联合分析,得到后续的检测 + 追踪组中的追踪长度 L_t,且初始化阶段的长度设置为 $L_{initial}$。在后续的检测 + 追踪组中,对于

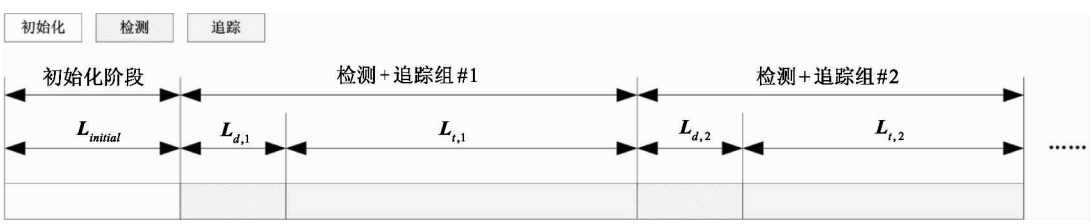


图 1 基于检测 + 追踪的稀疏视频检测架构示意图

第 j 组,将通过自适应组合 $L_{d,j}$ 帧长度的检测和 $L_{t,j}$ 帧长度的追踪操作,通过追踪计算取代检测计算的方式,来避免密集视频检测所造成的的计算资源开销大、处理速度慢等问题,并实现追踪和检测算法相辅相成,提高视频检测准确率。

1.1 基于 IOU 分析的自适应选择机制

在基于 IOU 分析的稀疏视频检测算法中,核心思想是通过计算量更低的追踪操作替代计算消耗较大的检测操作,因此,每一个检测 + 追踪组中的追踪帧长度的确定是影响算法性能的关键。根据上文分析,每一个检测 + 追踪组中的追踪帧长度是根据初始化阶段的训练和学习结果得到的,因此,初始化阶段对检测器和追踪器的评估非常重要。

初始化阶段的处理流程如图 2 所示。首先,定义检测器运行时得到视频帧中目标区域信息 roi_d 为 (x_d, y_d, w_d, h_d),其中 (x_d, y_d) 是检测框的中心点坐标,(w_d, h_d) 是检测框的长和宽。同理,定义追踪器工作时得到的目标信息 roi_t 为 (x_t, y_t, w_t, h_t),其中,(x_t, y_t) 是追踪框的中心点坐标,(w_t, h_t) 是追踪框的长和宽。

在进行视频稀疏检测时,算法中的检测器和追踪器将独立工作并计算出目标信息。其中,检测框的目标信息为

$$roi_d = w_d \times h_d \tag{1}$$

追踪框的目标信息为

$$roi_t = w_t \times h_t \tag{2}$$

因此,对于第 k 帧视频,其交并比信息 IOU_k 为

$$IOU_k = \frac{roi_{d,k} \cap roi_{t,k}}{roi_{d,k} \cup roi_{t,k}} \tag{3}$$

其中,$roi_{d,k}$ 是第 k 帧的检测框面积,$roi_{t,k}$ 是第 k 帧的追踪框面积。在该过程中,会出现检测或追踪在进行独立计算时检测或追踪失败的情况,即当 $IOU_k >$

T 时认为计算成功,记录检测 + 追踪结果,当 $IOU_k < T$ 时认为计算失败,删除当前帧的计算结果。通过大量测试,当 $T = 0.5$ 时,算法能达到较好的自适应选择效果。

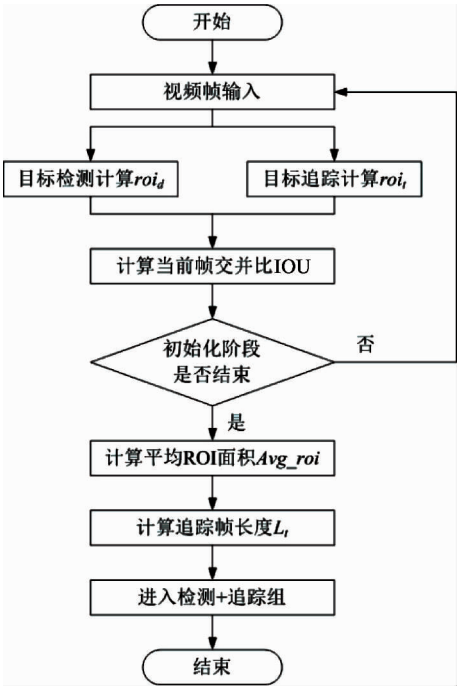


图 2　自适应 IOU 分析流程示意图(初始化阶段)

接下来,算法将在每次检测 + 追踪组结束时更新实际的追踪帧长度信息至列表 N 中,并进一步计算这次追踪中追踪框的平均面积,即:

$$N = [L_{t,1}, L_{t,2}, \cdots], L_t \in R \tag{4}$$

$$Avg\_roi_i = \frac{\sum_{j=1}^{L_{t,i}} (roi_{d,j} \cap roi_{t,j})}{L_{t,i}} \tag{5}$$

N 保存了每次检测器更新时前面追踪了的帧数,$L_{t,i}$ 指第 i 次检测器更新时中间帧被追踪的数量,Avg_roi_i 是第 i 次调用追踪器后追踪框的平均面积。在平均面积达到 Avg_roi_i 后隔 $L_{t,i}$ 帧交替检测追踪。根据多次测试,本文将(Avg_roi_i, $L_{t,i}$)通过 K-means 聚类方式聚类[17],区分为不同的类型,从而根据后续的实际 IOU 分析结果,确定其归属,并进一步计算得到 $L_{t,i}$ 的取值。此外,在实际操作时,$L_{initial}$ 的长度一般为预设值,且长度越长,分析结果越精确,本文后续的实验结果按照 $L_{initial} = 200$ 进行计算。$L_{d,i}$ 的取值一般情况下等于 1。

1.2　检测器的选取

在计算机视觉领域,目标检测的任务主要是找到视频图像中的感兴趣目标,并解决目标在哪里和目标是什么(即目标的定位和分类)2 个问题。目标检测是计算机视觉中最基础的部分,也是最重要的部分,目标追踪、目标识别、目标分割等应用都是建立在目标检测的基础上。目标检测算法主要包括基于传统学习的目标检测算法和基于深度学习的目标检测算法 2 类。接下来,将分别对这 2 类检测器进行分析。

首先,基于传统学习的目标检测算法基本处理流程如图 3 所示。该类算法通常采用滑动窗口和图像缩放等操作来进行区域选择并提取人工设计的特征,利用提取到的特征训练分类器得到分类模型,其特征训练的时间从几秒到几个小时不等,测试时间随着测试数据量的增加而增加,成本代价较高。同时,所提取的特征都是低层次的,对目标具有针对性。此外,这类算法其目标检测和目标分类是分别解决的,速度较慢,难以实现实时处理。

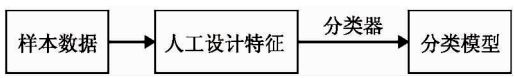


图 3　传统学习目标检测架构示意图

HOG + SVM 是非常具有代表性的基于传统学习的目标检测算法。图 4 是采用该算法进行行人检测的结果示例。从图中可以看到,该方法的行人检测有很多漏检的情况,同时检测的结果精确度较低。

(a) 过人行道的行人　　(b) 某大厦前行人

图 4　采用 HOG + SVM 进行行人检测结果示例

与此同时,基于深度学习的目标检测算法则采用了神经网络自动学习得到特征,这些特征具有较强的可扩展性和鲁棒性,如图 5 所示,通过对特征的训练学习能够定位到目标并检测出其类别。

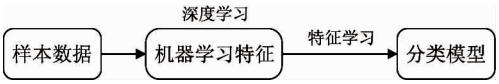


图 5　深度学习目标检测架构示意图

YOLOv3[9]是非常具有代表性的深度学习算法，尤其在处理速度上具有显著优势，比 R-CNN 快 1 000倍、比 Fast R-CNN 快 100 倍，优化在 GPU 上的 DPM 算法都无法和 YOLOv3 相比。与此同时，YOLOv3 算法准确率更高，是当前性能最佳的实时检测算法之一，能实现实时地检测从而达到追踪的目的。

如图 6 所示，YOLOv3 是将图像分成 $S \times S$ 个网格，各个网格只对中心落在网格内部的目标进行分析，每个网格需要预测 3 个尺度下的 9 个 bounding box 和类别信息，一次性预测所有区域所含目标的 bounding box、目标置信度以及类别概率。该算法借鉴 FPN 特征金字塔，利用多尺度检测来检测小目标，同时借鉴了残差网络结构，使得精度和速度都达到空前的高、快。然而，基于深度学习的目标检测算法的训练需要大量的数据来支撑，同时由于卷积网络的深度，数据的训练会需要很长的时间，但是测试运行需要很少的时间就可以完成。因此本文采用基于 YOLOv3 的轻量级的网络结构（tiny）进行特征的提取和模型的训练，以提高模型的训练速度和检测速度。

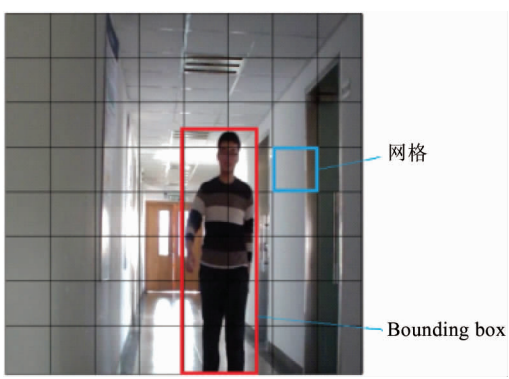
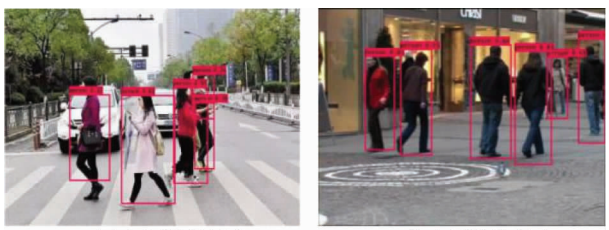


图 6　YOLOv3 网格划分示意

图 7 是采用 YOLOv3-tiny 进行行人检测的结果示例。同图 4 进行比较，可以看到通过基于深度学习的 YOLOv3 目标检测算法检测到的行人明显比通过 HOG + SVM 算法检测到的行人准确率要高，同时可以看到几乎没有漏检的目标。但是，如果在视频序列中每帧都采用训练好的 YOLOv3 模型进行检测，同一个变化很小的目标可能在连续的几帧中被多次检测，计算资源浪费严重。故本文利用视频帧序列之间的关系来进行视频序列的目标检测，即将追踪算法同检测算法融合。因此，本文在目标检测部分采用实时性和准确性相对较高的 YOLOv3 算法。



(a) 过人行道的行人　　(b) 某大厦前的行人

图 7　采用 YOLOv3-tiny 进行行人检测结果示例

1.3　追踪器的选取

目标追踪的核心思想是在初始化视频序列第一帧中目标的位置和大小后，预测出后续视频帧中该目标所在的位置和大小。目标追踪不需要目标识别，可以根据运动特征来进行追踪，而无需确切知道追踪的是什么，可以利用视频画面之间的临时关系来实现目标追踪，因此计算量和所占资源空间较小。

目前，比较常用的目标追踪算法是生成类和判别类 2 大类。生成类方法是在当前帧对目标区域进行建模，在下一帧寻找与上一帧模型最为匹配的区域就是该目标的位置轨迹，如典型的卡尔曼滤波、粒子滤波等。判别类方法是采用机器学习对提取到的目标图像特征训练分类器，下一帧通过分类器找到该目标，如典型的基于相关滤波的 KCF、CSRT 等算法和基于深度学习的 ECO 算法等。本文主要解决的是在保证准确率的前提下，提高视频中目标检测的速度并减少资源开销，降低硬件需求，使其在硬件设备配置较低的情况下也能运行。因此本文对中间帧采用的追踪算法力求速度快、准确率高、资源开销低。

相对于深度学习算法来说，CSRT 追踪器不依赖训练数据和网络模型，而且该追踪器使用了 2 个标准特征（HOG 特征和颜色特征），并在带有通道和空间可靠性的判别滤波器中使用空间可靠性图将滤

波器调整到能够支持从视频帧中选择部分感兴趣区域来进行追踪,使得追踪区域有自适应功能。经过多次对不同追踪器的测试分析,CSRT 追踪器准确率高,资源开销小,能够实时追踪,而且其追踪框能够自适应随目标的大小而改变。因此本文采用 CSRT 追踪器进行辅助视频检测。

2　实验方法及结果分析

实验环境:本文在 Ubuntu 16.04 系统下进行仿真实验,计算机硬件配置为 Intel(R) Xeon(R) CPU E5-1650 v4@3.60 GHz 处理器,内存为 1.5 G,磁盘大小为 183 G,显卡型号是 Quadro K4200。

检测目标:行人是视频序列中最受关注和最有价值的目标之一,而且行人检测在很多领域应用极其广泛,因此,本文将行人作为检测目标进行仿真实验。本文采用的数据集为 VOT2016,在此公开数据集上可以客观有效地评价面向视频的目标检测算法的准确性和实时性。此外本文还采用了实际监控摄像头采集的行人视频,该视频数据采集的是实验室人员在自然场景下活动情况,此数据能够在一定程度上反映真实环境中面向行人视频序列的检测能力。

对比算法:本文选取了基于深度学习的 YOLOv3[9] 目标检测算法和基于 DeepSort[18] + YOLOv3 的目标检测追踪算法进行对比实验,同本文提出的基于深度学习的视频目标检测方法在公开数据集 VOT2016 和实际采集视频中进行性能评估。

评价指标:本文采用查准率 P(precision)和查全率 R(recall)[19] 对所提算法的有效性进行评价。其中查准率反映的是视频帧检测的精度,查全率反映的是视频帧检测对目标信息检测全面程度的评价指标。查准率和查全率的计算公式如下:

$$P = \frac{TP}{TP + FP} \tag{6}$$

$$R = \frac{TP}{TP + FN} \tag{7}$$

其中,TP 是检测到的目标中是行人的数量,FP 是检测到的目标中不是行人的数量,FN 是视频帧中行人未被检测到的数量。因此 P 是指检测到的目标中是行人的概率,R 是指所有的行人中被检测到是行人的概率。

此外,本文综合考虑基于视频的检测算法的查准和查全能力,采用漏检率 M(miss)来评价所提算法:

$$M = \frac{n}{Num} \tag{8}$$

当视频帧中有行人但未检测到或者检测到但 $IOU < 0.5$ 时,视为漏检,n 为漏检数,Num 为被检测的视频帧数,漏检率 M 指在视频帧检测中未检测到行人的视频帧数量占据此次视频帧总数量的百分比。

3 种算法在 VOT2016 数据集和实际环境录制视频的测试结果示例如图 8 所示。在检测性能对比上,可以看出,本文所提算法与 YOLOv3 算法和 YOLOv3 + DeepSort 算法均能正确检测出视频中的行人目标。在计算资源开销上,可以根据示例图片中左上角的视频处理帧率进行对比。由于 3 种算法在相同的计算环境下处理相同的视频资源,因此,处理视频的帧率可以等效地评估出计算资源开销,即帧率越大,计算资源开销越小,反之亦然。从图 8 的实验结果分析,YOLOv3 和 YOLOv3 + DeepSort 算法在处理 VOT2016 数据集的视频时,均保持在 17fps 左右,而本文所提算法在处理该数据集数据时,能将处理帧率提高到 23 ~ 30 fps,即提高了 1.5 倍左右,即计算资源开销降低了 1.5 倍。在实际录制的视频中,YOLOv3 和 YOLOv3 + DeepSort 算法的处理帧率保持在 15 fps,而本文所提算法保持在 30 fps,即计算资源下降了 2 倍。

随后,对不同算法的处理性能进行统计分析。如表 1 所示,在图形处理单元(graphics processing unit, GPU)利用率对比中,可以看出,本文所提算法的 GPU 利用率在不同数据集上均控制在 10% ~ 40% 左右,比其他 2 种算法下降了 2 ~ 5 倍,说明本文算法能够有效地降低 GPU 资源的开销。在 CPU 利用率对比中,本文算法主要和同样采用了跟踪算法的 YOLOv3 + DeepSort 算法进行对比,可以看出,本文算法降低了约 2 倍左右的 CPU 资源开销。在处理帧率对比中,可以看到,YOLOv3 和 YOLOv3 + DeepSort 算法的处理帧率在 VOT2016 数据集和实

际录制视频中均稳定在 13 ~ 16 fps，而本文算法在 VOT2016 数据集中达到了 20 fps，在实际录制视频中达到了 25 fps。综上所述，本文所提算法能够大幅降低计算资源开销。

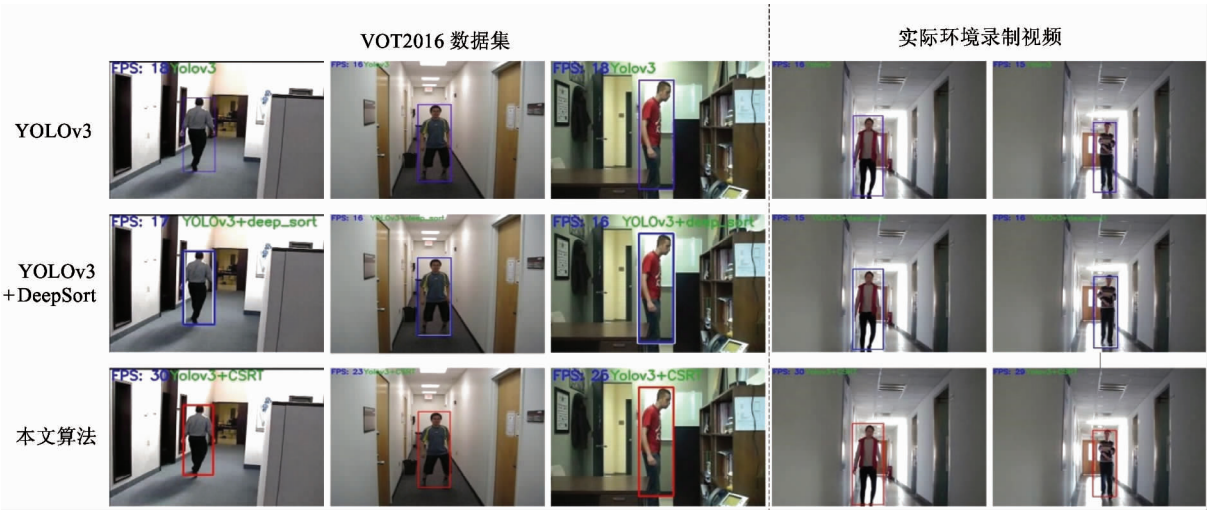


图 8　不同算法的检测结果示例

表 1　不同算法的客观指标评估

数据	实际录制视频			VOT2016 数据集		
算法	本文算法	YOLOv3	YOLOv3 + DeepSort	本文算法	YOLOv3	YOLOv3 + DeepSort
GPU 利用率	10% ~40%	40% ~80%	60% ~80%	10% ~40%	50% ~80%	50% ~70%
CPU 利用率	230% ~350%	120% ~140%	300% ~600%	200% ~300%	120% ~150%	350% ~650%
漏检率	1.51%	14.04%	26.81%	2.10%	2.90%	5.50%
处理帧率	25 fps	16 fps	13 fps	20 fps	15 fps	14 fps
查准率	100%	100%	100%	100%	100%	100%
查全率	98.39%	85.56%	73.15%	97.10%	96.90%	94.30%

在漏检率对比中，本文算法在 VOT2016 数据集中的漏检率最低，为 2.1%，而 YOLOv3 和 YOLOv3 + DeepSort 分别为 2.9% 和 5.5%。在实际录制视频中，本文算法的漏检率达到了 1.51%，而其他 2 种算法分别为 14.04% 和 26.81%。可见，本文算法的漏检率要明显优于其他 2 种算法，尤其在实际视频中更为明显。

在查全率对比中，本文算法在 VOT2016 数据集上达到了 97.1%，高于其他 2 种算法。在实际录制视频中，本文算法的查全率依然优势明显，保持在 98.39%，而其他 2 种算法下降比较严重。

综上所述，在计算资源开销方面，本文算法无论在公开数据集上还是在实际录制视频中，都优于 YOLOv3 和 YOLOv3 + DeepSort 算法，且在实际录制视频中，本文算法的优势尤为明显。在检测性能方面，由于本文算法采取了检测 + 追踪组合的模式，能够起到相辅相成的作用，因此本文算法无论在公开数据集上还是在实际录制视频中，都优于 YOLOv3 和 YOLOv3 + DeepSort 算法，同样，在实际录制视频中，本文算法的优势更为显著，这证明了本文算法不仅有较强的检测准确率，更具备显著的鲁棒性。

3　结　论

本文旨在保障视频检测的准确率、查全率等性能的基础上，研究基于目标检测与目标追踪结合的稀疏视频检测算法，以实现计算资源开销的大幅下降。为实现这一目标，本文在目标检测算法的基础上，以 IOU 分析为依据，自适应激活计算资源开销较低的目标追踪算法，从而替代目标检测算法造成

的巨额计算资源开销。在算法实现中,本文分析并对比了现有的目标检测和目标追踪算法,进而选取了适合于本文算法的 YOLOv3 目标检测算法和 CSRT 目标追踪算法进行实现。最后,本文基于 VOT2016 公开数据集和实际录制视频,以行人检测为目标,对本文算法性能进行了统计评估。实验结果表明,本文算法一方面能够大幅降低计算资源开销,同时又能达到更加优秀的视频检测性能,并显著提高了在实际录制视频检测时的鲁棒性,因此,具备良好的实用价值。

参考文献

[1] Liu L, Zhou Y Q, Garcia V, et al. Load aware joint CoMP clustering and Inter-cell resource scheduling in heterogeneous ultra dense cellular networks[J]. *IEEE Transaction on Vehicular Technology*, 2018, 67(3): 2741-2755

[2] Zhou Y Q, Liu H, Pan Z D, et al. Energy efficient Two-stage cooperative multicast based on device to device transmissions: effect of user density[J]. *IEEE Transaction on Vehicular Technology*, 2016, 65(9): 7297-7307

[3] Liu L, Zhou Y Q, Tian L, et al. CPC-based backward compatible network access for LTE cognitive radio cellular networks[J]. *IEEE Communications Magazine*, 2015, 53(7): 93-99

[4] Liu C, Tian L, Zhou Y Q, et al. Video content redundancy elimination based on the convergence of computing, communication and cache[C]// 2016 IEEE Global Communications Conference, Washington, USA, 2017: 1-6

[5] Xu Y Z, Yu G, Wang Y P, et al. A hybrid vehicle detection method based on Viola-Jones and HOG + SVM from UAV images[J]. *Sensors*, 2016, 16(8): 1325-134

[6] Girshick R. Fast R-CNN[C]// 2015 IEEE International Conference on Computer Vision, Santiago, Chile, 2015: 1440-1448

[7] Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks [C]// Advances in Neural Information Processing Systems, Montreal, Canada, 2015: 91-99

[8] Lin T Y, Dollár P, Girshick R, et al. Feature pyramid networks for object detection[C]// IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, USA, 2017: 2117-2125

[9] Redmon J, Farhadi A. YOLOv3: An Incremental Improvement[R]. New York: Cornell University, 2018

[10] 乔少杰, 韩楠, 朱新文, 等. 基于卡尔曼滤波的动态轨迹预测算法[J]. 电子学报, 2018, 46(2): 418-423

[11] 常天莉, 黄浩晖, 陈玮. 基于粒子滤波的机器人主动定位算法[J]. 计算机工程与设计, 2018, 39(2): 570-573

[12] Wu W, Wang D, Luo X, et al. An improved KCF tracking algorithm based on multi-feature and multi-scale[C]// 10th International Symposium on Multispectral Image Processing and Pattern Recognition, Xiangyang, China, 2018: 1-6

[13] Mechelmans D J, Strelchuk D, Doñamayor N, et al. Reward sensitivity and waiting impulsivity: shift towards reward valuation away from action control[J]. *International Journal of Neuropsychopharmacology*, 2017, 20(12): 971-978

[14] Danelljan M, Bhat G, Khan F S, et al. ECO: efficient convolution operators for tracking[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, USA, 2017: 6638-6646

[15] Wojke N, Bewley A, Paulus D. Simple online and real-time tracking with a deep association metric[C]// IEEE International Conference on Image Processing, Beijing, China, 2017: 3645-3649

[16] Rahman M A, Wang Y. Optimizing intersection-over-union in deep neural networks for image segmentation[C]// International Symposium on Visual Computing, Las Vegas, USA, 2016: 234-244

[17] Moriya T, Roth H R, Nakamura S, et al. Unsupervised pathology image segmentation using representation learning with spherical K-means[C]// Medical Imaging 2018: Digital Pathology. International Society for Optics and Photonics, Houston, USA, 2018: 1-7

[18] Veeramani B, Raymond J W, Chanda P. DeepSort: deep convolutional networks for sorting haploid maize seeds [J]. *BMC Bioinformatics*, 2018, 19(9): 85-93

[19] Morstatter F, Wu L, Nazer T H, et al. A new approach to bot detection: Striking the balance between precision and recall[C]// 2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, San Francisco, USA, 2016: 533-540

A novel sparse video detection method based on IOU estimation

Liu Chang[*,**,***], Wang Pengjun[****], Zhang Meiling[**], Tian Lin[*,**], Zhou Yiqing[*,**,***], Shi Jinglin[*,**,***]

([*] Beijing Key Laboratory of Mobile Computing and Pervasive Device, Beijing 100190)

([**] Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([***] University of Chinese Academy of Sciences, Beijing 100049)

([****] Unit 96901, Beijing 100094)

Abstract

Recently, the performance of object detection algorithms based on deep learning is continually improved. However, in video detection applications, the computational resource consumption of object detection algorithms is more and more huge, which is caused by the processing speed constraint and data size of video. To address this, a sparse video detection method based on intersection over union (IOU) estimation is proposed. In the proposed method, object tracking algorithm with much lower computational resource consumption is adaptively activated by IOU estimation to replace object detection algorithm. Experimental result shows that the proposed method not only greatly reduces the overall computational resource consumption, but also improves robustness for video detection.

Key words: video detection, object tracking, intersection over union (IOU) estimation, computational resource, processing speed