计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2021,30(12):128−138 [doi: 10.15888/j.cnki.csa.008214]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

# 基于注意力机制和卡尔曼滤波的多目标跟踪①

秦泽宇, 黄　进, 杨　旭, 郑思宇, 付国栋

(西南交通大学 电气工程学院, 成都 611756)
通讯作者: 黄　进, E-mail: jhuang@swjtu.edu.cn

**摘　要**: 为了解决目前多目标跟踪算法在行人遮挡后无法再次准确跟踪的问题, 提出了一种融入注意力机制和改进卡尔曼滤波的多目标跟踪算法, 选用联合检测和重识别框架提取特征, 同时完成目标检测和重识别任务. 设计了并行支路注意力机制, 包括空间注意力和通道注意力两个部分, 每个部分都采用并行支路的方式完成池化和卷积操作. 在跟踪阶段, 本文提出了速度先验卡尔曼滤波, 实现对行人运动状态更精确的预测. 采用 CUHK-SYSU 数据集对算法进行训练, 并在 MOT16 数据集上做算法的验证和测试. 本算法的多目标跟踪准确度 (MOTA) 达到了 65.1%, 多目标跟踪精确度 (MOTP) 达到了 78.8%, 识别 *F*1 值 (*IDF*1) 达到 62.3%. 实验表明, 提出的跟踪算法可以有效地提高跟踪的整体性能, 实现对目标的持续跟踪.

**关键词**: 多目标跟踪; 卡尔曼滤波; 特征融合; 注意力机制; 目标遮挡

引用格式: 秦泽宇,黄进,杨旭,郑思宇,付国栋.基于注意力机制和卡尔曼滤波的多目标跟踪.计算机系统应用,2021,30(12):128–138. http://www.c-s-a.org.cn/1003-3254/8214.html

## Multi-Target Tracking Using Attention Mechanism and Kalman Filter

QIN Ze-Yu, HUANG Jin, YANG Xu, ZHENG Si-Yu, FU Guo-Dong

(School of Electrical Engineering, Southwest Jiaotong University, Chengdu 611756, China)

**Abstract**: Given that the existing multi-target tracking algorithm cannot track accurately after occlusion, a multi-target tracking algorithm using the improved attention mechanism and Kalman filter is proposed. The structure of joint detection and embedding is used to extract features and accomplish object detection and identification simultaneously. A parallel-structured attention mechanism is proposed, containing both spatial and channel parts. Each part is designed into parallel branches for pooling and convolution. During tracking, the proposed velocity-prediction Kalman filter is adopted for the more accurate prediction of pedestrian movements. The CUHK-SYSU dataset is used for training, and the algorithm is verified and tested on the MOT16 dataset. The proposed algorithm can achieve 65.1% MOTA, 78.8% MOTP, and 62.3% *IDF*1. The experimental results show that the proposed tracking algorithm can improve the overall tracking performance and achieve continuous tracking.

**Key words**: multi-target tracking; Kalman filter; feature fusion; attention mechanism; target occlusion

多目标跟踪融合了图像处理、模式识别、计算机技术等知识, 还涉及到了多个领域的内容, 一直以来是计算机视觉研究领域的热点和难点. 运动目标跟踪已经有很多年的研究和发展历史, 深度学习算法在近几

① 基金项目: 国家自然科学基金 (61733015); 高铁联合基金 (U1934204); 四川省重点研发计划 (2020YFQ0057); 四川省自然资源科研项目 (KYL202106-0099)

Foundation item: National Natural Science Foundation of China (61733015); High-Speed Railway Joint Funds of National Natural Science Foundation of China (U1934204); Key Science and Technology Program of Sichuan Province (2020YFQ0057); Nature Resource Program of Science and Technology, Sichuan Province (KYL202106-0099)

收稿时间: 2021-03-02; 修改时间: 2021-03-29; 采用时间: 2021-04-06

年的突破性进展, 以及相比传统方法表现出了极大的优势, 使得多目标跟踪成为深度学习一个重要的研究领域. 计算机视觉在市场的需求不断增加, 多目标跟踪技术被广泛应用于诸多场景中, 例如行人跟踪, 交通管理以及自动驾驶等领域[1]. 但是多目标跟踪在复杂环境下仍然面临诸多的挑战, 包括目标之间的遮挡, 目标轨迹的绘制与预测, 不同目标之间存在一定的相似性, 以及背景带来的干扰.

多目标跟踪算法的研究根据初始化方式不同, 主要分为两种: (1) 基于检测的跟踪; (2) 无检测的跟踪. 基于检测的跟踪通过检测算法将视频中的目标检测出来, 然后利用跟踪器将检测目标与轨迹相关联. 无检测的跟踪需要在第一帧中手动初始化跟踪对象, 然后再做跟踪. 基于检测的跟踪是目前的主流方式, 其原因在于无需更多交互操作, 当新目标出现时会被检测器自动检测出来. 在基于检测的跟踪方式中, Milan 等[2] 提出连续能量函数最小化方法, 通过设计能量函数, 寻求函数的最优解, 从而提高跟踪效果, 有效地解决多目标跟踪过程中的遮挡问题. Song 等[3] 使用混合高斯概率假设密度滤波器, 设计了基于检测的多目标跟踪算法. 该算法采用分层跟踪框架, 在处理遮挡和漏检情况具有较好的效果.

深度学习的发展使得越来越多的检测算法运用到多目标跟踪中, 并将离线训练与在线跟踪相结合. Leal-Taixé等[4] 提出了一种针对行人跟踪处理数据关联的方法. 该方法采用两阶段方案去匹配检测对. 第 1 阶段对 Siamese 网络进行训练, 学习两个输入图像之间的时空结构, 聚合像素值和光流信息. 第 2 阶段通过梯度提升器实现目标预测, 生成目标匹配概率. 该方法在行人跟踪中具有领先的优势. Wojke 等[5] 首先利用检测器对目标进行检测, 然后设计了行人外观特征 (appearance feature) 提取网络, 解决了因遮挡导致运动信息没用时错误分配目标身份的问题. Yu 等[6] 等也提出了基于检测的跟踪算法, 在行人检测器和外观特征提取两处均使用了基于深度学习的方法, 通过在每帧上使用检测器检测行人位置, 利用行人检测框的外观特征进行前后帧行人框的匹配, 从而实现对行人的跟踪. 上述方法将检测和重识别分为两个独立的模型, 但是重识别模型在推理阶段耗时较大, 最终导致跟踪的实时性较差.

Wang 等[7] 将上述方法称为检测与重识别分离算法 (Separate Detection and Embedding, SDE), 也就是两步法. Zhang 等[8] 和 Wang 等[7] 采用了称为联合检测和重识别 (Joint Detection and Embedding, JDE) 框架, 从而将检测算法和重识别相结合. 其基本思想是使用单个网络提取特征, 同时用于完成目标检测和重识别的任务, 输出检测框并提取检测框相应的外观特征, 通过共享特征的方式有效地减少计算时间, 达到实时性. Wang 等利用 YOLO[9] 作为检测器, 而 Zhang 等认为, 基于锚框的检测不适合去学习重识别信息, 无锚检测才能更好的提取重识别特征, 从而选用了 CenterNet[10] 作为检测器. Xiao 等[11] 将行人检测和行人重识别结合起来研究, 提出了一种 OIM 损失 (Online Instance Matching) 来训练网络, 且只需要使用单个卷积神经网络来进行训练, 真正意义上实现了端到端训练.

针对多目标跟踪遮挡和漏检的问题, 本文对文献 [8] 中联合检测和重识别框架进行了改进, 选择 ResNet50[12] 作为骨干网络, 设计了并行支路注意力将其融入到骨干网络 ResNet50 中以提升网络的性能, 并设计了速度先验卡尔曼滤波, 实现了根据行人的速度动态调整卡尔曼方程. 在公共数据集的实验结果表明, 本文提出的跟踪算法能有效地提升算法的跟踪能力, 使算法能长时间跟踪目标.

## 1 多目标跟踪算法框架

多目标跟踪的任务是给定视频序列帧, 对感兴趣的目标的位置、运动状态等信息进行分析, 从而获得目标的轨迹进行持续的跟踪. 基于检测的跟踪是最常用的方法, 即先利用检测器检测目标, 然后将检测目标与上一帧的跟踪目标进行关联, 从而形成跟踪轨迹. 但是基于检测的跟踪往往存在以下两个问题. 第一, 跟踪效果非常依赖目标检测器的性能. 第二, 跟踪方式只能针对特定的目标类型, 如: 行人、车辆.

本文的多目标跟踪算法整体框架如图 1 所示, 主要由联合检测和重识别框架与跟踪器两部分组成. 本文算法的整体流程为: 输入视频序列帧后经过融入并行支路注意力机制的骨干网络提取目标特征, 用于目标检测和重识别任务. 目标检测任务获取目标边框以及中心点坐标, 重识别任务提取不同目标的特征信息. 跟踪器根据目标检测的输出结果进行卡尔曼滤波预测, 并将预测结果与重识别特征融合后进行数据关联, 确定目标编号.

### 1.1 联合检测和重识别框架

如图 2 所示, 联合检测和重识别框架主要包含 3 大部分: 骨干网络特征提取、目标检测和重识别. 输

入图片通过骨干网络提取特征, 并分出 4 个分支, 分别对应为热力图分支, 目标框分支, 中心点偏移分支以及重识别分支. 每个分支由一个 3×3 卷积层后面接一个 1×1 卷积层实现, 除了最后输出通道维度的不同, 其组成都相似.

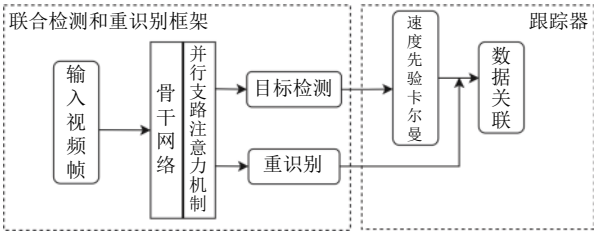


图 1 多目标跟踪算法整体框架

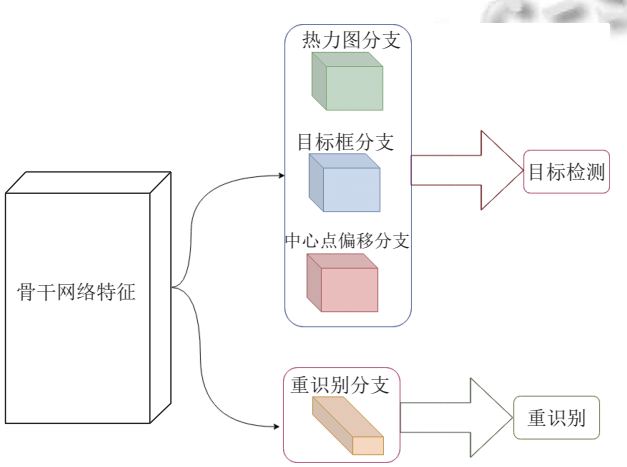


图 2 联合检测和重识别框架

### 1.2 目标检测与重识别

联合检测和重识别框架获得检测框的同时提取检测框内物体的身份信息. 本文选用基于中心点的目标检测算法 CenterNet, 并通过目标中心点位置提取目标特征. 选择 CenterNet 检测算法主要有以下两个原因. 第一, 相比于基于锚框的检测算法, CenterNet 无需处理大量的锚框, 提高检测速度. 第二, CenterNet 可以预测物体的中心点, 便于精确提取目标的特征.

联合检测和重识别框架中的热力图分支, 目标框分支和中心点偏移分支负责完成目标检测部分. 热力图分支估计目标中心的位置, 输出向量维度为 $1\times H\times W$, 目标框分支用来估计目标边界框的高度和宽度, 输出向量维度为 $2\times H\times W$, 中心点偏移分支的作用是更精确的定位目标的中心点位置, 输出维度为 $2\times H\times W$.

CenterNet 中边框的中心点即表示检测的目标, 将中心点的位置直接做回归得到目标的位置. 热力图损失函数为变形的 focal loss, 其公式如下:

$$L_{\text{hm}}=-\frac{1}{N}\sum_{xy}\begin{cases}(1-\hat{M}_{xy})^{\alpha}\ln(\hat{M}_{xy}), & \text{if } M_{xy}=1\\ (1-M_{xy})^{\beta}(\hat{M}_{xy})\ln(1-\hat{M}_{xy}), & \text{otherwise}\end{cases} \tag{1}$$

式中, $\hat{M}_{xy}$为预测的热力图, $M_{xy}$是真实热力图.

重识别分支负责完成重识别部分, 根据目标检测中心点的位置, 提取目标特征, 用以区分不同对象的特征信息, 输出向量维度为 $128\times H\times W$, 每个目标用一个 128 维的向量表示.重识别分支将每个物体视为一个类别, 其损失函数如下:

$$L_{id}=-\sum_{i=1}^{N}\sum_{k=1}^{K}L^{i}(k)\ln(p(k)) \tag{2}$$

式中, $N$ 为图像中的目标数量, $K$ 为类别数量.

### 1.3 骨干网络的设计

在联合检测和重识别框架中, 目标检测和重识别任务共享大部分特征, 但是这两个任务需要从不同层提取特征以便获得更好的结果. 目标检测任务需要网络深层的语义信息来预测目标类别和位置, 而重识别任务需要更多的低层信息来辨别不同的目标. 加入多层特征融合能有效地解决上述两个任务的矛盾.

因此良好的特征提取骨干网络对后续的目标检测与重识别具有非常重要的意义. 本文在设计骨干网络中有以下几点来考虑: (1) 骨干网络的深度选择; (2) 加入多层特征融合; (3) 设计注意力机制.

多目标跟踪问题中的行人由于摄像头的位置、拍摄角度和摄像头移动等问题会发生一定的形变, 在发生遮挡等情况下容易失去原有的跟踪信息. 因此需要选取较深的网络提取更深层次的特征信息. 但是随着网络层数的增加, 训练容易出现饱和现象, 过深的网络会存在梯度消失和梯度爆炸的问题. 综合上述因素, 本文选用残差网络 ResNet50 作为骨干网络做特征提取.

ResNet 网络的主要构成是残差块, 残差块的作用是将浅层提取的特征, 通过跨层连接的方式传递给深层, 使得深层部分获得浅层信息, 起到了特征信息补充的作用. 3 层残差块如图 3 所示.

设残差网络的输入为$x$, 残差块中的各层权重为$W_1,W_2,W_3$. 第 3 层卷积的输出为$F(x,W_1,W_2,W_3)$, 则最后的输出函数为:

$$y(x,W_1,W_2,W_3)=x+F(x,W_1,W_2,W_3) \tag{3}$$

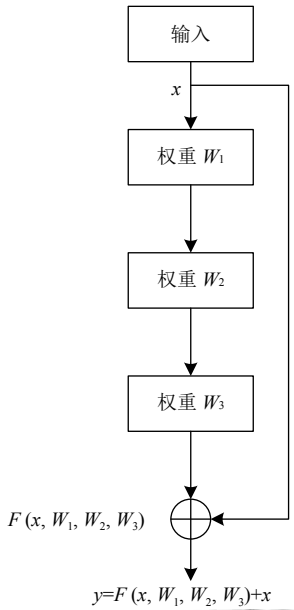


图 3　3 层残差块结构

ResNet50 在 layer1, layer2, layer3, layer4 中使用了残差块, 残差块的数量分别为 3、4、6、3, 加上最前面和最后面的卷积池化层, 总共由 6 个部分构成.

本文在 ResNet50 的 layer4 输出后面加入了特征金字塔网络[13], 如图 4 所示. 本文 ResNet50 中的 layer1, layer2, layer3, layer4 的输出分别是原图的 1/4, 1/8, 1/16, 1/32 倍, 通过上采样后向下传递, 将深层特征中的语义信息传递到低层特征上, 使得低层也拥有深层语义信息, 从而构成了特征金字塔网络结构, 最后将各层的特征融合在一起输出做预测.

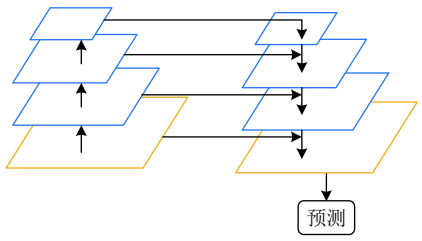


图 4　特征金字塔网络

## 2　并行支路注意力机制

在已有的注意力机制中, 比如 SE (Squeeze-Excitation) 模块[14], 该注意力机制主要学习通道之间的相关性, 并筛选出针对通道的注意力, 提高网络对通道的关注度. CBAM (Convolutional Block Attention Module) 模块[15] 不仅要求注意力告诉我们重点关注哪里, 还要提高关注点的表示, 因此引入空间注意力机制和通道注意力机制, 如图 5 所示. 空间注意力关注图像中物体的位置, 而通道注意力则关注图像中的目标.

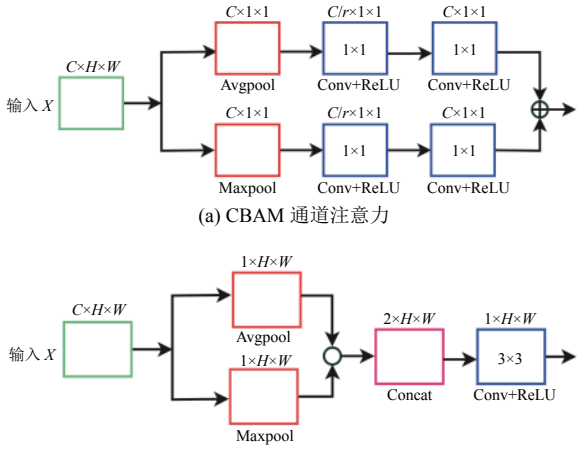


图 5　CBAM 注意力

然而尽管 CBAM 引入了空间注意力机制和通道注意力机制, 但依旧有以下两个问题. (1) CBAM 模块的通道注意力支路上仅是将最大池化与平局池化的结果通过简单的加法相结合, 这种结合方法过于简单, 加法操作没法保留原有池化的结果. (2) 在空间注意力机制中, CBAM 采用的是直接压缩特征图的方式来获得最大池化与平局池化的结果, 这会导致特征图中部分纹理信息的丢失. 针对上述问题, 为了能更好的获取行人目标的特征, 减少冗余信息的干扰, 本文借鉴了 CBAM 模块的思想, 设计了在空间和通道上均采用并行支路的注意力机制, 以获取丰富的特征信息.

本文设计的并行支路注意力机制的具体流程为: 针对特定的输入特征图$F \in R^{C\times H\times W}$, 首先经过通道注意力机制, 得到输出结果$H_c \in R^{C\times 1\times 1}$, 将通道注意力输出结果与输入特征图逐个元素相乘得到加权结果 $F_1$, 其次将 $F_1$ 经过空间注意力机制, 得到输出结果$H_s \in R^{1\times H\times W}$, 将空间注意力输出结果与 $F_1$逐个元素相乘得到最终加权结果 $F_2$.

### 2.1　通道注意力机制

通道注意力机制关注图像中物体“是什么”, 采用池化的方式对空间维度进行压缩, 映射空间信息, 从而降低空间的干扰, 然后对池化结果再进行卷积运算. 与 CBAM 思想不同的是, 本文设计的并行支路通道注意力机制将池化的结果进行拼接, 使其维度变为原来的两倍,

然后经过 1×1 卷积运算后输出. 与简单的相加操作相比, 采用拼接的方式能够保留原来池化结果的输出信息, 如图 6 所示.

并行支路通道注意力机制的具体做法是将输入特征图分成两条并行支路分别进行池化, 然后将池化结果经过两次 1×1 卷积运算, 在通道维度进行压缩, 本文的通道压缩 $r$ 设置为 8. 最后将两条支路的输出结果拼接起来, 经过 1×1 卷积运算, 使通道数减半, 输出 $C$×1×1 维度的特征图.

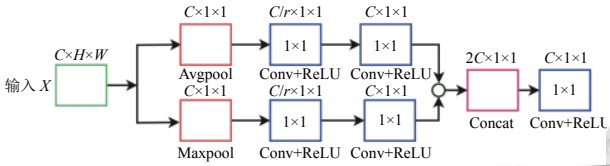


图 6 通道注意力

上述过程的表达式为:

$$F_{a1} = ReLU(Conv(ReLU(Conv(avgout)))) \quad (4)$$

$$F_{a2} = ReLU(Conv(ReLU(Conv(maxout)))) \quad (5)$$

$$F_a = Conv(concat(F_{a1}, F_{a2})) \quad (6)$$

式中, $F_{a1}$和$F_{a2}$是两条并行支路的输出结果, $Conv$ 为 1×1 卷积, $ReLU$ 为卷积后的激活函数, $concat()$ 是拼接操作. $F_a$是最终输出的特征图. 由于通道注意力机制与所选择特征图的通道数相关, 本文选用的通道数为 64 和 2048.

### 2.2 空间注意力机制

空间注意力机制的目的是尽量减少背景对目标的干扰, 并获得目标的特征信息. 本文设计了并行支路的空间注意力机制, 该空间注意力机制拥有两条支路, 每条支路对输入特征图进行两次 1×1 卷积, 卷积的目的是将特征图的维度减少一半, 同时保持特征图的宽高不变, 经过减半维度的特征图具有较少的参数量. 然后对两条支路的特征图分别计算最大池化和平均池化, 将二者池化结果进行拼接, 最后使用 3×3 卷积, 在保持宽高不变的情况下, 将特征图的维度变为 1. 本文设计的并行结构多次使用小卷积核, 减少大卷积核带来的计算量, 获取更丰富的特征信息.

针对输入 $X$ 的维度为 $C$×$H$×$W$, 经过两次 1×1 卷积后输出维度变成 $C/4$×$H$×$W$, 然后分别计算最大池化和平均池化, 将二者池化结果进行拼接, 得到维度 2×$H$×$W$, 最后经过 3×3 卷积输出 1×$H$×$W$ 维度的特征图. 空间注意力机制的网络结构如图 7 所示.

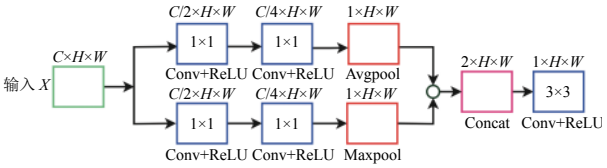


图 7 空间注意力

上述过程的描述如下:

$$F_a = ReLU(Conv_{1\times1}(ReLU(Conv_{1\times1}(X)))) \quad (7)$$

$$F_{b1} = MaxPool(F_a) \quad (8)$$

$$F_{b2} = AvgPool(F_a) \quad (9)$$

$$F_b = Conv_{3\times3}(concat(F_{b1}, F_{b2})) \quad (10)$$

式中, $X$ 是输入特征图, $ReLU$ 为卷积后的激活函数, $F_a$ 是两条支路经过两次 1×1 卷积后输出的特征图, $MaxPool$ 和 $AvgPool$ 函数分别代表最大池化和平均池化操作, $concat()$ 是拼接操作, $F_{b1}$和$F_{b2}$是两条分支的输出特征图, $F_b$为最后空间注意力的输出结果.

本文将上述的空间和通道注意力机制合称为并行支路注意力机制, 其原因在于空间和通道注意力机制均采取并行支路结构再进行拼接的方法得到特征图.

### 2.3 注意力模型融入 ResNet

本文设计的并行支路注意力机制是一种即插即用的模块, 可以将其融入到网络的任何位置. 但是在融入网络的时候需要注意以下几点.

(1) 骨干网络的作用是提取特征, 增加过多的注意力模型非但不会提高网络的性能, 还会由于加入了更多的参数量而变得臃肿, 反而不会达到理想的效果. 本文仅在骨干网络中添加了两次注意力模型.

(2) 浅层特征具有通用性, 而深层特征具有抽象性. 故可以在浅层网络和深层网络部分的适当位置添加注意力模型.

综合上述两点考虑, 本文将注意力模型融入到 ResNet50 网络中, 其结构如图 8 所示. 第一个注意力模型融入到第一次卷积之后, 可以有效地调整浅层网络的特征, 第二个注意力模型融入到最后一个残差块的输出之后, 目的是对深层网络特征进行调整.

输入图像维度为 3×608×1088, 经过第一次卷积后维度变为 64×304×544, 经过池化后输出维度变成 64×152×272, 经过 4 层残差网络后维度变成了 2048×19×34, 最后经过多层特征融合, 输出 64×152×272 大小的特征图.

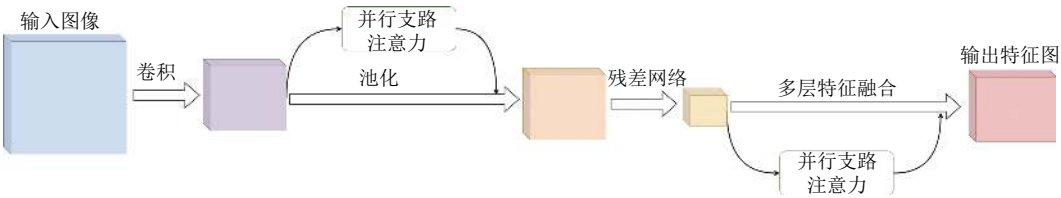


图 8 注意力模型融入到 ResNet50 网络

## 3 速度先验卡尔曼滤波

卡尔曼滤波算法[16]在目标跟踪技术中有着广泛的应用, 该算法通过对输入信号进行估计, 利用系统的观测值更新状态变量, 最后将估计值作为系统的输出值.卡尔曼滤波算法的状态方程是利用系统的上一时刻值对当前时刻进行估计, 观测方程针对当前时刻进行观测.

状态方程的表达式为:

$$x_k = Ax_{k-1} + w \tag{11}$$

测量方程的表达式为:

$$z_k = Hx_k + v \tag{12}$$

式中, $x_k$和$x_{k-1}$分别表示 $k$ 和 $k$–1 时刻的状态向量, $A$ 表示系统状态转移矩阵, $w$ 和 $v$ 分别表示状态噪音和观测噪音, 两者的协方差矩阵分别为 $Q$ 和 $R$. $H$ 表示观测矩阵.

系统预测阶段主要有两个方程, 分别对状态和协方差进行预测.

$$x_k = Ax_{k-1} \tag{13}$$

$$P_k = AP_{k-1}A^{\mathrm{T}} + Q \tag{14}$$

系统更新阶段包括了对卡尔尔曼增益系数的更新, 状态修正, 以及协方差修正.

$$K_k = P_k H^{\mathrm{T}}\left(HP_k H^{\mathrm{T}} + R\right)^{-1} \tag{15}$$

$$\hat{x}_k = x_k + K_k(z_k - Hx_k) \tag{16}$$

$$\hat{P}_k = (I - K_k H)P_k \tag{17}$$

在多目标跟踪中, 假设行人运动为匀速线性运动, 对每个行人建立运动模型, 设系统的状态变量为$x_k$, 则:

$$x_k = \left[x, y, r, h, \dot{x}, \dot{y}, \dot{r}, \dot{h}\right] \tag{18}$$

式中, $x$,$y$ 为目标框的中心点, $r$ 为目标框的宽高比, $h$ 为目标框的高, 后面 4 个变量分别为前 4 个变量对应的速度变化. 引入行人速度系数 $dt$, 则系统的状态方程如下:

$$\begin{cases} x_k = x_{k-1} + dt \times \dot{x}_{k-1} \\ y_k = y_{k-1} + dt \times \dot{y}_{k-1} \\ r_k = r_{k-1} + dt \times \dot{r}_{k-1} \\ h_k = h_{k-1} + dt \times \dot{h}_{k-1} \end{cases} \tag{19}$$

在多目标跟踪算法中, 行人速度系数的值被设为 1, 也就是在整个跟踪过程中都假定行人的速度不会发生变化, 这样的设置有两个问题. (1) 行人在相邻两帧之间的位置移动变化量不大, $dt$ 设为 1 会使行人的速度分量权重变大, 与行人实际的位置有较大的偏差, 随着视频帧不断推进, 最终会导致行人跟踪准确度下降. (2) 由于每位行人行走的速度都不一样, 因此不宜将 $dt$ 的值始终固定不变. 结合上述两点原因, 本文设计了速度先验的卡尔曼滤波算法.

本文设计了一个速度先验的计算公式, 该公式计算了预测目标中心点与下一时刻真实目标中心点之间的距离, 由于计算的时间长度是相邻两帧, 所以该距离值也是速度值, 本文称其为“速度先验公式”. 速度先验表达式为:

$$d = |x_{t+1} - \hat{x}_{t+1}| + |y_{t+1} - \hat{y}_{t+1}| \tag{20}$$

式中, 速度先验 $d$ 表示的是预测中心点与真实中心点相差的像素点个数, $(x_{t+1}, y_{t+1})$为真实目标中心点位置, $(\hat{x}_{t+1}, \hat{y}_{t+1})$为预测目标中心点位置.

具体算法流程如下:

(1) 设第 $t$ 时刻的跟踪轨迹集合为$M_t$, $M_t$中含有当前时刻各目标框的中心点位置$(x^{(i)}{}_t, y^{(i)}{}_t)$. 将第 $t$ 时刻的跟踪轨迹利用卡尔曼预测方程式 (13)、式 (14) 对第 $t$+1 时刻进行预测, 得到集合$\hat{M}_{t+1}$, 其中$(\hat{x}^{(i)}_{t+1}, \hat{y}^{(i)}_{t+1})$为各预测目标框的中心点位置.

(2) 将 $t$+1 时刻检测到的目标与第 $t$ 时刻的跟踪轨迹进行匹配, 得到 $t$+1 的跟踪轨迹时刻集合为$M_{t+1}$, 则集合$M_{t+1}$中各目标框的中心点为$(x^{(i)}{}_{t+1}, y^{(i)}{}_{t+1})$.

循环遍历集合$\hat{M}_{t+1}$和$M_{t+1}$, 用式 (20) 对每个目标计算其速度先验值.

设阈值为 $T$, 若 $d$ 的值大于阈值 $T$, 表示该目标的运动状态非常快, 需要提高该行人运动模型的速度分

量, 故将 $dt$ 值设置为 1; 若 $d$ 的值小于 1, 表明行人几乎没有移动, 此时设置其为 $\gamma$. $\gamma$ 是一个非常小的数. 在其他情况下, 将 $dt$ 值设置为 $d$ 的倒数.

$$dt=\begin{cases}1, & d\geq T\\ \gamma, & d\leq 1\\ 1/d, & \text{else}\end{cases} \tag{21}$$

至此便是速度先验卡尔曼滤波算法, 上述算法不仅可以根据每个行人的运动情况动态调节行人速度系数, 同时也起到了自适应修正系统状态方程的效果, 使跟踪算法更具有鲁棒性.

## 4 实验分析

### 4.1 实验说明

为了验证算法的可行性, 本文选择公共数据集 CUHK-SYSU[11] 作为训练集, 选择 MOT16 train[17] 作为验证集, 用来评估算法的有效性. 为了与其他算法作对比, 本文在 MOT16 测试集对算法进行测试, 并将算法测试结果上传到 MOT Challenge 官网, 与其他算法进行比较. CUHK-SYSU 数据集是一种使用在行人检索领域的数据集, 该数据集同样在文献 [7,8] 中使用. 训练数据集总共有 11 206 张图片, 每张图的图像分辨率均为 600×800, 每个行人均有身份识别号和标注的目标框.

作为验证集的 MOT16 train 文件夹的内容如表 1 所示.

表 1 数据集信息

| 数据集 | 图片分辨率 | 摄像头 | 图片数量 |
|---|---|---|---|
| MOT16-02 | 1920×1080 | 静止 | 600 |
| MOT16-04 | 1920×1080 | 静止 | 1050 |
| MOT16-05 | 640×480 | 移动 | 837 |
| MOT16-09 | 1920×1080 | 静止 | 525 |
| MOT16-10 | 1920×1080 | 移动 | 654 |
| MOT16-11 | 1920×1080 | 移动 | 900 |
| MOT16-13 | 1920×1080 | 移动 | 750 |

本实验环境基于 Ubuntu 18.04 操作系统, 采用 PyTorch 深度学习框架, CPU 和 GPU 配置为 Intel(R) Xeon(R) E5-2620 v4 和 NVIDIA Titan Xp.

在数据处理阶段, 首先将输入图像的宽高调整为 1088×608, 训练时的初始化采用预训练权重初始化. 预训练权重可以加快网络收敛, 减少训练时间. 训练初始学习率为 0.0001, 衰减方式为按轮次衰减, 每隔 20 轮学习率衰减为上一次的 0.1 倍, 总训练轮数为 30 轮. 阈值 $T$ 为 30, $\gamma$ 为 0.02.

### 4.2 评价指标

本文选取多目标跟踪领域常见的评价指标对算法进行评价, 包括多目标跟踪准确度 (Multiple Object Tracking Accuracy, MOTA)、多目标跟踪精确度 (Multiple Object Tracking Precision, MOTP)、最多跟踪轨迹数量 (Mostly Tracked, MT)、最多丢失轨迹数量 (Mostly Lost, ML)、识别 $F1$ 值 (ID $F1$ Score, $IDF1$) 和身份变换次数. MOTA 和 MOTP 是评价跟踪器首选的两个指标, 而识别 $F1$ 值则考虑了跟踪器能否长时间跟踪目标, $IDF1$ 对轨迹中 ID 信息的准确性更敏感, 也是评价跟踪器的重要指标.

选取的指标说明如表 2, 其中↑表示数值越大越好, ↓表示数值越小越好.

表 2 评价指标

| 指标名称 | | 说明 |
|---|---|---|
| MOTA | ↑ | 综合考虑漏检率、虚检率、身份变换等指标. |
| MOTP | ↑ | 检测框与预测框的匹配程度 |
| MT | ↑ | 大部分时间能持续跟踪到的轨迹数量 |
| ML | ↓ | 大部分时间不能持续跟踪到的轨迹数量 |
| *IDF*1 | ↑ | 每个行人框中身份识别的*F*1值 |
| IDs | ↓ | 目标身份发生变化的总数 |

### 4.3 实验结果

本文算法的 MOT16 测试集的测试结果上传到 MOT Challenge 的官方测试平台, 方便与其他主流算法进行比较, 该测试结果可在 MOT Challenge 官网上看到. 本文选取了近年来多目标跟踪的主流算法进行对比. 并将 ResNet50 中上采样的普通卷积替换为可变性卷积[18], 从而解决多目标跟踪中行人因摄像头等原因发生的形变问题. 表 3 是 MOT16 测试集的比较结果.

表 3 MOT16 测试集结果

| 方法 | MOTA (%) | MOTP (%) | MT | ML | *IDF*1 (%) | IDs |
|---|---|---|---|---|---|---|
| TAP[19] | 64.8 | 78.7% | 292 | 164 | **73.5** | **571** |
| Deepsort_2[5] | 61.4 | 79.1 | 249 | 138 | 62.2 | 781 |
| POI[6] | 66.1 | 79.5 | 258 | 158 | 65.1 | 805 |
| CNNMTT[20] | 65.2 | 78.4 | 246 | 162 | 62.2 | 946 |
| 本文 | 65.1 | 78.8 | 229 | 144 | 62.3 | 2129 |
| 文献[8] | **69.3** | **80.2** | **306** | **127** | 72.3 | 815 |

将本文算法与 MOT16 排行榜中 private 赛道的近年主流算法进行对比, 其中 TAP, Deepsort_2, POI,

CNNMTT 选用的检测器都是 Faster-RCNN[21], 且骨干网络是 VGG-16[22].

由表 3 可知, 本文算法与前 4 个算法相比较都有较好的跟踪性能, 联合检测和重识别框架能有效地完成目标检测与重识别任务. 与 Deepsort_2 相比, 在 MOTA 值上提高了 3.7%, *IDF*1 相差不大, 其原因在于 Deepsort_2 采用的是检测与重识别分离框架, 说明本文算法在跟踪准确度上略显优势. 本文的算法与 POI 和 CNNMTT 的整体性能差别不大. 与 TAP 相比, MOTA 相差不大, 而 *IDF*1 小了近 10%, 说明 TAP 的长时跟踪能力优于本文的算法.

本文算法与文献 [8] 均采用了联合检测和重识别框架, 文献 [8] 总共选用了 6 个数据集作为训练数据, 其训练规模比本文算法更大, 但本文算法与文献 [8] 的跟踪性能总体差距并不大. 综合而言, 本文算法具有良好的综合性能.

为了更直观的说明本文算法的长时跟踪性能, 将 MOT16 测试集中的 MOT16-03 跟踪结果可视化.

由图 9 可知, 在第 194 帧的时候, id 编号 4 绿色框的行人被灯柱完全遮挡, 直到第 285 帧再次出现的时候, 其 id 编号并没有发现变化, 跟踪依然在持续进行. 在第 513 帧对于 id 编号为 50 的深蓝色框的行人, 在密集的场景下也能被跟踪到, 第 554 帧的时候该行人被完全遮挡后直到第 578 帧才出现, 其 id 编号并没有发现变化. 本文算法在行人受到遮挡后仍能准确分辨出来, 分配同样的 id 编号, 使跟踪持续进行.

(a) 第 128 帧 (b) 第 194 帧 (c) 第 285 帧

(d) 第 513 帧 (e) 第 554 帧 (f) 第 578 帧

图 9 MOT16-03 跟踪结果

### 4.4 消融实验

以下从骨干网络和速度先验卡尔曼滤波进行对比实验并分析结果.

(1) 选取不同的骨干网络. 本文对比了 3 种不同的骨干网络: ResNet50dcn、ResNet50dcn_fpn、ResNet50dcn_fpn_att 来验证所改进的骨干网络的可行性. 其中 ResNet50dcn 是将上采样中的普通卷积替换为可变性卷积, ResNet50dcn_fpn 在此基础上加入了多层特征融合, ResNet50dcn_fpn_att 又融合了并行支路注意力模型. 3 种骨干网络在验证集的平均指标对比如表 4 所示.

表 4 不同骨干网络的实验结果对比

| 网络 | MOTA (%) | MOTP (%) | MT | ML | *IDF*1 (%) | IDs |
|---|---|---|---|---|---|---|
| ResNet50dcn | 53.4 | 78.2 | 125 | 169 | 53.2 | 788 |
| ResNet50dcn_fpn | 56.7 | 78.5 | 142 | 135 | 54.3 | 825 |
| ResNet50dcn_fpn_att | **57.0** | 77.7 | **154** | **126** | **56.3** | 1025 |

根据上述的对比可知, 加入多层特征融合的 ResNet-50dcn_fpn 的跟踪性能指标相比于 ResNet50dcn 均有较大的提升, 其中 MOTA 提升了 3.3%, MOTP 提升了 0.3%, 在跟踪轨迹上 MT 和 ML 均优于 ResNet50dcn. 说明在多目标跟踪算法中, 多层特征融合对目标的特征提取有效果, 并提升跟踪器的跟踪性能. 而加入了并行支路注意力机制的算法要优于 ResNet50dcn_fpn. 相

比于 ResNet50dcn_fpn, 加入了并行支路注意力机制后, MOTA 提高了 0.3%, *IDF*1 提高了 2%, 持续跟踪到的轨迹数量从 142 增加到 154, 而不能持续跟踪到的轨迹数量由 135 降到了 126. 说明本文提出的并行结构注意力进一步改善了特征的表达能力, 能有效、持续地跟踪目标, 具有较好的鲁棒性.

(2) 速度先验卡尔曼滤波. 本文首先将 ResNet50dcn_fpn_att 与本文算法进行对比, 其中 ResNet50dcn_fpn_att 中用的是没有改进的卡尔曼滤波, 而本文算法则使用的是速度先验卡尔曼滤波算法. 两者的对比结果如表 5 所示.

表 5 改进卡尔曼滤波算法的实验结果对比

| 算法 | MOTA (%) | MOTP (%) | MT | ML | *IDF*1 (%) | IDs |
|---|---|---|---|---|---|---|
| ResNet50dcn_fpn_att | 57.0 | 77.7 | 154 | 126 | 56.3 | 1025 |
| 本文算法 | **57.1** | 77.6 | 153 | 130 | **60.4** | **911** |

根据表 5 的对比可知, 本文算法在 *IDF*1 上提高了 4.1%, 在身份变换次数上从 1025 减少到了 911, 降低了 11%, 而 MOTA 以及 MOTP 的变化很小, *IDF*1 对轨迹中目标的身份信息变化更敏感. 本文设计的速度先验卡尔曼滤波在跟踪精度不变的前提下, 提高了 *IDF*1 值, 同时降低了身份变化次数, 使跟踪器能长时间跟踪目标.

本文将验证集下的 7 个数据集的身份变换次数和 *IDF*1 值进行了对比, 其中 Att 代表 ResNet50dcn_fpn_att, Ours 为本文算法.

由图 10(a) 可知, 本文算法相比于 ResNet50dcn_fpn_att 身份变换次数降低了很多, 其中在 MOT16-02 中减少了 24.2%, 在 MOT16-04 中降低了 19.7%. 图 10(b) 中的对比可知, 本文算法的 *IDF*1 准确值有较大的提升. 当摄像头是静止的时候, MOT16-02 中提高了 4.6%, MOT16-04 中提高了 7.05%, MOT16-09 提高了 7.39%. 而当摄像头移动的时候, *IDF*1 值则有所下降. 例如 MOT16-10 中 *IDF*1 减少了 3.15%, MOT16-13 中 *IDF*1 减少了 0.5%.

本文在验证集中选取了其中的 6 个数据集, 对速度先验值的分布进行可视化.

为了对可视化图像进行平滑处理, 本文采用每隔 20 帧对速度先验值进行抽取, 将抽取帧中的速度先验值取平均并用折线图画出, 灰色阴影部分则是抽取帧中的所有速度先验值分布情况. 从图 11 可以看出来, 当摄像头处于静止时, MOT16-02、MOT16-04 以及 MOT16-09 的速度先验值分布较相似, 而在其他 3 个摄像头移动的场景中, 速度先验值的分布显得更加突兀.

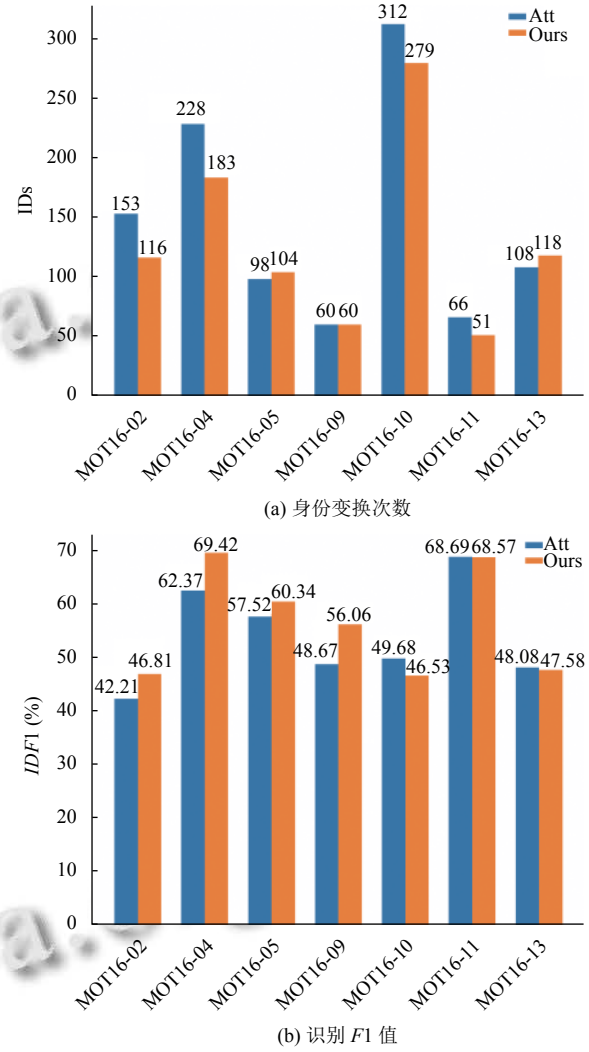


图 10 验证集中指标对比

通过图 10 和图 11 的分析可以知道, 本文设计的速度先验卡尔曼滤波更适用于摄像头静止的场景, 并不适用于摄像头移动的场景. 其原因可能在于尽管目标移动速度较慢, 速度先验距离较小, 但由于摄像头移动速度较快, 导致 *IDF*1 值下降, 最终无法长时间跟踪目标.

## 5 结束语

本文针对多目标跟踪中行人被遮挡后再次无法准确跟踪的问题, 选用了联合检测和重识别框架, 提出了并行支路注意力机制并将其融入骨干网络中, 并设计了

速度先验卡尔曼滤波. 在 MOT16 测试集上通过对比实验表明, 在多目标跟踪算法中引入本文设计的并行支路注意力机制并采用速度先验卡尔曼滤波可以有效地提高多目标跟踪精度和性能, 使算法具有一定的鲁棒性.

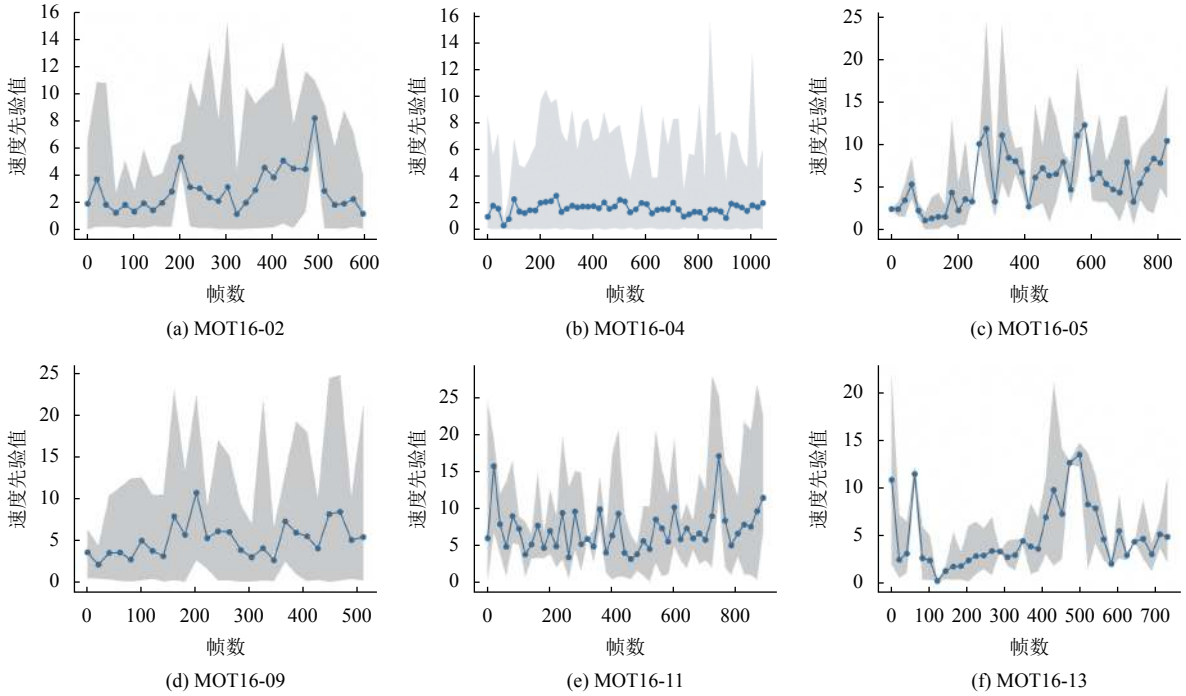


图 11 距离可视化

## 参考文献


1 Ciaparrone G, Sánchez FL, Tabik S, *et al*. Deep learning in video multi-object tracking: A survey. Neurocomputing, 2020, 381: 61–88. [doi: 10.1016/j.neucom.2019.11.023]

2 Milan A, Roth S, Schindler K. Continuous energy minimization for multitarget tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2014, 36(1): 58–72. [doi: 10.1109/tpami.2013.103]

3 Song YM, Jeon M. Online multiple object tracking with the hierarchically adopted GM-PHD filter using motion and appearance. Proceedings of 2016 IEEE International Conference on Consumer Electronics-Asia. Seoul: IEEE, 2016. 1–4. [doi: 10.1109/icce-asia.2016.7804800]

4 Leal-Taixé L, Canton-Ferrer C, Schindler K. Learning by tracking: Siamese CNN for robust target association. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops. Las Vegas: IEEE, 2016. 33–40. [doi: 10.1109/CVPRW.2016.59]

5 Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. Proceedings of 2017 IEEE International Conference on Image Processing. Beijing: IEEE, 2017. 3645–3649. [doi: 10.1109/icip.2017.8296962]

6 Yu FW, Li WB, Li QQ, *et al*. POI: Multiple object tracking with high performance detection and appearance feature. Proceedings of the European Conference on Computer Vision. Amsterdam: Springer, 2016. 36–42. [doi: 10.1007/978-3-319-48881-3_3]

7 Wang ZD, Zheng L, Liu YX, *et al*. Towards real-time multi-object tracking. arXiv: 1909.12605, 2020.

8 Zhang YF, Wang CY, Wang XG, *et al*. FairMOT: On the fairness of detection and re-identification in multiple object tracking. arXiv: 2004.01888, 2020.

9 Redmon J, Divvala S, Girshick R, *et al*. You only look once: Unified, real-time object detection. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 779–788. [doi: 10.1109/cvpr.2016.91]

10 Zhou XY, Wang DQ, Krähenbühl P. Objects as points. arXiv: 1904.07850, 2019.

11 Xiao T, Li S, Wang BC, *et al*. Joint detection and identification feature learning for person search. Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu: IEEE, 2017. 3415–3424. [doi: 10.1109/cvpr.2017.360]

12 He KM, Zhang XY, Ren SQ, *et al*. Deep residual learning for image recognition. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 770–778. [doi: 10.1109/cvpr.2016.90]

13 Lin TY, Dollár P, Girshick R, *et al*. Feature pyramid networks for object detection. Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu: IEEE, 2017. 2117–2125.

14 Hu J, Shen L, Sun G. Squeeze-and-excitation networks. Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City: IEEE, 2018. 7132–7141. [doi: 10.1109/cvpr.2018.00745]

15 Woo S, Park J, Lee JY, *et al*. CBAM: Convolutional block attention module. Proceedings of the 15th European Conference on Computer Vision. Munich: Springer, 2018. 3–19. [doi: 10.1007/978-3-030-01234-2_1]

16 Jang DS, Kim GY, Choi HI. Kalman filter incorporated model updating for real-time tracking. Proceedings of Digital Processing Applications. Perth: IEEE, 1996. 878–882. [doi: 10.1109/tencon.1996.608463]

17 Milan A, Leal-Taixé L, Reid I, *et al*. MOT16: A benchmark for multi-object tracking. arXiv: 1603.00831, 2016.

18 Dai JF, Qi HZ, Xiong YW, *et al*. Deformable convolutional networks. Proceedings of 2017 IEEE International Conference on Computer Vision. Venice: IEEE, 2017. 764–773. [doi: 10.1109/iccv.2017.89]

19 Zhou ZW, Xing JL, Zhang MD, *et al*. Online multi-target tracking with tensor-based high-order graph matching. Proceedings of the 24th International Conference on Pattern Recognition. Beijing: IEEE, 2018. 1809–1814. [doi: 10.1109/ICPR.2018.8545450]

20 Mahmoudi N, Ahadi SM, Rahmati M. Multi-target tracking using CNN-based features: CNNMTT. Multimedia Tools and Applications, 2019, 78(6): 7077–7096. [doi: 10.1007/s11042-018-6467-6]

21 Ren SQ, He KM, Girshick R, *et al*. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 39(6): 1137–1149. [doi: 10.1109/tpami.2016.2577031]

22 Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. Proceedings of the 3rd International Conference on Learning Representations. San Diego: ICLR. 2015. 1–5.