Abstract

基于视觉的多对象跟踪（MOT）的最新趋势正朝着利用深度学习的代表性功能联合学习检测和跟踪对象的方向发展。但是，现有方法仅使用损失函数来训练某些子模块，这些子函数通常与已建立的跟踪评估措施不相关，例如多对象跟踪准确性（MOTA）和精度（MOTP）。由于这些措施不可微分的，因此**选择合适的损失函数**进行多目标跟踪方法的**端到端训练**仍然是一个开放的研究问题。在本文中，我们通过**提出可微分的 MOTA 和 MOTP 代理**来弥合这一差距，我们**将它们组合成适合深层多目标跟踪器端到端训练的损失函数**。作为**关键要素**，我们提出了**近似匈牙利匹配算法**的 **Deep Hungarian Net（DHN）模块**。 DHN 允许估算对象轨迹与 GT 对象之间的对应关系，以计算出 MOTA 和 MOTP 的可区分代理，然后将它们**直接用于优化深度跟踪器**。我们通过实验证明了所提出的可区分框架提高了现有多对象跟踪器的性能，并且我们在 MOTChallenge 基准上建立了新的技术水平。我们的代码可从 https://github.com/yihongXU/ deepMOT 公开获得。

1.Introduction

基于视觉的多目标跟踪（MOT）是长期存在的研究问题，在移动机器人技术和自动驾驶中具有应用。 通过跟踪，我们可以了解周围的对象实例并预测它们的未来运动。 现有的大多数行人跟踪方法是按照检测跟踪范例进行的，主要是关注随着时间的推移检测器响应的关联。 大量研究调查了针对此挑战性数据关联问题的组合优化技术[39、38、45、55、7、6]。

MOT 中最新的数据驱动趋势利用深层网络的表示能力来学习数据关联的身份保存嵌入[26、47、51]，学习单个目标的外观模型[12,56]并学习回归检测到的目标的姿态[3]。 但是，这些方法使用代理损耗（例如，学习身份嵌入的三元组损耗[44]）来训练 MOT 管道的各个部分，**这些损耗仅与 MOT 评估措施[5]间接相关**。 定义类似于标准跟踪评估方法的损耗函数的主要困难在于，需要计算预测物体轨迹与 GT 物体之间的最佳匹配度。通常使用匈牙利（Munkres）算法（HA）[25]解决此问题，**该算法包含不可微操作**。

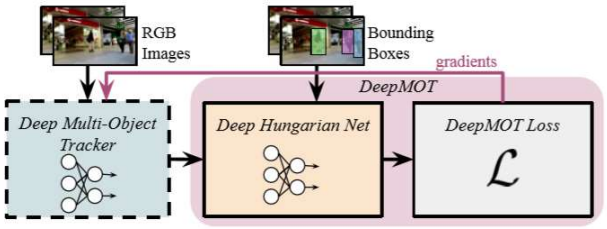

Figure 1. We propose DeepMOT, a general framework for training deep multi-object trackers including the DeepMOT loss that directly correlates with established tracking evaluation measures [5]. The key component in our method is the Deep Hungarian Net (DHN) that provides a soft approximation of the optimal prediction-to-ground-truth assignment, and allows to deliver the gradient, back-propagated from the approximated tracking performance measures, needed to update the tracker weights.

本文的重要贡献是一种新颖的，可微的框架，用于训练多目标跟踪器（图 1）。特别是，我们提出了标准 CLEAR-MOT [5]评估措施的可微分形式，我们将其组合为一个新颖的损失函数，适用于端到端的 MOT 方法培训。特别是，我们引入了一个可微分的网络模块-匈牙利深网（DHN）-近似匈牙利算法，并为 GT 分配提供了最佳预测的软近似。所提出的**近似值基于双向递归神经网络（BiRNN）**，该双向神经网络基于预测和 GT 距离矩阵计算（软）分配矩阵。然后，**我们将多目标跟踪精度（MOTA）和精度（MOTP）[5]表示为计算的（软）分配矩阵和距离矩阵的可微函数**。然后，它们**充当传递所需的梯度的桥梁**，以更新从近似跟踪性能指标向后传播的跟踪器权重。这样，我们可以使用与标准 MOT 评估度量直接相关的损

失，以数据驱动的方式训练对象跟踪器。总而言之，本文做出了**以下贡献**：

（i）我们**提出了新颖的损失函数**，这些函数直接受到标准 MOT 评估方法的启发[5]，用于多目标跟踪器的**端到端训练**。

（ii）**为了通过网络反向传播损失**，我们**提出了一个新的网络模块-匈牙利深网**-该模块学习以不同的方式将预测的轨迹与 GT 进行匹配。

（iii）通过使用我们提出的框架培训最新发布的 Tracktor [3]，我们证明了提出的损失函数和可区分匹配模块的优点。我们展示了在基线之上的改进，并在 MOTChallenge 基准数据集上建立了一个最新的结果[34，27]。

2.Related Work

Tracking as Discrete Optimization. 随着可靠的目标检测器[13、16、28]的出现，逐次检测跟踪已成为领先的跟踪范例。 这些方法首先在每个图像中执行对象检测，并随时间关联检测，这可以通过轨迹和检测之间的帧对帧双向匹配在线进行[25]。 由于早期的检测器嘈杂且不可靠，因此有几种方法可以寻找最佳的批量生产方式，这通常是网络流量优化问题[38、45、55、7、6]。

可替代地，通过寻找最佳的轨道集合作为顺序轨道状态的条件分布，可以将跟踪作为最大后验（MAP）估计问题。 几种方法使用条件随机域（CRF）[35、11、37]，马尔可夫链蒙特卡洛（MCMC）[36]或变分期望最大化[1，2]进行推理。 这些方法通常将手工制作的描述符用于外观模型，例如颜色直方图[35，9]，基于光学流的描述符[11]和/或运动模型[28，37]作为关联提示。 因此，通常只有少数几个参数是可训练的，并且通常使用网格/随机搜索或 parzen 窗口估计器树来学习[4，37]。 在基于 CRF 的方法中，可以使用结构化 SVM 来训练权重[49，52]。

Deep Multi-Object Tracking. MOT 利用深度神经网络表示能力的最新数据驱动趋势。Xiang 等。[53]通过将其建模为马尔可夫决策过程（MDP）来学习跟踪出生/死亡/关联策略。作为标准评估方法[5]不可区分，通过强化学习来学习策略。

几种现有的方法使用损失来训练其跟踪方法的一部分，与跟踪评估方法没有直接关系[5]。 Kim 等。 [22]利用预先学习的 CNN 功能或双线性 LSTM [23]学习长期外观模型。两者都被合并到 MHT 跟踪框架中[39]。其他方法[26、51、47]使用深层神经网络来学习数据保存的身份识别嵌入，并使用对比[17]，三元组[44]或四元组丢失[47]进行训练。在推论时，这些用于计算数据关联能力。 [12,56]的方法使用共享卷积主干的单对象跟踪器集成来学习单个目标的外观模型。时空机制（使用交叉熵损失在线学习）指导在线外观适应并防止漂移。所有这些方法都只是部分训练的，有时是不同的阶段。此外，还不清楚如何训练这些方法以最大化已建立的跟踪指标。

与我们的目标最相似的是 Wang 等。 **[52]提出了一种学习线性成本关联函数参数的框架，适用于基于网络流量优化[55]的多目标跟踪器。** 他们使用结构化 SVM 训练参数。 与我们的方法相似，他们设计了一种类似于 MOTA 的损失函数：**帧内损失惩罚 FP 和丢失的目标**，而**损失的帧间分量则惩罚错误的关联，ID 切换和丢失的关联**。 然而，**它们的损失是不可区分的，仅适用于建议的最小成本流框架内的训练参数**。 舒尔特等。 [45]参数化（任意）成本函数具有神经网络，并通过针对最小流量训练目标进行优化来端对端地对其进行训练。与[45]不同，我们的方法**超越了学习关联函数的范围，可以被任何可学习的跟踪方法使用**。

Bergmann 等。[3]提出了一种回归方法来跟踪 MOT。 使用边界框回归器的平滑 L1 损失对方法进行训练，以进行对象检测任务。 根据经验，他们的方法能够使高帧率视频序列中的边界框回归，而没有明显的摄像机运动。 **除了跟踪生死管理外，此方法是完全可培训的，因此它是证明我们培训框架优点的理想方法**。 使用我们提出的损失对序列级数据进行

这种方法的训练进一步提高了性能，并建立了基于 MOTChallenge 基准的最新技术[27]。

3.Overview and Notation

任何 MOT 方法的目的都是预测视频序列中的轨道。每一条轨道X^i与一个 ID i绑定，并且由L_i图像边界框$x_{t_l}^i \in \mathbb{R}^4$（二维位置和大小）组成，$l = 1\ldots, L_i$。 多目标跟踪器的任务是通过时间准确估计所有身份的边界框。

在评估时，标准指标逐帧运行。 在帧t处，必须将N_t个预测边界框$x_t^{i_1}, \ldots, x_t^{i_{N_t}}$与$M_t$个 GT 对象$y_t^{i_1}, \ldots, y_t^{i_{M_t}}$进行比较。 我们首先需要计算预测的边界框和 GT 物体之间的对应关系。这不是一个简单的问题，因为多个 GT 的盒子可能会重叠，因此可能会适应多个假设。 在下文中，我们将省略时间索引 t 以简化阅读。 除非另有说明，否则所有表达式将相对于时间索引 t 进行求值。

在[5]中提出的标准度量标准使用双向匹配来解决此关联问题。 首先，计算预测框到 GT 框的距离矩阵$D \in \mathbb{R}^{N\times M}$， $d_{nm} \in [0,1]$。 对于基于视觉的跟踪，通常使用基于 IoU 的距离。然后，通过使用匈牙利算法[25]求解以下整数程序，获得最优的预测框到 GT 框的真值分配二进制矩阵：

$$\mathbf{A}^* = \underset{\mathbf{A}\in\{0,1\}^{N\times M}}{\operatorname{argmin}} \sum_{n,m} d_{nm}a_{nm}, \ \text{s.t.} \sum_m a_{nm} \leq 1, \forall n;$$
$$\sum_n a_{nm} \leq 1, \forall m; \ \sum_{m,n} a_{nm} = \min\{N, M\}.$$

通过求解此整数程序，我们获得了 GT 与轨迹预测之间的相互一致的关联。 约束条件确保分配的所有行和列的总和应为 1，从而避免两组之间的多个分配。找到最佳关联A^*后，我们可以使用A^*和D计算 MOTA 和 MOTP 度量：

$$\text{MOTA} = 1 - \frac{\sum_t(\text{FP}_t + \text{FN}_t + \text{IDS}_t)}{\sum_t M_t}, \qquad (1)$$

$$\text{MOTP} = \frac{\sum_t \sum_{n,m} d_{tnm}a^*_{tnm}}{\sum_t |\,\text{TP}_t\,|}, \qquad (2)$$

其中a^*_{tnm}是时间 t 处 A^* 的第(n, m)个输入。真正 (TP) 对应于匹配的预测轨道的数量，假正 (FP) 对应于不匹配的预测轨道的数量。假阴性(FN)表示不匹配的预测的数量。 最后，要计算 ID 交换次数 IDS，我们需要跟踪过去的帧分配。 每当分配给 GT 物体的轨道发生变化时，我们都会增加 IDS 的数量并更新分配结构。

由于这些评估措施不可微分，因此现有策略只能优化跟踪器的超参数（使用例如随机搜索或网格搜索），以最大化 MOTA 或 MOTP 或两者的组合。 在当前版本中，MOTA 和 MOTP 不能直接用于具有梯度下降技术的跟踪器优化。

4. DeepMOT

计算 CLEAR-MOT 跟踪评估措施的第一步是在地面真轨对象集和预测轨道之间进行双向匹配。一旦建立了两组之间的对应关系，我们就可以计算表达 MOTA 所需的匹配数（TP），错过的目标（FN）和 ID 开关（IDS）以及 MOTP 的平均匹配距离。

作为本文的主要贡献，我们建议采用相同的两步策略，从这些措施中获得可弥补的损失。我们首先建议**使用可微分函数（参数化为深层神经网络）在两组之间执行软匹配。 建立匹配后，我们将（软）分配矩阵和距离矩阵的微分函数组合起来，设计近似 CLEAR-MOT 度量**

的损失。 诸如 IDF1 [41]之类的替代措施着眼于跟踪器正确识别目标的时间，而不是不匹配发生的频率。 但是，MOTA 和 IDF1 具有很强的相关性。 我们的结果反映了这一点–通过优化损失，我们还改进了 IDF1 措施（请参见第 5.3 节）。 在下文中，我们将讨论差分匹配模块（第 4.1 节）和 CLEAR-MOT 度量值的差分版本[5]（第 4.2 节）。

4.1.Deep Hungarian Net: DHN

在本节中，我们介绍了 Deep Hungarian Net（DHN），它是 DeepMOT 框架的基本组成部分。 DHN 网络产生一个可微分的代理$\tilde{A}$。 因此，DHN 提供了一个桥梁，可将梯度从损失（稍后描述）传递到跟踪方法。我们用输入距离矩阵D并输出代理分配矩阵$\tilde{A}$的非线性映射形式对 DHN 进行形式化。

DHN 由具有参数ω_d的神经网络$\tilde{A} = g(D, \omega_d)$建模。 重要的是，DHN 映射必须满足几个属性：

（i）输出$\tilde{A}$必须是最佳分配矩阵A^*的良好近似;

（ii）此近似必须对于D可微分;

（iii）输入和输出矩阵都相等，但大小不同

（iv）g必须像 HA 一样做出全局决策。

要求（i）将在训练 DHN 时通过设置适当的损失函数来实现（请参阅第 5.1 节），（ii）通过将 DHN 设计为可微分函数的组合来确保。 要求（iii）和（iv）促使我们设计一个可以处理可变（但相等）输入和输出大小的网络，其中每个输出神经元的接受域都等于整个输入。**我们选择双向递归神经网络（Bi-RNNs）**。 或者，可以考虑使用全卷积网络，因为它们将能够处理可变的输入/输出大小。 但是，大的分配问题将导致部分接受领域，因此会导致局部分配决策。

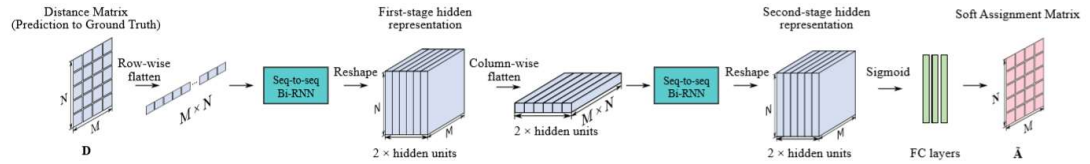

Figure 2. Structure of the proposed Deep Hungarian Network. The row-wise and column-wise flattening are inspired by the original Hungarian algorithm, while the Bi-RNN allows for all decisions to be taken globally, thus is accounting for all input entries.

我们在图 2 中概述了我们提出的架构。为了使用递归神经网络处理 2D 距离矩阵 D，我们对距离矩阵进行行（列）填充（隐藏表示）。 这是受到原始匈牙利算法的启发的，该算法依次执行逐行和逐列的归约和零项验证。 这些表示被馈送到 Bi-RNN（请参见下面的详细信息），从而为$g(\cdot)$做出全局分配决策提供了可能性。

更详细地讲，我们依次执行 flattening，即，首先按行，然后按列。行平化距离矩阵 D 输入到第一个 Bi-RNN，该 Bi-RNN 输出大小为 N×M×2h 的第一阶段隐藏表示，其中 Bi-RNN 隐藏层的大小保持不变。直观地，第一阶段的隐藏层表示对行中间分配进行编码。然后，我们逐列地填充第一阶段的隐藏表示，以输入到第二个（不同的）Bi-RNN，后者生成大小为 N×M×2h 的第二阶段的隐藏表示。两个 BiRNN 具有相同的隐藏大小，**但是它们不共享权重**。直观地，第二阶段的隐藏表示对最终的分配进行编码。为了将这些编码转换为最终的编码，我们通过三个完全连接的层（沿着 2h 维度，即，独立于原始 D 的每个元素）提供第二阶段隐藏表示。最后，S 形激活产生最优的 N×M 个软分配矩阵$\tilde{A}$。请注意，与匈牙利算法的二进制输出相反，DHN 输出一个（软）分配矩阵$\tilde{A} \in [0,1]^{N \times M}$。

Distance Matrix Computation. 衡量两个边界框之间相似度的最常用指标是 IoU。 注意，原则上，D的输入可以是任何（可微分的）距离函数。 但是，如果两个边界框没有交集，则距离 1 -IoU 将始终为常数 1。在这种情况下，损耗的梯度将为 0，并且不会向后传播信息。

因此，我们的距离是欧几里得中心点距离和 Jaccard 距离J（定义为 1–IoU）的平均值：

$$d_{nm} = \frac{f(\mathbf{x}^n, \mathbf{y}^m) + \mathcal{J}(\mathbf{x}^n, \mathbf{y}^m)}{2}. \tag{3}$$

f is the Euclidean distance normalized w.r.t. the image size:

$$f(\mathbf{x}^n, \mathbf{y}^m) = \frac{\|c(\mathbf{x}^n) - c(\mathbf{y}^m)\|_2}{\sqrt{H^2 + W^2}}, \tag{4}$$

其中函数$c(\cdot)$计算边界框的中心点，而 H 和 W 分别是视频帧的高度和宽度。 归一化的欧几里得距离和 Jaccard 距离的值都在[0,1]范围内，因此所有项d_{nm}也是如此。 我们的框架允许任何表示为可微分距离函数组成的距离。 在实验部分，我们演示了添加一个 term 的好处，该 term 可以测量两个学习的外观嵌入之间的余弦距离。

在下一部分中，我们将说明如何根据距离矩阵D和软分配矩阵$\tilde{A}$来计算 MOTA 和 MOTP 的可微分代理。

4.2.Differentiable MOTA and MOTP

在本节中，我们将详细介绍拟议的 DeepMOT 损耗的两个组成部分：微分 MOTA（dMOTA）和 MOTP（dMOTP）。如第二节所述。3，为了计算经典的 MOTA 和 MOTP 评估方法，我们首先找到了预测轨道和 GT 物体之间的最佳匹配。 基于最佳分配矩阵A^*，我们计算假阴性 FN，假阳性 FP 和 ID 开关 IDS。 后者是通过比较两个连续帧之间的分配来计算的。 为了计算建议的 dMOTA 和 dMOTP，我们需要将所有这些表示为距离矩阵 D 和使用 DHN 计算的软分配矩阵$\tilde{A}$的可微函数（请参见第 4.1 节）。

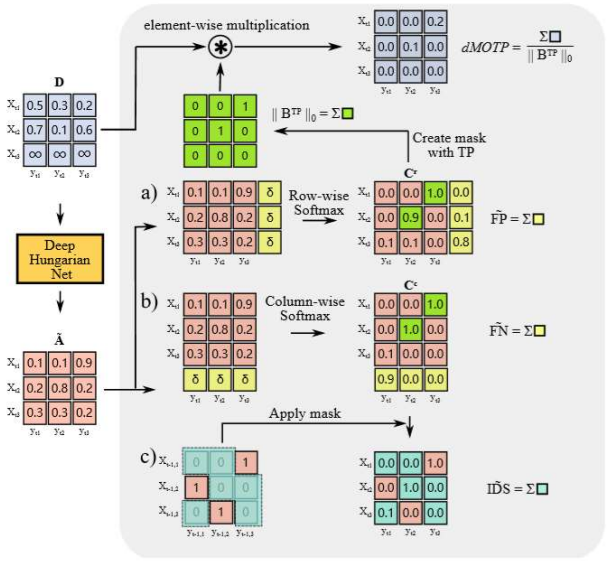

Figure 3. DeepMOT loss: $dMOTP$ (top) is computed as the average distance of matched tracks and $dMOTA$ (bottom) is composed with $\tilde{\text{FP}}$, $\tilde{\text{IDS}}$ and $\tilde{\text{FN}}$.

下面描述的操作如图 3 所示。首先，我们需要计算假阳性和假阴性。 因此，我们需要获取不匹配的轨迹和不匹配的地面真实物体的数量。 为此，我们首先通过在$\tilde{A}$处附加一列来构造矩阵C^r，并填充阈值δ，然后执行逐行 softmax（图 3a）。 类似地，我们通过将行附加到$\tilde{A}$来构造C^c并执行逐列的 softmax（图 3b）。 然后，我们可以将误报和误报的数量的近似表示为：

$$\tilde{\text{FP}} = \sum_n \mathbf{C}^r_{n,M+1}, \quad \tilde{\text{FN}} = \sum_m \mathbf{C}^c_{N+1,m}. \tag{5}$$

直观地，如果$\tilde{A}$中的所有元素都小于阈值δ（例如$\delta = 0.5$），则$C^r_{n,M+1}$和$C^c_{N+1,m}$整体将接近 1，这表明我们具有 FN 或 FP。 否则，分别在C^r和C^c的每个行/列中具有最大值的元素将

接近 1，表示我们有一个匹配项。 因此，C^c的第 N + 1 行（图 3b）和C^r的 M + 1 列（图 3a）的总和提供了 FN 数量和 FP 数量的软估算，分别。 我们将这些称为$\widetilde{FN}$和$\widetilde{FP}$。

为了计算 ID 转移$\widetilde{IDS}$和 dMOTP 的软逼近，我们还需要构造两个二进制矩阵B^{TP}和B_{-1}^{TP}，它们的非零项分别在当前帧和前一帧发出真正信号。 这些矩阵的行索引对应于分配给我们的轨道的索引，列索引对应于地面真实物体的标识。 我们需要填充B_{-1}^{TP}进行逐元素乘法，因为轨道和对象的数量在帧与帧之间有所不同。 为此，我们通过填充B_{-1}^{TP}的行和列以通过从B^{TP}复制它们对应的行和列来适应当前帧处新出现的对象的矩阵大小。 请注意，我们不需要修改B^{TP}来补偿新出现的对象，因为它们不会引起 IDS。 通过这种构造，$C_{1:N,1:M}^{c}\odot\bar{B}_{-1}^{TP}$的总和（其中$\bar{B}$是$B$的二进制补码）产生了 IDS 的（近似）数（图 3c）：

$$\widetilde{\mathrm{IDS}} = \| \mathbf{C}^{c}_{1:N,1:M} \odot \overline{\mathbf{B}}^{\mathrm{TP}}_{-1} \|_1, \tag{6}$$

其中$\|.\|_1$是平化矩阵的 L1 范数。 使用这些成分，我们可以评估 dMOTA：

$$dMOTA = 1 - \frac{\widetilde{\mathrm{FP}} + \widetilde{\mathrm{FN}} + \gamma\,\widetilde{\mathrm{IDS}}}{M}. \tag{7}$$

参数γ控制我们分配给 IDS 的代价。 同样，我们可以将 dMOTP 表示为：

$$dMOTP = 1 - \frac{\| \mathbf{D} \odot \mathbf{B}^{\mathrm{TP}} \|_1}{\| \mathbf{B}^{\mathrm{TP}} \|_0}. \tag{8}$$

直观地讲，L1 范数表示匹配的轨迹和地面真实物体之间的相似性，零范数$\|.\|_0$计算匹配的次数。 由于我们应该训练跟踪器以最大程度地提高 MOTA 和 MOTP，因此建议以下 DeepMOT 损失：

$$\mathcal{L}_{\mathrm{DeepMOT}} = (1 - dMOTA) + \lambda(1 - dMOTP), \tag{9}$$

其中λ是损耗平衡因子。 通过最小化我们提出的损失函数L_{DeepMOT}，我们对假阳性，假阴性和 ID 转换进行了惩罚-所有这些都由 CLEAR-MOT 措施使用[5]。 与标准 CLEAR-MOT 度量一样，必须在每个时间帧 t 上计算 dMOTA，dMOTP 和L_{DeepMOT}。

4.3.How To Train Your Deep Multi-Object Tracker

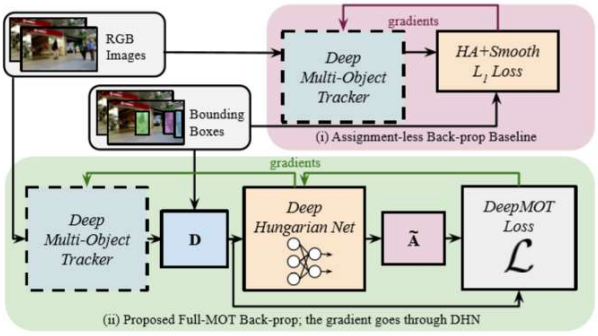

Figure 4. The proposed MOT training strategy (bottom) accounts for the track-to-object assignment problem, that is solved by the proposed deep Hungarian network, and approximates the standard MOT losses, as opposed to the classical training strategies (top) using the *non-differentiable* Hungarian algorithm.

整个跟踪器的训练过程如图 4 所示，其过程如下。我们从训练视频序列中随机采样一对连续的帧。这两个图像连同其真实边界框构成一个训练实例。对于每个这样的实例，我们首先使用 GT 边界框初始化轨道（在时间 t 处），然后运行前向通过以获取下一个视频帧（时间 t + 1）中的轨道边界框预测。为了模拟不完美检测的效果，我们向 GT 边界框添加随机扰动（有关详细信息，请参见补充材料）。然后，我们计算轨道边界框预测与 GT 边界框之间的距离矩阵，并使用我们提出的 DHN 来计算软分配（第 4.1 节）。最后，我们根据距离矩阵和

轨道与 GT 之间的预测分配来计算代理损耗（第 4.2 节）。这为我们提供了一个用于分配任务的渐变，该渐变用于更新跟踪器的权重。

5.Experimental Evaluation

在本节中，我们首先通过实验验证我们提出的 DHN 匹配网络是匈牙利算法[25]的良好近似，适用于 MOT 评估方法（第 5.1 节）所要求的双部分匹配。 为了展示所提议的框架实践的主题，我们在几个数据集上进行了几次跟踪实验，以评估行人的跟踪性能（第 5.2 节）。

5.1.Deep Hungarian Net

在本节中，我们将深入介绍可区分匹配模块的性能，并概述培训和评估细节。

DHN Training. 为了训练 DHN，我们创建一个具有成对矩阵（D和A^*）的数据集，将其分为 114,483 个矩阵进行训练和 17,880 个矩阵进行测试。 我们**使用 GT 边界框和公共检测生成距离矩阵D**，由 MOT 挑战数据集提供[34，27]。 我们**使用匈牙利算法生成相应的分配矩阵A^*（作为训练的标签）**。 我们**利用 focal loss 将 DHN 训练作为一个二元分类任务**[30]。 我们通过使用$w0 = n1/(n0 + n1)$加权显性零类来补偿类不平衡（在A^*中的零$n0$和$n1$之间）。 我们用$w1 = 1 - w0$加权一类。 我们通过计算加权精度（WA）来评估 DHN 的性能：

$$\mathrm{WA} = \frac{w_1 n_1^* + w_0 n_0^*}{w_1 n_1 + w_0 n_0}, \qquad (10)$$

其中n_1^*和n_0^*分别是正确和错误肯定的数目。 由于 DHN 的输出介于 0 和 1 之间，因此我们将输出阈值设为 0.5。 在这些条件下，图 2 中的网络的 WA 得分为 92.88%。 在补充材料中，我们提供（i）选择循环单元的消融研究，（ii）讨论替代架构，（iii）分析距离矩阵大小对匹配精度的影响，以及（iv） 我们通过实验评估 DHN 保留分配矩阵属性的程度。

DHN Usage. 一旦采用上述策略对 DHN 进行了训练，就固定了其权重：在深度跟踪器的训练期间，它们不会以任何方式更新。

5.2.Experimental Settings

我们通过评估在行人跟踪的多个数据集上使用提议的框架进行训练时，评估现有（深层）多对象跟踪器的性能，从而证明了提出的框架的实际意义。 我们对损耗术语和跟踪架构进行制表。 我们还评估了该框架对其他培训替代方案的影响。 最后，我们在 MOTChallenge 基准上建立了最新的技术得分。

Datasets and Evaluation Metrics. 我们使用 MOT15，MOT16 和 MOT17 数据集，这些数据集提供了在真实的室外和室内场景中捕获的拥挤的行人视频序列。 对于消融研究，我们将训练序列分为训练和验证。拆分的细节可以在补充材料中找到。除了标准的 MOTP 和 MOTA 措施[5]以外，还使用 IDF1 [41]措施来提高性能，定义为正确识别的检测物与地面真实物体和物体轨迹的平均数量之比。 我们还报告了大多数跟踪目标（MT）和大多数丢失目标（ML），它们定义为跟踪假说所覆盖的真实轨迹的比例分别超过其寿命的 80%和不足 20%。

Tracktor. Tracktor [3]是 Faster RCNN [40]对象检测器对 MOT 任务的改编。 它使用区域提议网络（RPN）和检测器的分类/回归头来（i）检测对象，并且（ii）使用包围盒回归头在连续帧中跟踪检测到的目标。 由于 Tracktor 的大多数部分都是可训练的，因此使此方法成为演示我们框架的好处的理想选择。 请注意，Tracktor 最初仅在 MOTChallenge 检测数据集中训练，并且仅在推理期间应用于视频序列。 在下文中，我们将在这种设置下经过培训的 Tracktor 称为 Vanilla Base Tracktor。 由于 DeepMOT，我们可以直接在视频序列上训练 Tracktor，针对标准 MOT 措施进行优化。 我们将此变体称为 DeepMOT BaseTracktor。

Tracktor+ReID. Vanilla Tracktor 没有轨道标识的概念。因此，[3]建议在推理过程中使用外部训练的 ReID 模块以降低 IDS 的风险。这个外部 ReID 模块是一个特征提取器网络，具有

在 MOTChallenge 视频序列上使用 tripletloss [44]训练的 ResNet-50 骨干。我们将此变体称为+ ReIDext。请注意，这在培训期间不会给 Tracktor 带来任何身份认同的概念。这意味着，惩罚 IDS 数量的 DeepMOT 损失将对最终性能产生重大影响。为此，我们建议使用轻量级的 ReID 头替换外部 ReID 模块，我们可以与 Tracktor 一起使用 DeepMOT 端对端训练。这反过来又使我们能够利用惩罚 IDS 的损耗项，并全面优化 CLEAR-MOT 度量的所有组件的性能。我们将此变体称为+ ReIDhead。它采用完全连接层的形式，其中 128 个单元已插入 Tracktor。在补充材料中，我们提供了有关如何将 ID 信息嵌入距离矩阵 D 的详细信息。

即使以前在[51]中已使用过这种网络头，也已使用三重态损耗对其进行了外部培训[44]。据我们所知，我们**是第一个通过直接优化整个网络以跟踪评估措施来优化外观模型**的方法。

MOT-by-SOT. 为了证明我们方法的通用性，我们通过利用两个现成的（可训练的）单对象跟踪器（SOT）：GOTURN [18]和 SiamRPN [29]，提出了另外两个简单的可训练基线来执行 MOT。 在推理过程中，我们基于对象检测来初始化和终止轨迹。 对于每个对象，SOT 将人的时间 t-1 处的参考图像和图像 t 中的搜索区域作为输入。 然后，根据此参考框和搜索区域，SOT 为每个对象独立回归边界框。

Track Management. 在所有情况下，我们都使用简单（不可训练）的轨道管理程序。我们（i）使用检测器响应来初始化区域中未被现有轨道覆盖的对象轨道（对于 Tracktor，可以是公共检测，也可以是 Faster RCNN 检测响应）； （ii）使用 SOT 或 Tracktor 回归头将轨迹从帧 t 退回到帧 t + 1，并且（iii）终止与检测不重叠的轨迹（SOT 基线）或调用 Tracktor 的分类头，以信号是否 轨道覆盖了一个反对者或没有覆盖。 作为直接终止的替代方法，我们可以将轨道设置为 K 帧不可见。

5.3.Results and Discussion

	Method	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
Van.	Base	59.97	89.50	70.84	35.13	27.66	276	31827	326
	+ReIDext	**60.20**	**89.50**	**71.15**	**35.13**	**27.80**	**276**	**31827**	**152**
DeepMOT	Base	60.43	91.82	71.44	35.41	27.25	218	31545	309
	+ReIDext	60.62	91.82	71.66	35.41	27.39	218	31545	149
	+ReIDhead	**60.66**	**91.82**	**72.32**	**35.41**	**27.25**	**218**	**31545**	**118**

Table 1. Impact of the different ReID strategies for the two training strategies on Tracktor's performance.

Beyond Bounding Box Regression.如图 1 所示，我们首先在验证集上建立 Vanilla Base Tracktor 性能，并将其与 DeepMOT Base Tracktor 进行比较。 该实验（i）验证了我们基于 DHN 提出的训练 pipline 可将梯度传递到网络并提高整体性能，并且（ii）确认直觉认为训练目标跟踪器使用与跟踪评估措施直接相关的损失具有积极意义影响。 请注意，对 IDS 的影响微乎其微，这乍看之下可能是令人惊讶的，因为我们提出的损失除了 FP，FN 和边界框未对准之外，还会对 IDS 造成不利影响。