Improving Multiple Object Tracking with Single Object Tracking

Linyu Zheng[1,2], Ming Tang[1], Yingying Chen[1,2,3], Guibo Zhu[1,2], Jinqiao Wang[1,2,3], Hanqing Lu[1,2]
[1] National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] ObjectEye Inc., Beijing, China
{linyu.zheng,tangm,yingying.chen,gbzhu,jqwang,luhq}@nlpr.ia.ac.cn

Abstract

Despite considerable similarities between multiple object tracking (MOT) and single object tracking (SOT) tasks, modern MOT methods have not benefited from the development of SOT ones to achieve satisfactory performance. The major reason for this situation is that it is inappropriate and inefficient to apply multiple SOT models directly to the MOT task, although advanced SOT methods are of the strong discriminative power and can run at fast speeds.

In this paper, we propose a novel and end-to-end trainable MOT architecture that extends CenterNet by adding an SOT branch for tracking objects in parallel with the existing branch for object detection, allowing the MOT task to benefit from the strong discriminative power of SOT methods in an effective and efficient way. Unlike most existing SOT methods which learn to distinguish the target object from its local backgrounds*, the added SOT branch trains a separate SOT model per target online to distinguish the target from its* surrounding targets*, assigning SOT models the novel discrimination. Moreover, similar to the detection branch, the SOT branch treats objects as points, making its online learning efficient even if multiple targets are processed simultaneously. Without tricks, the proposed tracker achieves MOTAs of 0.710 and 0.686, IDF1s of 0.719 and 0.714, on MOT17 and MOT20 benchmarks, respectively, while running at 16 FPS on MOT17.*

1. Introduction

Multiple object tracking (MOT), which aims to estimate trajectories of multiple target objects in a video sequence, is a long-standing problem with many applications in mobile robotics, autonomous driving, and video surveillance analyses [26]. This problem is challenging because a successful method needs to not only detect the objects of interest accurately in each frame, but also associate them throughout the video. Moreover, fast running speeds are always desired.

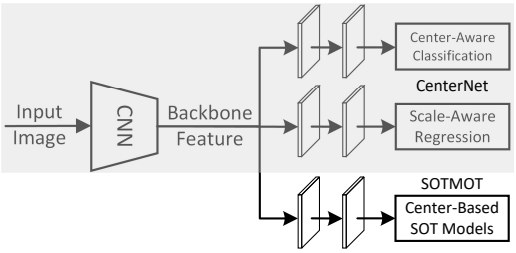

Figure 1: The architecture sketch of the proposed SOTMOT which extends CenterNet by adding a branch for tracking objects. The whole network can be trained in an end-to-end manner.

In recent years, many state-of-the-art methods [38, 42, 25] address the MOT problem by exploiting two modules: *detection* module to locate the objects of interest by bounding boxes in each frame and *Re-ID* one to associate each object to one of the existing trajectories. The latest ones, JDE [36] and FairMOT [45], integrate the Re-ID module into a single-shot detector and allow the above two modules to be learned in a shared model, achieving high accuracy and fast running speed simultaneously. Despite competitive performance, it seems that they encounter bottlenecks in robust object association, especially in crowded scenes. For instance, the top method, FairMOTv2, still performs relatively poorly on MOT20 [12], 0.673 in IDF1. Therefore, it may not be the only choice for MOT to associate same objects in different frames with the Re-ID technique.

It is well known that there are considerable similarities between MOT and single object tracking (SOT) tasks. They both are temporal problem and aim to estimate the trajectories of target objects in videos under the challenges of distracters, occlusions, and so on. In fact, it is no doubt that a multiple object tracker can be realized with multiple single ones [9, 52]. On the other hand, although the key issue of MOT is usually considered as the object association while that of SOT not, we argue that if the class of the target were known and a detector could provide high-recall proposals in the search region of the target, SOT would also be treated

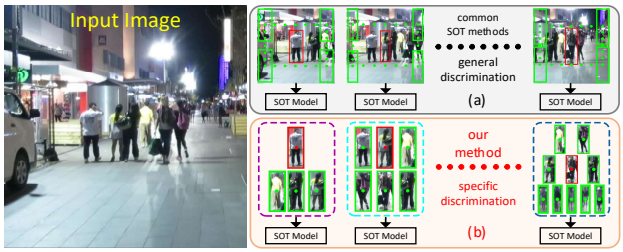

Figure 2: Illustration of the samples used to train SOT models in most existing SOT methods (a) and our method (SOT branch) (b). The former needs to deal with a large amount of backgrounds, whereas ours only exploits a small number of foregrounds.

as the problem of the association of proposals. Therefore, some techniques developed in SOT can be applied to MOT for the robust association.

Recently, discriminative model based methods [11, 4, 46, 47] have shown top performance in the field of SOT. Nevertheless, if we treat the localization of each target object in MOT as an SOT task and apply an SOT model directly to track it (Fig. 2a), the following two problems will arise. (i) Inappropriate discrimination. Most existing SOT methods train the discriminative model to distinguish the target from its local backgrounds, obtaining *general discrimination*. However, the MOT task focuses more on the ability of the discriminative model to distinguish the target from its surrounding targets, *i.e.*, *specific discrimination*, because most backgrounds can be filtered out by the detector. (ii) Although advanced SOT methods can run at a high speed (40 FPS), the time consumption is still unacceptable if dozens of targets are to be tracked at the same time in MOT.

To solve the above problems, in this paper, we propose a novel and end-to-end trainable MOT architecture, allowing the MOT task, more specifically, the object association in MOT, to benefit from the strong discriminative power of SOT methods in an effective and efficient way. As shown in Fig. 1, we extend the CenterNet detector [50] by adding an SOT branch for tracking objects in parallel with the existing branch for object detection. To obtain the specific discrimination, unlike most existing SOT methods, the added SOT branch trains a separate SOT model per target online to distinguish the target from its surrounding targets in the current frame (Fig. 2b). Afterwards, the trained SOT models perform the object association (*i.e.*, track the targets) in the next frame. This way, besides the stronger discriminative power for the MOT task, much more efficient online learning and tracking (*i.e.*, association) than the popular way of applying multiple SOT models directly to MOT are achieved, because foregrounds are much fewer than backgrounds. Moreover, to improve the efficiency further, similar to the detection branch, the SOT branch treats objects as points. Specifically, given the center of an object on a feature map, the object is represented by the feature vector at the center. Thereby, the SOT branch is able to run efficiently even though dozens of targets are present at the same time.

In offline training, the network receives a pair of images as its input. In the SOT branch, SOT models are trained with one image, and tested with the other, as done in [47]. Then, the CenterNet detector and the feature embeddings for ridge regression based single object tracker [47] are jointly trained to achieve the optimal feature embeddings for both detecting target objects and distinguishing a target object from its surrounding similar ones. In online tracking, different from JDE and FairMOT, ridge regression based SOT models, rather than Re-ID features, are used to associate the objects. Without tricks, the proposed tracker, SOTMOT, achieves MOTAs of 0.710 and 0.686, IDF1s of 0.719 and 0.714, on MOT17 [27] and MOT20 [12] benchmarks, respectively, while running at 16 FPS on MOT17 (including detection time). As far as we know, among all trackers that introduce an SOT method into MOT task, our SOTMOT is the first to achieve both state-of-the-art accuracy and fast running speed. We believe that our simple yet effective and efficient approach will benefit the future research on MOT, especially on the combination of SOT and MOT.

2. Related Work

MOT Methods. Most modern MOT methods [3, 38, 42, 25, 36, 45, 33, 13, 39] follow the tracking-by-detection paradigm. A detector [14, 30, 50] first locates all objects of interest in each frame with bounding boxes. Tracking is then performed by the object association between frames. SORT [3] tracks objects using Kalman filter and associates them based on the maximization of IoU between inter-frame bounding boxes. DeepSORT [38] augments the IoU-based association in SORT with deep appearance (Re-ID) features. Many recent methods focus on increasing the robustness of object association. POI [42] explores both high-performance detection and Re-ID features. LMP [33] leverages Re-ID and human pose features. RAN [13] proposes a novel association strategy based on RNN. Despite competitive accuracy, these trackers are difficult to run efficiently due to the separation of detection and Re-ID modules. In order to achieve high accuracy and fast running speed simultaneously, one-shot MOT methods are presented. JDE [36] and FairMOT [45] incorporate the Re-ID module into a single-shot detector, such that the whole network can output detections and Re-ID features simultaneously.

Similar to JDE and FairMOT, our SOTMOT is also an one-shot and tracking-by-detection based method. The difference between SOTMOT and them is that SOTMOT does not use any Re-ID module. Instead, the SOT module, which is online discriminatively trained, is developed to achieve the robust object association.

SOT Methods. In recent years, discriminative model based trackers [11, 46, 34, 4, 47] promote the development of

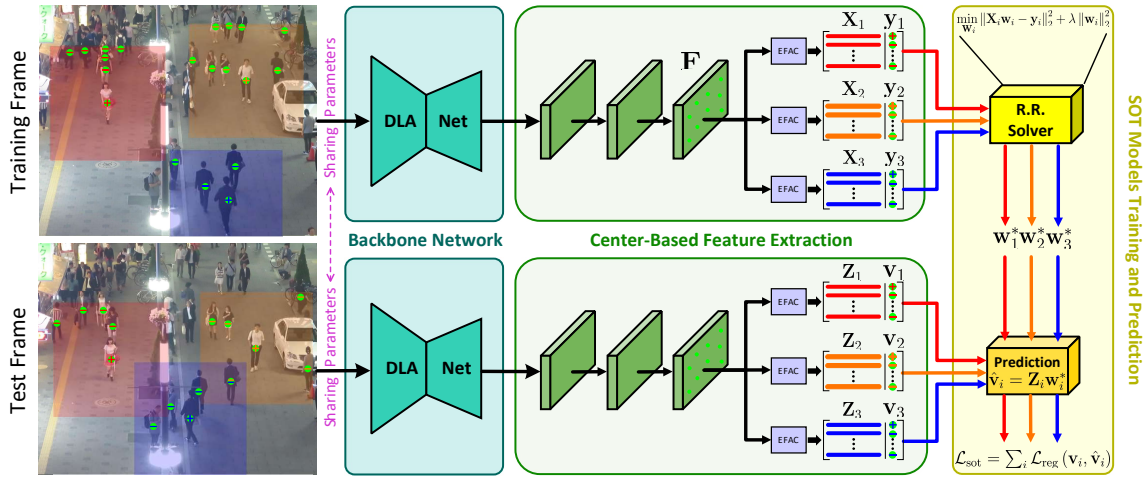

Figure 3: Architecture of SOT branch in our SOTMOT. EFAC is the extraction of feature vectors across the channel dimension at the center of target object. For each target of the training frame, 1) the center-based feature vectors of it and its neighbouring targets are extracted; 2) a discriminative SOT model, *i.e.*, ridge regression, is trained to distinguish it from its neighbours. Then, the trained SOT model predicts the labels of the target and its neighbouring targets in the test frame. Finally, the overall prediction loss of all trained SOT models is calculated.

SOT. These methods follow the basic pipeline of training discriminative model, *e.g.*, ridge regression, in the current frame to fit the training samples to their labels online and then evaluating the test samples in the next frame. Different from the previous methods [11, 46] which employ the features extracted via the ImageNet pre-trained CNNs to train discriminative models, modern methods [34, 4, 47] integrate the solver of discriminative model into the offline training of CNNs to learn the optimal feature embeddings for the SOT task.

MOT with SOT. There have been several methods [41, 8, 9, 52, 7] proposed to introduce an SOT method into the MOT task. These methods are mainly dedicated to studying how to combine the localization outputs of SOT and detection modules to deal with the challenge of tracking drift caused by occlusions and interactions. Particularly, the above two modules operate relatively independently and each SOT model needs to deal with a large amount of backgrounds as shown in Fig. 2a, resulting in low efficiency. In addition, it is difficult for the SOT modules in many of them to benefit from the end-to-end training of CNNs or to keep the objective of offline training consistent with that of online tracking, limiting the power of the SOT module. As a result, the latest of these trackers, UMA [41], can only run at 5 FPS on MOT17 with MOTA of 0.531 and IDF1 of 0.544, and others are all run below 1 FPS.

Our SOTMOT employs the DCFST [47] based SOT model. In order to apply DCFST to the MOT task in an effective and efficient way, different from the previous SOT-based MOT methods, each SOT model of SOTMOT only exploits the target object and its surrounding ones, which are regarded as foregrounds by the detector, to train discriminative model and locate the target. Additionally, the feature extraction of samples in DCFST and the previous SOT-based MOT methods is RoI-based, whereas it is center-based in SOTMOT for the high efficiency. To our best knowledge, among all trackers that introduce an SOT method into MOT task, SOTMOT is the first to achieve both state-of-the-art accuracy and fast running speed (0.710 in MOTA, 0.719 in IDF1, and 16 FPS on MOT17).

3. SOTMOT

Our method, SOTMOT, builds on the CenterNet detector [50]. CenterNet has three parallel branches appended to its backbone network. For each input image, the three branches generate the heatmap and offsets of object centers, and bounding box sizes, respectively. By adding an extra SOT branch in the CenterNet architecture, we construct the SOTMOT network. The SOT branch trains a separate SOT model per target in one frame and locates the targets in another frame (Fig. 3). Similar to the existing branches, the added SOT branch treats objects as points.

In the rest of this section, we will present the details of SOTMOT, with special focuses on the training of SOT models, offline training, and online inference of the SOT branch.

Backbone Network. We adopt a variant of DLA-34 [45] proposed by FairMOT as backbone for a good tradeoff between tracking accuracy and speed. Compared to the original DLA-34 [43], there are more skip connections between low-level and high-level features and convolution layers in all up-sampling modules are replaced by the deformable convolution [10] in the variant DLA-34. Denote the shape of an input RGB image as $3 \times H_{\text{img}} \times W_{\text{img}}$. Then, the output feature map of backbone has the shape of $C \times H \times W$, where $H = H_{\text{img}}/4$ and $W = W_{\text{img}}/4$.

CenterNet. In order to be self-contained, we briefly review

the CenterNet detector. For the sake of simplicity, we assume that all the objects of interest fall into one category, which is common in the field of MOT [22, 26, 12]. Tacking a single image as input, CenterNet produces a set of detections $\mathbb{D} = \{(\mathbf{c}_i + \triangle\mathbf{c}_i, \mathbf{s}_i)\}_{i=1}^{N}$. Specifically, the heatmap branch with the output shape of $1 \times H \times W$ identifies all objects through their centers $\mathbf{c}_i \in \Re^2$. In the heatmap, the response values at the locations corresponding to the centers of ground-truth objects are expected to be 1. The offset branch with the output shape of $2 \times H \times W$ refines the center of each object from down-sampling accuracy (located with the heatmap) to pixel-level one by estimating the offset $\triangle\mathbf{c}_i \in \Re^2$ between two accuracies, locating objects more precisely. The scale branch with the output shape of $2 \times H \times W$ estimates the scale (width and height) $\mathbf{s}_i \in \Re^2$ of bounding box for each object. These branches are all center-based, that is, the information of each object is encoded at its center location on the output map. Given an image with a set of annotated objects $\{(x_i, y_i, w_i, h_i)\}_{i=1}^{N}$, the heatmap branch uses the focal loss-based training objective $\mathcal{L}_{\text{heat}}$ [21], and the other two branches use the mean square error-based ones, denoted as $\mathcal{L}_{\text{off}}$ and $\mathcal{L}_{\text{size}}$, respectively.

3.1. SOT Branch

Center-Based Feature Extraction. Given the backbone feature map of an input image, we pass it through three convolutional layers to obtain the SOT feature map $\mathbf{F}$ of $C_{\text{sot}} \times H \times W$. The convolutional kernels are 3×3 with stride 1×1 and the convolutional layers are followed by BatchNorm and ReLU. Further, given the center $\mathbf{c} = \{x^c, y^c\}$ of an object on $\mathbf{F}$, the object is represented by the feature vector $\mathbf{x} \equiv \mathbf{F}(\mathbf{c})$ which is extracted from $\mathbf{F}$ at $\mathbf{c}$ without extra calculations, and $\mathbf{x} \in \Re^{C_{\text{sot}}}$. It is easy to see that the extraction of $\mathbf{x}$ is time-saving, even for dozens of objects.

SOT Models Training. Given a training image and the set of centers $\mathbb{N} = \{(x_i^c, y_i^c)\}_{i=1}^{N}$ of its target objects, the training sample matrix $\mathbf{X} \equiv \left[\mathbf{x}_1^\top; \cdots; \mathbf{x}_N^\top\right] \in \Re^{N \times C_{\text{sot}}}$ is constructed with the extracted feature vectors of all target objects. Further, a neighbourhood matrix $\mathbf{A} \in \{0, 1\}^{N \times N}$, which indicates whether any two centers in $\mathbb{N}$ are neighbouring, is constructed by

$$\mathbf{A}_{i,j} = \begin{cases} 1 & \text{if } \min\left(\left|x_i^c - x_j^c\right|, \left|y_i^c - y_j^c\right|\right) \leqslant r \\ 0 & \text{otherwise} \end{cases}, \quad (1)$$

where r is the threshold of distance.

For each target object $\mathbf{x}_i$, a neighbourhood sample matrix $\mathbf{X}_i$ along with its label vector $\mathbf{y}_i$ are constructed, where $\mathbf{X}_i$ is composed of the feature vectors of the targets whose centers are neighbours of (x_i^c, y_i^c), that is, $\{\mathbf{x}_j \mid \forall j : \mathbf{A}_{i,j} = 1\}$. All components of $\mathbf{y}_i$ are negative (0), except that the component which indicates the label of $\mathbf{x}_i$ is positive (1). Then, a ridge regression based discriminative model $\mathbf{w}_i^*$ is trained to distinguish the target $\mathbf{x}_i$ from its neighbouring targets. Specifically, we set

$$\min_{\mathbf{w}_i} \|\mathbf{X}_i\mathbf{w}_i - \mathbf{y}_i\|_2^2 + \lambda \|\mathbf{w}_i\|_2^2, \quad (2)$$

where λ is the regularization parameter. The optimization solution of Problem 2 can be expressed as

$$\mathbf{w}_i^* = \left(\mathbf{X}_i^\top \mathbf{X}_i + \lambda \mathbf{I}\right)^{-1} \mathbf{X}_i^\top \mathbf{y}_i. \quad (3)$$

It is worth mentioning that the number of rows of $\mathbf{X}_i$ depends on how many targets there are around the target $\mathbf{x}_i$, *i.e.*, $\sum_j \mathbf{A}_{i,j}$. No matter how $\sum_j \mathbf{A}_{i,j}$ is, $\mathbf{X}_i^\top \mathbf{X}_i$ and $\mathbf{X}_i^\top \mathbf{y}_i$ always belong to $\Re^{C_{\text{sot}} \times C_{\text{sot}}}$ and $\Re^{C_{\text{sot}} \times 1}$, respectively. Therefore, given $\left(\mathbf{X}_i^\top \mathbf{X}_i\right)$s and $\left(\mathbf{X}_i^\top \mathbf{y}_i\right)$s, multiple $\mathbf{w}_i^*$s can be solved simultaneously in a batch way.

Offline Training. As shown in Fig. 3, the proposed network receives a pair of RGB images, one for training and another for testing, in offline training, and is trained in the way of two-stream of sharing parameters [4, 47]. For the training image, $\{\mathbf{w}_i^*\}_{i=1}^{N}$ can be obtained with Eq.(3). For the test image, given the set of centers $\mathbb{M} = \left\{\left(x_j^c, y_j^c\right)\right\}_{j=1}^{M}$ of target objects, the test sample matrix $\mathbf{Z} = \left[\mathbf{z}_1^\top; \cdots; \mathbf{z}_M^\top\right] \in \Re^{M \times C_{\text{sot}}}$, neighbourhood sample matrices $\mathbf{Z}_j$s along with their ground-truth label vectors $\mathbf{v}_j$s can also be obtained through the way similar to the above. Afterwards, we rearrange $\{\mathbf{w}_i^*\}_{i=1}^{N}$ and $\{\mathbf{Z}_j\}_{j=1}^{M}$ into $\{\mathbf{w}_1^*, ..., \mathbf{w}_k^*, ..., \mathbf{w}_N^*\}$ and $\{\mathbf{Z}_1, ..., \mathbf{Z}_k, ..., \mathbf{Z}_M\}$, so that for each of the first k pairs of $(\mathbf{w}_i^*, \mathbf{Z}_i)$, the positive sample $\mathbf{x}_i$ which is used to generate $\mathbf{w}_i^*$ and the $\mathbf{z}_i$ which is the only positive sample in $\mathbf{Z}_i$ represent the identical target object. Finally, the training loss is calculated with

$$\mathcal{L}_{\text{sot}} = \sum_{i=1}^{k} \mathcal{L}_{\text{reg}}\left(\mathbf{v}_i, \hat{\mathbf{v}}_i\right), \quad (4)$$

where $\mathcal{L}_{\text{reg}}(\cdot, \cdot)$ is the shrinkage loss proposed in [47] for alleviating the imbalance of samples and is written as

$$\mathcal{L}_{\text{reg}}\left(\mathbf{v}, \hat{\mathbf{v}}\right) = \left\| \frac{\exp\left(\mathbf{v}\right) \odot \left(\mathbf{v} - \hat{\mathbf{v}}\right)}{1 + \exp\left(a \cdot \left(c - \left|\mathbf{v} - \hat{\mathbf{v}}\right|\right)\right)} \right\|_2^2, \quad (5)$$

$\mathbf{v}_i$ is the label vector of $\mathbf{Z}_i$, and $\hat{\mathbf{v}}_i = \mathbf{Z}_i\mathbf{w}_i^*$ is its prediction.

Because ridge regression model is differentiable and its solver (Eq. 3) can be integrated into the offline training of CNNs [2, 47], the SOT branch can be trained in an end-to-end way following the above, learning the optimal feature embeddings for the ridge regression model based single object tracker which tracks the target object by distinguishing it from its surrounding similar ones.

Online Inference. The whole online tracking scheme of our SOTMOT is based on DeepSORT and FairMOT. In particular, the SOT branch in the scheme is responsible for initializing new trajectories or updating the existing ones by

training SOT models online, and associating the detected target objects in the current frame to the existing trajectories by calculating matching scores.

Without loss of generality, given frame t and the set of centers $\mathbb{N}^t$ of the targets output by the CenterNet detector, the sample matrix $\mathbf{X}^t\,(\mathbf{Z}^t) \in \Re^{N\times C_{\text{sot}}}$ [1] is constructed as described above ($\mathbf{X}^t$ is the $\mathbf{X}$ in frame t). Afterwards, we always need to perform the association between $\mathbb{N}^t$ and the set of center locations $\mathbb{M}^t$ of the existing trajectories $\{\mathcal{T}_i\}_{i=1}^M$ first, then update each $\mathcal{T}_i$ with the new detections. Here, each $\mathcal{T}_i$ contains at least the estimated center of the target in the current frame, *i.e.*, the i-th element in $\mathbb{M}^t$, a sample pool $\mathcal{X}_i$, and an SOT model $\mathbf{w}_i^*$.

For the association, the neighbourhood matrix $\mathbf{B}^t \in \{0,1\}^{M\times N}$ between $\mathbb{M}^t$ and $\mathbb{N}^t$ is first constructed in a manner similar to Eq. 1, where $\mathbf{B}_{i,j}^t$ indicates whether the i-th center in $\mathbb{M}^t$ and the j-th center in $\mathbb{N}^t$ are neighbouring. Then, the neighbourhood sample matrices $\mathbf{Z}_i^t$s are constructed based on $\mathbf{Z}^t$ and $\mathbf{B}_{i,j}^t$, where for each i, $\mathbf{Z}_i^t$ is composed of the feature vectors of the targets in $\mathbb{N}^t$ whose centers are neighbours of the center of $\mathcal{T}_i$, that is, $\left\{\mathbf{z}_j^t \mid \forall j : \mathbf{B}_{i,j}^t = 1\right\}$. Finally, the matching scores between each $\mathcal{T}_i$ and its neighbouring targets are calculated using the SOT model $\mathbf{w}_i^*$ of $\mathcal{T}_i$, *i.e.*, $\hat{\mathbf{v}}_i^t = \mathbf{Z}_i^t\mathbf{w}_i^*$.

For the online update, $\mathbf{A}^t \in \{0,1\}^{N\times N}$, $\mathbf{X}_i^t$s along with $\mathbf{y}_i^t$s are first constructed based on $\mathbb{N}^t$ and $\mathbf{X}^t$ in the aforementioned way ($\mathbf{X}_i^t$ is the $\mathbf{X}_i$ in frame t). After associating with the existing trajectories, any target object $\mathbf{x}_i^t$ in $\mathbf{X}^t$ will fall into one of the following two situations:

(S1) If $\mathbf{x}_i^t$ is a new target which does not associate to any of the existing trajectories, a new trajectory $\mathcal{T}_k$ is established by initializing its sample pool $\mathcal{X}_k$ with the batch of training samples $(\mathbf{X}_i^t, \mathbf{y}_i^t)$ and training its SOT model $\mathbf{w}_k^*$ with Eq. 3.

Here, in order to facilitate a unified expression for both training and update formulas (Eq. 3 and subsequent Eq. 6), when we add the $(\mathbf{X}_i^t, \mathbf{y}_i^t)$ into a sample pool $\mathcal{X}_k$, its subscript will be changed to $(\mathbf{X}_k^t, \mathbf{y}_k^t)$. In other words, in the sample pool, $(\mathbf{X}_k^t, \mathbf{y}_k^t)$ is the batch of training samples collected in frame t for trajectory $\mathcal{T}_k$.

(S2) If $\mathbf{x}_i^t$ is associated with one of the existing trajectories $\mathcal{T}_k$ whose sample pool $\mathcal{X}_k = \{(\mathbf{X}_k^p, \mathbf{y}_k^p)\}_{p=s}^{t-1}$, we add $(\mathbf{X}_i^t, \mathbf{y}_i^t)$ into $\mathcal{X}_k$ and update the SOT model by solving

$$\min_{\mathbf{w}_k} \sum_{p=s}^{t} \beta^p \left\|\mathbf{X}_k^p\mathbf{w}_k - \mathbf{y}_k^p\right\|_2^2 + \lambda \left\|\mathbf{w}_k\right\|_2^2, \tag{6}$$

where s is the start frame of the trajectory $\mathcal{T}_k$ and β^p is the weight of the training samples from frame p with $\beta^s = (1-\delta)^{t-s}$, $\sum_{p=s}^{t}\beta^p = 1$, and $\beta^{p-1}/\beta^p = 1-\delta \quad (p > s+1)$. In fact, Eq. 6 with such β^ps is a standard expression of moving-average based model update which is commonly used in the field of SOT [48]. The optimization solution of Problem 6 can be expressed as

$$\mathbf{w}_k^* = \left[\sum_{p=s}^{t}\beta^p\left(\mathbf{X}_k^p\right)^\top\mathbf{X}_k^p + \lambda\mathbf{I}\right]^{-1}\left[\sum_{p=s}^{t}\beta^p\left(\mathbf{X}_k^p\right)^\top\mathbf{y}_k^p\right]. \tag{7}$$

3.2. Whole Schemes

Based on the above, we present the whole offline training and online tracking schemes of the proposed SOTMOT.

Offline Training. Receiving a pair of RGB images as input, our whole network, including backbone network, detection branch (the three branches of CenterNet detector), and the proposed SOT branch, can be jointly trained in an end-to-end way. For the SOT branch, one of the pair of images is used as a training image and the other as a test image. They produce the SOT loss together (Sec. 3.1). For the detection branch, there is no difference between the roles of the two images. They produce their own detection losses independently. To deal with the task of multi-task learning, detection and tracking, we adopt the learning scheme proposed in [18] for automatic loss balancing. Specifically, the total offline training loss is formulated as

$$\mathcal{L}_{\text{total}} = \frac{1}{2}\left(\frac{1}{e^{w_1}}\mathcal{L}_{\text{det}} + \frac{1}{e^{w_2}}\mathcal{L}_{\text{sot}} + w_1 + w_2\right), \tag{8}$$

where $\mathcal{L}_{\text{det}} = \mathcal{L}_{\text{det}}^1 + \mathcal{L}_{\text{det}}^2$ contains the detection losses of the two input images and

$$\mathcal{L}_{\text{det}}^i = \mathcal{L}_{\text{heat}}^i + \mathcal{L}_{\text{off}}^i + 0.1\mathcal{L}_{\text{size}}^i \quad i \in \{1,2\} \tag{9}$$

where the fixed weights in $\mathcal{L}_{\text{det}}^i$ are suggested in CenterNet.

Online Tracking. The online tracking scheme of our SOTMOT is based on DeepSORT [38] and FairMOT [45]. In particular, Kalman Filter [37] is used to predict the locations of the existing trajectories in the current frame, and motion-based information and IoU of bounding boxes are employed to assist object association. Since our SOTMOT is obviously different from the previous SOT-based MOT methods in terms of both the construction of SOT model and the problem to be solved by introducing SOT methods into the MOT task, we outline the main procedure of SOTMOT in Algorithm 1 so that readers can have a more detailed understanding of the role of the proposed SOT branch in the scheme. For simplicity, some operations which are common in the filed of MOT or completely unrelated to our SOT-based association are not included. We suggest readers referring to the released codes of DeepSORT and FairMOT, or our upcoming one, for more details.

[1] Different from offline training where the training and test images are different ones, in online inference, for any input image, we need to perform both online training for SOT model update and test for object association, based on the detected target objects. Therefore, $\mathbf{X}^t = \left[\mathbf{x}_1^{t\top}; \cdots; \mathbf{x}_N^{t\top}\right]$ and $\mathbf{Z}^t = \left[\mathbf{z}_1^{t\top}; \cdots; \mathbf{z}_N^{t\top}\right]$ are identical in processing frame t.

Algorithm 1 Online Tracking Scheme of SOTMOT.

Inputs:

1. SOT feature map $\mathbf{F} \in \Re^{C_{\text{sot}} \times H \times W}$ of the current frame;
2. The set of detections $\mathbb{D} = \{\mathbf{c}_i, \mathbf{s}_i\}_{i=1}^{N}$ output by the CenterNet detector in the current frame. $\mathbb{N} = \{\mathbf{c}_i\}_{i=1}^{N}$;
3. The set of existing trajectories $\mathbb{T} = \{\mathcal{T}_i\}_{i=1}^{M}$ in which each $\mathcal{T}_i$ contains at least the Kalman state of the target location, a sample pool $\mathcal{X}_i$, and an SOT model $\mathbf{w}_i^*$;

Main Processes: (Test: 1 - 8, Train + Update: 9 - 12)

1: Predict the current target locations $\{\hat{\mathbf{c}}_i, \hat{\mathbf{s}}_i\}$s of all $\mathcal{T}_i$s in $\mathbb{T}$ with Kalman Filter [37]. $\mathbb{M} = \{\hat{\mathbf{c}}_i\}_{i=1}^{M}$;
2: Construct the neighbourhood matrix $\mathbf{B} \in \{0,1\}^{M \times N}$ between $\mathbb{M}$ and $\mathbb{N}$ in a manner similar to Eq. 1;
3: Construct the sample matrix $\mathbf{X}$ ($\mathbf{Z}$) based on $\mathbb{N}$ and $\mathbf{F}$;
4: Construct the neighbourhood sample matrices $\mathbf{Z}_i$s based on $\mathbf{Z}$ and $\mathbf{B}$;
5: Calculate the matching scores between each $\mathcal{T}_i$ in $\mathbb{T}$ and its neighbouring targets in $\mathbb{N}$, *i.e.*, $\hat{\mathbf{v}}_i = \mathbf{Z}_i \mathbf{w}_i^* \ \ \forall i$;
6: Fuse the motion metric into each $\hat{\mathbf{v}}_i$, as done in DeepSORT;
7: Perform Hungarian matching between $\mathbb{T}$ and $\mathbb{D}$ based on $\hat{\mathbf{v}}_i$s, then output the matched and unmatched trajectory sets, $\mathbb{P}$ and $\mathbb{Q}$, and the unmatched detection set $\mathbb{K}$;
8: Perform Hungarian matching between $\mathbb{Q}$ and $\mathbb{K}$ based on the IoUs between target locations in them, then update $\mathbb{P}$, $\mathbb{Q}$, $\mathbb{K}$;
9: Construct the neighbourhood matrix $\mathbf{A} \in \{0,1\}^{N \times N}$ of $\mathbb{N}$;
10: Construct the neighbourhood sample matrix $\mathbf{X}_i$s along with their label vectors $\mathbf{y}_i$s based on $\mathbf{X}$ and $\mathbf{A}$;
11: Update each $\mathcal{T}_i$ in $\mathbb{P}$ as mentioned in (S2) and update its Kalman state with its new location;
12: Initialize new $\mathcal{T}_i$s for the detections in $\mathbb{K}$ as mentioned in (S1);
13: Deal with the $\mathcal{T}_i$s in $\mathbb{Q}$ as in FairMOT;
14: Output the new set of trajectories composed of the output ones from Step 11, 12, and 13 after some common post-processing.

4. Discussions

SOT Model vs. Re-ID Model. As mentioned above, the main difference between our SOTMOT and FairMOT is their association models, SOT and Re-ID ones. Formally, the association model used to distinguish the target object x_i from the others $\{x_j : \forall j \neq i\}$ in Re-ID model and our constructed SOT model are $\mathbf{x}_i = \phi_{\text{reid}}(x_i)$ and

$$\mathbf{w}_i^* = f\left(\left(\phi_{\text{sot}}(x_i), 1\right), \{\left(\phi_{\text{sot}}(x_j), 0\right)\}_{j \neq i}\right),$$

respectively, where $\phi_{\text{reid}}(\cdot)$ and $\phi_{\text{sot}}(\cdot)$ are feature extractors, and $f(\cdot,\cdot)$ is a solver of discriminative SOT model.

The purpose of the Re-ID task is to learn a feature embedding $\phi_{\text{reid}}(\cdot)$ with which the distance between objects of the same label is smaller than those of different labels, distinguishing different instances of the same object from all other different objects. Given a video sequence in online tracking, the generalization of the learned $\phi_{\text{reid}}(\cdot)$ will be sufficient for some of the target objects, whereas may be insufficient for others because the generalization of a Re-ID system is always limited due to the limited offline training data. The larger the number of target objects appearing in an identical frame, the higher the possibility of failure in generalization and tracking. This is the main reason why FairMOT performs well on MOT17 (sparse scene) but poorly on MOT20 (crowded scene).

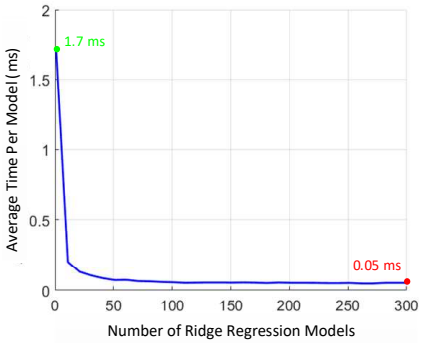

Figure 4: The average time consumption per model when training multiple 128-dimensional ridge regression models in a batch way.

Different from the Re-ID model, in our approach, given a video sequence, each SOT model learns to distinguish a *unique target* from its surrounding targets in an online adaptive way, treating a few objects only in each frame. On the contrast, the Re-ID model has to deal with several of countless objects by only resorting to the generalization ability obtained through offline learning. Obviously, the challenging of generalization Re-ID model faces is much larger than SOT model does in each frame. Consider that the motion of objects is continuous in a video, the local discrimination of SOT model is almost enough in most videos. According to the above, the possibility of generalization failure in SOT model is much lower than that in Re-ID one. Therefore, our SOTMOT performs well on not only MOT17 but also MOT20. It seems that the SOT-based association model in our novel method is more robust for the MOT task than the current Re-ID-based ones.

Efficiency of Training SOT Models. Now that the proposed SOT branch needs to train a separate ridge regression (RR) model per target online, and the time cost of training a RR model is usually not negligible, naturally there is the concern on the efficiency of the above online training process when dozens of targets are present at the same time. Fortunately, as our mentioned above, multiple RR models can be trained simultaneously in a batch way. We show that this property can greatly alleviate the above concern by exploiting the parallel computation of GPU. Fig. 4 shows how the average time consumption varies with the increase of the number of RR models. It is seen that adding a target, *i.e.*, adding a RR model, adds very little to the overall training time of all RR models when the number of targets beyond 20. This characteristic allows SOTMOT to track dozens of targets at the same time efficiently in principle.

5. Experiments

Our SOTMOT is implemented in Python using PyTorch. On a single RTX 2080Ti GPU, it achieves the average running speed of 16 FPS (including the time consumption of both detection and tracking) on MOT17 [27] without deliberate optimization. Code will be made available.

5.1. Implementation Details

Training Dataset. As suggested in JDE [36], performing experiments on small datasets may lead to biased results and conclusions may not hold when applying the same algorithm to large-scale datasets. Both modern tracking-by-detection based one-shot MOT methods, JDE and our baseline FairMOT [45], use the large-scale training set made by JDE, denoted as JDE dataset, to train their networks. For a fair comparison with them, we also employ JDE dataset in offline training. During network training, each pair of training and test images is sampled from a video snippet within the nearest 100 frames or from still images (in this case, the SOT branch is not trained). The detailed training pipelines for video images and still images are shown in the supplementary material, respectively.

Training Setting. We use the model pre-trained on COCO [24] to initialize the weights of backbone network and fine-tune them during offline training. The weights of our head networks are randomly initialized with zero-mean Gaussian distributions. We train the whole network for 50 epochs with 3.6k iterations per epoch and 12 pairs of images per batch. The ADAM [20] optimizer is used with initial learning rate of 10^{-4}, using a factor 0.1 decay at 30-th epoch.

Parameters Setting. The parameters in our method are set in a common way. The size of input image is $H_{\text{img}} \times W_{\text{img}} = 736 \times 1280$. Thereby, $H \times W = 184 \times 320$. The r in Eq. 1 is set to 75. The C_{sot} is set to 128. The λ in Eq. 2 is set to 0.1. The a and c in Eq. 5 are set to 10 and 0.2, respectively, as in DCFST [47]. The δ of β in Eq. 6 is set to 0.1.

5.2. Datasets and Evaluation Metrics

We demonstrate the tracking performance of SOTMOT on three public benchmarks, MOT16 [26], MOT17 [27], and MOT20 [12], focusing on pedestrian tracking, and compare it against many state-of-the-art methods. Note that, different from MOT16 and MOT17, MOT20 pays more attention to the MOT in crowded scenes with higher requirement for the robustness of object association.

We use the official evaluation metrics in the MOT challenge where Multiple Object Tracking Accuracy (MOTA) and ID F1 Score (IDF1) are mainly reported which quantify two of the main aspects a MOT method, namely, object coverage and identity. All results of our method and others on the test sets of the above benchmarks are directly obtained from the official evaluation server of MOT challenge.

Table 1: Ablation studies on the validation set of MOT17 benchmark. (a) Comparisons of different backbone networks. (b) Comparisons of SOT models with different discriminant attributes.

(a) Backbone Networks.

Backbone	MOTA↑	IDF1↑	FPS↑
ResNet34-FPN	66.3	70.9	21
DLA-34	70.2	73.5	16

(b) Discriminant Attributes.

Attribute	IDF1↑	ID Sw.↓	FPS↑
General Discri.	71.2	621	9
Specific Discri.	73.5	504	16

5.3. Ablation Studies

Backbone Network. Any deep convolutional networks that provide multi-scale features can be used in our framework. In particular, we compare the tracking performance of the DLA-34 network chosen in this work and the other classic one, ResNet34-FPN [23], in our SOTMOT. Table 1a shows the results. It is seen that the running speed of using ResNet34-FPN in SOTMOT is slightly faster than that of using DLA-34, whereas the accuracy of using DLA-34 is obviously higher than that of using ResNet34-FPN.

Discriminant Attribute. To demonstrate our core claim, that is to achieve the robust and efficient object association, the discriminative SOT models in our SOTMOT should be trained to obtain the *specific discrimination* (SD) (Fig. 2(b)) rather than the *general discrimination* (GD) (Fig. 2(a)), we conduct experiments to show the performance gap between training each SOT model with the *surrounding targets* (SOT-SD) and with the *local backgrounds* (SOT-GD) of the target in our approach. Table 1b shows the results. It is not surprising that SOT-SD outperforms SOT-GD in both tracking accuracy and speed. To this, we give the following two analyses: (1) The main reason for the FPS of SOT-GD being much lower than that of SOT-SD is that the number of training samples for SOT models in SOT-GD is far more than that in SOT-SD, leading to a significant drop in the running speed naturally. (2) The detector has filtered out most backgrounds before the SOT-based association. Therefore, the main task of each SOT model is to distinguish the target from its surrounding targets. This is exactly what SOT-SD does. Compared to SOT-SD, SOT-GD introduces a large amount of backgrounds into the training of SOT models, weakening the discriminative power it actually needs.

5.4. State-of-the-art Comparisons

Private Detections. Since our SOTMOT employs the detections output by the trained CenterNet detector in online tracking, we first compare it with the recent methods which also employ private detections. Table 2 shows the results. It is seen that: 1) SOTMOT achieves a good balance between tracking accuracy and speed. 2) SOTMOT outperforms the strong baseline method FairMOT [2] on all

[2]FairMOTv2 is an improved version of FairMOT by introducing self-supervised learning and the extra large-scale dataset, CrowdHuman [32] which is 1.7 times larger than the JDE dataset in terms of bounding box

Table 2: State-of-the-art comparisons on the test sets of MOT16, MOT17, and MOT20 under the "private detections". The FPS considers the total running time (detection and tracking) of a method. The best three results of MOTA, IDF1, and FPS are shown in red, blue, and orange, respectively. The proposed SOTMOT achieves competitive results with other trackers. It outperforms its baseline, FairMOT, with large margins in terms of MOTA and IDF1.

	Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	ID Sw.↓	FPS↑
MOT16	EAMTT [31]	52.5	53.3	19.0	34.9	4407	81223	910	<5.5
	SORT [3]	59.8	53.8	25.4	22.7	8698	63245	1423	<8.6
	DeepSORTv2 [38]	61.4	62.2	32.8	18.2	12852	56668	781	<8.0
	RAR16 [13]	63.0	63.8	39.9	22.1	13663	53248	482	<1.5
	VMaxx [35]	62.6	49.2	32.7	21.1	10604	56182	1389	<3.9
	TubeTK [28]	64.0	59.4	33.5	19.4	10962	53626	1117	1.0
	JDE [36]	64.4	55.8	35.4	20.0	-	-	1544	22.2
	TAP [51]	64.8	73.5	38.5	21.6	12980	50635	571	<8.0
	CNNMTT [25]	65.2	62.2	32.4	21.3	6578	55896	946	<5.3
	POI [42]	66.1	65.1	34.0	20.8	5061	55914	805	<5.0
	CTracker [29]	67.6	57.2	32.9	23.1	8934	48305	1897	6.8
	LMP [33]	71.0	70.1	46.9	21.9	7880	44564	434	0.5
	FairMOT [45]	69.3	72.3	40.3	16.7	13501	41653	815	25.9
	FairMOTv2 [45]	74.9	72.8	44.7	15.9	10163	34484	1074	25.9
	SOTMOT (ours)	72.1	72.3	44.0	13.2	14344	34784	1681	16.0
MOT17	SST [6]	52.4	49.5	21.4	30.7	-	-	8431	<3.9
	TubeTK [28]	63.0	58.6	31.2	19.9	27060	177483	4137	3.0
	CTracker [29]	66.6	57.4	32.2	24.2	22284	160491	5529	6.8
	CenterTrack [49]	67.8	64.7	34.6	24.6	18498	160332	3039	17.5
	FairMOT [45]	67.5	69.8	37.7	20.8	-	-	2868	25.9
	FairMOTv2 [45]	73.7	72.3	43.2	17.3	27507	117477	3303	25.9
	SOTMOT (ours)	71.0	71.9	42.7	15.3	39537	118983	5184	16.0
MOT20	FairMOT [45]	58.7	63.7	66.3	8.5	-	-	6013	13.2
	FairMOTv2 [45]	61.8	67.3	68.8	7.6	103440	88901	5243	13.2
	SOTMOT (ours)	68.6	71.4	64.9	9.7	57064	101154	4209	8.5

the three benchmarks in terms of tracking accuracy, *i.e.*, MOTA and IDF. This confirms the robustness of our proposed SOT model-based object association method, since the main difference between SOTMOT and FairMOT lies on their association models. 3) Although FairMOTv2 takes advantage of self-supervised learning and much more training data, SOTMOT still outperforms it with large margins in tracking accuracy on MOT20. This further confirms that the proposed SOT-based association method is also robust even for MOT in crowded scenes which is challenging for most modern methods. 4) The ID switches of SOTMOT on MOT16 and MOT17 are remarkably higher than those of FairMOT and FairMOTv2, whereas lower than those of FairMOT and FairMOTv2 on MOT20. We analyse the reasons for this phenomenon in the supplementary material. 5) The running speed of SOTMOT is slightly lower than those of some modern one-shot MOT methods, FairMOT, CenterTrack, and JDE. We believe that this issue can be addressed by exploiting more efficient backbone networks or SOT models into the proposed framework in the near future.

Public Detections. We also evaluate our SOTMOT with the public detections provided by the official MOT challenge and compare it with many state-of-the-art MOT methods.

annotation, in offline training. Our SOTMOT does not take advantage of the above improvements. Therefore, FairMOT rather than FairMOTv2 is the baseline method of SOTMOT.

Table 3: State-of-the-art comparisons on the test set of MOT17 under the "public detections". +D means adding the detection time. The proposed SOTMOT achieves a good balance between tracking accuracy and speed.

	Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	ID Sw.↓	FPS↑
SOT-Based	DMAN [52]	48.2	55.7	19.3	38.3	26218	263608	2194	<0.3
	FAMNet [8]	52.0	48.7	19.1	33.4	14138	253616	3072	<0.6
	LSST [15]	52.7	57.9	17.9	36.6	22512	241936	2167	<1.7
	UMA [41]	53.1	54.4	21.5	31.8	22893	239534	2251	<4.2
Public Detections	Tracktorv2 [1]	56.3	55.1	21.1	35.3	8866	235449	3763	<1.4
	DeepMOT [40]	53.7	53.8	19.4	36.6	11731	247447	1947	<4.1
	TT17 [44]	54.9	63.1	24.4	38.1	20236	233295	1088	<2.3
	MPNTrack [5]	58.8	61.7	28.8	33.5	17413	213594	1185	<5.2
	STRN [39]	50.9	56.5	20.1	37.0	27532	246924	2593	<8.9
	jCC [19]	51.2	54.5	20.9	37.0	25937	247822	1802	<1.7
	LifT [16]	60.5	65.6	27.0	33.6	14966	206619	1189	<0.5
	UnsupTrack [17]	61.7	58.1	27.2	32.4	16872	197632	1864	<1.9
	CenterTrack [49]	61.5	59.6	26.4	31.9	14076	200672	2583	17.5+D
	SOTMOT (ours)	**62.8**	**67.4**	24.4	33.0	6556	201319	2017	16.0+D

In this experiment, we follow the public-detection configuration used in CenterTrack [49] to deal with the bounding boxes output by the CenterNet detector. The compared methods are divided into two categories according to whether they exploit SOT models or not. Table 3 shows the results. It is seen that SOTMOT surpasses all SOT-based MOT methods with large margins in both tracking accuracy and speed. This confirms that the way SOTMOT exploits SOT models is more effective and efficient than the previous methods' did. Moreover, SOTMOT outperforms CenterTrack in IDF1 with a large margin, although their MOTAs are relatively close. This confirms that SOTMOT is more robust than CenterTrack on the object association.

6. Conclusion

Through extending the CenterNet detector with an SOT branch, a novel and state-of-the-art multiple object tracker SOTMOT is proposed. Instead of the commonly used Re-ID models, SOT model is introduced into the MOT task to achieve the robust object association. Moreover, benefiting from the one-shot framework and center-based feature extraction, SOTMOT is able to track dozens of targets at the same time in a fast speed. Experiments demonstrate that SOTMOT can track targets robustly and efficiently even in crowded scenes. We thus believe that our simple yet effective and efficient approach will benefit the future research on MOT, especially on combination of SOT and MOT.

Acknowledgements. This work was supported by the Research and Development Projects in the Key Areas of Guangdong Province (No. 2020B010165001). This work was also supported by National Natural Science Foundation of China under Grants 61772527, 61976210, 61806200, 61876086, 62076235, 62002356, 62006230, 62002357, and 51975044, and by the Technology Cooperation Project of Application Innovate Laboratory, Huawei Technologies Co., Ltd. (FA2018111061-2019SOW05).

References

[1] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE international conference on computer vision*, pages 941–951, 2019. 8

[2] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019. 4

[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2, 8

[4] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019. 2, 3, 4

[5] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020. 8

[6] Long Chen, Haizhou Ai, Chong Shang, Zijie Zhuang, and Bo Bai. Online multi-object tracking with convolutional neural networks. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 645–649. IEEE, 2017. 8

[7] Peng Chu, Heng Fan, Chiu C Tan, and Haibin Ling. Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 161–170. IEEE, 2019. 3

[8] Peng Chu and Haibin Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6172–6181, 2019. 3, 8

[9] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4836–4845, 2017. 1, 3

[10] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773, 2017. 3

[11] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017. 2, 3

[12] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020. 1, 2, 4, 7

[13] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475. IEEE, 2018. 2, 8

[14] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010. 2

[15] Weitao Feng, Zhihao Hu, Wei Wu, Junjie Yan, and Wanli Ouyang. Multi-object tracking with multiple cues and switcher-aware classification. *arXiv preprint arXiv:1901.06129*, 2019. 8

[16] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. *arXiv preprint arXiv:2006.14550*, 2020. 8

[17] Shyamgopal Karthik, Ameya Prabhu, and Vineet Gandhi. Simple unsupervised multi-object tracking. *arXiv preprint arXiv:2006.02609*, 2020. 8

[18] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. 5

[19] Margret Keuper, Siyu Tang, Bjoern Andres, Thomas Brox, and Bernt Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE transactions on pattern analysis and machine intelligence*, 42(1):140–153, 2018. 8

[20] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[21] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750, 2018. 4

[22] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv preprint arXiv:1504.01942*, 2015. 4

[23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 7

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 7

[25] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools and Applications*, 78(6):7077–7096, 2019. 1, 2, 8

[26] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 1, 4, 7

[27] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 2, 7

[28] Bo Pang, Yizhuo Li, Yifan Zhang, Muchen Li, and Cewu Lu. Tubetk: Adopting tubes to track multi-object in a one-step training model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6308–6318, 2020. 8

[29] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. *arXiv preprint arXiv:2007.14557*, 2020. 8

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2

[31] Ricardo Sanchez-Matilla, Fabio Poiesi, and Andrea Cavallaro. Online multi-target tracking with strong and weak detections. In *European Conference on Computer Vision*, pages 84–99. Springer, 2016. 8

[32] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018. 7

[33] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3539–3548, 2017. 2, 8

[34] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5000–5008. IEEE, 2017. 2, 3

[35] Xingyu Wan, Jinjun Wang, Zhifeng Kong, Qing Zhao, and Shunming Deng. Multi-object tracking using online metric learning with long short-term memory. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 788–792. IEEE, 2018. 8

[36] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. *arXiv preprint arXiv:1909.12605*, 2019. 1, 2, 7, 8

[37] Greg Welch, Gary Bishop, et al. An introduction to the kalman filter, 1995. 5, 6

[38] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 1, 2, 5, 8

[39] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3988–3998, 2019. 2, 8

[40] Yihong Xu, Aljosa Osep, Yutong Ban, Radu Horaud, Laura Leal-Taixé, and Xavier Alameda-Pineda. How to train your deep multi-object tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6787–6796, 2020. 8

[41] Junbo Yin, Wenguan Wang, Qinghao Meng, Ruigang Yang, and Jianbing Shen. A unified object motion and affinity model for online multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6768–6777, 2020. 3, 8

[42] Fengwei Yu, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *European Conference on Computer Vision*, pages 36–42. Springer, 2016. 1, 2, 8

[43] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018. 3

[44] Yang Zhang, Hao Sheng, Yubin Wu, Shuai Wang, Weifeng Lyu, Wei Ke, and Zhang Xiong. Long-term tracking with deep tracklet association. *IEEE Transactions on Image Processing*, 29:6694–6706, 2020. 8

[45] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *arXiv preprint arXiv:2004.01888*, 2020. 1, 2, 3, 5, 7, 8

[46] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Fast-deepkcf without boundary effect. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4020–4029, 2019. 2, 3

[47] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Learning feature embeddings for discriminant model based tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, August 2020. 2, 3, 4, 7

[48] Linyu Zheng, Ming Tang, and Jinqiao Wang. Learning robust gaussian process regression for visual tracking. In *IJCAI*, pages 1219–1225, 2018. 5

[49] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020. 8

[50] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *arXiv preprint arXiv:1904.07850*, 2019. 2, 3

[51] Zongwei Zhou, Junliang Xing, Mengdan Zhang, and Weiming Hu. Online multi-target tracking with tensor-based high-order graph matching. In *2018 24th International Conference on Pattern Recognition (ICPR)*, pages 1809–1814. IEEE, 2018. 8

[52] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018. 1, 3, 8