DeepMOT: A Differentiable Framework for Training Multiple Object Trackers

Yihong Xu Yutong Ban Xavier Alameda-Pineda Radu Horaud
Inria
Grenoble Rhone-Alpes, France
{firstname.lastname}@inria.fr

Abstract

Multiple Object Tracking accuracy and precision (MOTA and MOTP) are two standard and widely-used metrics to assess the quality of multiple object trackers. They are specifically designed to encode the challenges and difficulties of tracking multiple objects. To directly optimize a tracker based on MOTA and MOTP is difficult, since both the metrics are strongly rely on the Hungarian algorithm, which are non-differentiable. We propose a differentiable proxy for the MOTA and MOTP, thus allowing to train a deep multiple-object tracker by directly optimizing (a proxy of) the standard MOT metrics. The proposed approximation is based on a bidirectional recurrent network that inputs the object-to-hypothesis distance matrix and outputs the optimal hypothesis-to-object association, thus emulating the Hungarian algorithm. Followed by a differentiable module, the estimated association is used to compute the MOTA and MOTP. The experimental study demonstrates the benefits of this differentiable framework on two recent deep trackers over the MOT17 dataset. Moreover, the code is publicly available from https://gitlab.inria.fr/yixu/deepmot.

1. Introduction

Object tracking is one of the core scientific challenges of computer vision. In the recent past, thanks to the advances of neural networks, great progress has been achieved in object tracking [42, 28, 46, 27, 11, 10]. Most of the research in visual tracking deals with *single* object tracking (SOT),[1] where the main difficulties are (i) the properly modeling of the target dynamics and (ii) the learning of a robust appearance model. In addition to the challenges of the single-object tracking, the complexity of multiple object tracking is further characterized by the *data-to-track assignment* problem [5]. Indeed, when dealing with tracking of multiple objects, one needs to associate data points (i.e. detections) to each of the tracks. Data association is essential not only at inference time, where the detections must be associated to different tracks, but also at evaluation time (training and performance evalution), where the inferred tracks have to be associated to the ground-truth. We will rather focus in the latter. Furthermore, these assignment problems are combinatorial and global. Combinatorial, because there is a number of assignment possibilities exponentially growing with the number of elements to be associated (tracks) and polynomially growing with the number of elements to be associated to (ground-truth objects). Global, because the decision has to be taken, considering the distance between all inferred tracks and all ground-truth objects. Moreover, both the number of inferred tracks and ground-truth objects vary over time, and therefore any assignment method must be able to cope with input distance matrices of variable size.

Traditionally, this generic assignment problem has been addressed with the Hungarian or Munkres algorithm [22]. In the case of MOT, a modified version of the Hungarian algorithm is used to assign tracks to ground-truth objects [6]. On one side, there can be a different number of tracks and ground-truth objects. On the other side, when the distance between a track and a ground-truth object exceeds a threshold, we should avoid this assignment (see [6] for more details). Therefore, not all tracks will be associated to a ground-truth object, and not all ground-truth objects will have an associated track. In an extreme case, there could be no associations at all at a given frame. What the standard and the modified Hungarian algorithms have in common is that the optimal assignment cannot be expressed as a differentiable function of the input distance matrix. This is problematic when one wishes to use MOTA or MOTP as training loss for a deep multiple object tracker. Current deep trackers are trained with ad-hoc differentiable losses that have little or nothing to do with MOTA or MOTP.

In this paper, we introduce a differentiable operator that approximates the Hungarian algorithm like in [6]. To

[1] In this paper SOT may refer to single object tracking or to single object tracker depending on the context. The same holds for MOT.

that aim, we use recurrent neural networks, and we call it deep Hungarian network (DHN). Once trained, the DHN can be used in two different ways. Firstly, to provide an approximation of the optimal track-to-ground-truth assignment, that is differentiable w.r.t. the distance matrix (and thus to approximate MOTA and MOTP to directly train deep MOT). Secondly, to convert any fully trainable deep single-object tracker, into a deep multiple object tracker. Indeed, the tracks can be inferred from a purely MOT method, from several SOT methods running in parallel, or from any combination of MOT and SOT methods working in parallel.

The rest of the paper is structured as follows. We first discuss works related to ours. We then present the structure of the overall pipeline, including the deep Hungarian network (DHN) and the operators to compute the MOTA and MOTP metrics. After, we evaluate the usefulness of the proposed pipeline in several ways. First, the advantages of the proposed training framework compared with standard losses are shown and discussed. Second, we show that thanks to the proposed training framework, a single-object tracker can achieve state-of-the-art multiple object online-tracking performance.

2. Related Work

2.1. Single Object Tracking

Single object tracking (SOT) methods [11, 10] aim to learn discriminative models to track one object and separate it from the background. A simple feed-forward regression network learning a generic relationship between object motion and appearance was proposed in [15]. Moreover, siamese trackers [42, 28, 46, 27] have recently achieved state-of-the-art performance. The original siamese tracker was proposed in [42], which is based on a correlation filter learner. It is able to extract deep features that are tightly coupled with the correlation filter. An extension of the siamese tracker using region proposal networks (RPN) was introduced in [28]. It employs the siamese subnetwork for feature extraction and the RPN to perform classification and regression. Most of recent SOT trackers can be trained end-to-end, but their performance significantly drops when applied directly to MOT.

2.2. Multiple Object Tracking

Multiple object tracking often follows the tracking-by-detection paradigm. Unlike single object tracking, the goal of a MOT tracker is to solve the data association problem. Standard benchmarks are proposed in [25, 31] for pedestrians tracking. Based on whether the algorithms use future information, MOT methods can be split into online and offline tracking.

Offline MOT [39] formulates the multi-person tracking by a multi-cut problem and use a pair-wise feature which is robust to occlusions. Person re-identification methods have been combined into a tracking framework in [40]. Moreover, [17] solves the problem by co-clustering the low-level feature point motions from optical flow and the high-level bounding-box trajectories. Quadruplet convolutional neural networks are used in [38]. They perform metric learning for target appearances together with the temporal adjacencies, which is then used for data association. In addition, [19] proposes a bilinear LSTM to learn the long-term appearance models, where the memory and the input have a linear relationship. An iterative multiple hypothesis tracking (MHT) is proposed in [37], which includes the prior association information from the previous frames. [16] fuses both the head and full-body detection into one tracking framework. Another approach is proposed in [24], which introduces a Siamese network. It encodes both the appearance information from the RGB image and the motion information from the optical-flow map. The obtained features are then processed by a linear programming based tracker.

Online MOT [1, 4] formulate the problem in a probabilistic framework, then use a variational expectation-maximization algorithm to find the track solution. Moreover, [9] proposes an aggregated local flow descriptor which encodes the relative motion pattern and then performs tracking. Another solution is proposed in [43], which uses Markov Decision Processes and reinforcement learning for the best data association. Alternatively, [35] presents a framework based on Recurrent Neural Networks (RNN). The dynamics of the appearance change, motion, and people interactions are modeled independently by a RNN. Then different information are fused together with a convolutional network. Besides, a model with dual matching attention networks is introduced by [45], which uses both spatial and temporal attention mechanisms. The estimation of the detection confidence is realized in [2], where the detections with different scores are then clustered into different classes and they are processed separately. [32] proposes a recurrent neural network (RNN) based method for both motion dynamics and detection-to-track data association and further explored it to NP-hard problems [33]. Finally, in [36] an optical-flow based approach is proposed.

2.3. Single vs. Multiple Object Trackers

Multiple object tracking frameworks have the clear advantage to be able to address the MOT problem, but most of the existing trackers are not trainable end-to-end. On the contrary, the literature on end-to-end trainable single object trackers is much more mature, but adapting to multiple-object tracking is still unclear and tracker-dependent. In this

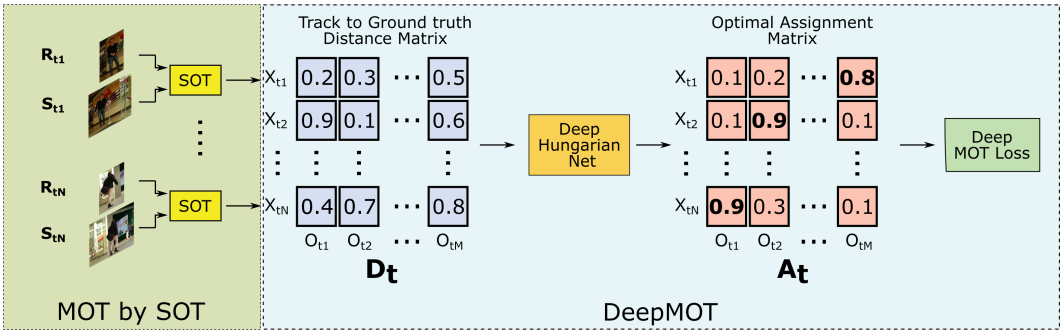

Figure 1. Overview of the DeepMOT training framework.

paper, we propose a deep proxy for the standard MOT metrics: a deep recurrent network that emulates the Hungarian algorithm. Interestingly, this allows us to directly use – and train – end-to-end trainable single object trackers, for multiple object tracking. Therefore, our formulation allows us to take full advantage of the maturity of the single object tracking literature and to train these trackers in an end-to-end fashion, to tackle the multiple object tracking problem.

3. Problem Formulation

The objective of MOT is to predict the *trajectories* of all objects at each time step, including their bounding boxes and associated identities. First, the trajectory should be precise in the sense that each bounding box should enclose its associated object well. Second, the trajectories should be accurate, meaning that only real objects and all of them should be captured (no clutter objects or missed objects) by the bounding boxes; and each trajectory should always have a unique and consistent identity through time. These properties are respectively measured by MOTP and MOTA.

With the emergence of deep learning, people address the MOT problem with (deep) neural networks, mainly by constructing robust models that capture information about motion, appearance and/or object interactions [35, 45]. However, an end-to-end training framework with a dedicated loss function for MOT remains undiscovered. As our first contribution, we propose a differentiable approximation of the two common MOT performance metrics, MOTA and MOTP, so that any trainable MOT method can be optimised to maximise these (now differentiable) metrics.

As an intermediate component for calculating the MOT metrics, the Hungarian algorithm of $O(n^3)$ time complexity give an approximated solution to the linear sum assignment problem (LSAP) by minimizing the sum distance of assigning ground-truth bounding boxes to the predicted ones. Given the algorithmic nature of this operation, no gradient of the metrics with respect to the predicted bounding boxes can be computed. And therefore the standard MOTA and MOTP cannot be used as optimisation criteria. To solve this problem, we propose a Deep Hungarian Network, or DHN, which can be seen as a differentiable function that approximates the Hungarian algorithm. In this way, the full pipeline is differentiable and the gradient of the (approximated) MOT loss can be back-propagated to any trainable MOT methods.

4. Methodology

Figure 1 shows the overview of the proposed method. Taking tracking pedestrians in a video sequence as an example, at time t, several single object trackers are used to track multiple people, providing N_t *estimated bounding boxes* (e-bb), $\boldsymbol{X}_{t1}, \ldots, \boldsymbol{X}_{tN_t}$, $\boldsymbol{X}_{tn} \in \mathbb{R}^4$, $\forall t, n$. These estimates are then compared to the M_t *ground-truth bounding boxes* (gt-bb), $\boldsymbol{O}_{t1}, \ldots, \boldsymbol{O}_{tM_t}$, $\boldsymbol{O}_{tm} \in \mathbb{R}^4$, $\forall t, m$. The pair-wise distance between e-bb and gt-bb is stored in a *distance matrix* $\mathbf{D}_t \in \mathbb{R}^{N_t \times M_t}$. The proposed deep Hungarian network is then used to estimate the optimal *soft-assignment matrix* $\mathbf{A}_t \in [0,1]^{N_t \times M_t}$. Finally, the *expected values of MOTA and MOTP*, $\overline{\text{MOTA}}$ and $\overline{\text{MOTP}}$ are computed from $\mathbf{A}_t$ and $\mathbf{D}_t$. If all septs used to compute these expected values are differentiable, then we would be able to train the tracking network(s) by directly optimising $\overline{\text{MOTA}}$ and $\overline{\text{MOTP}}$.

The remaining of the section is split in two blocks, corresponding to the green and blue boxes of Figure 1. We first describe the tracking systems, based on state-of-the-art single object trackers, together with the birth and death processes. Then, we present the structure of the deep Hungarian network and provide details on how $\overline{\text{MOTA}}$ and $\overline{\text{MOTP}}$ are computed. Importantly, the training is done in two stages (see details in the experimental sections): training, once and for all, of the DHN and training the tracking network(s) with the DHN's weights fixed.

4.1. MOT Pipeline

4.1.1 MOT by SOT

DeepMOT is able to train *any* tracking method that outputs multiple bounding box estimates. This includes running several SOT methods in parallel. In this paper, as detailed later on, we instantiate two different single object trackers [15, 29]. At time t, and for each tracked person n, both trackers expect a reference image $\mathbf{R}_{tn} \in \mathbb{R}^{H \times W \times C}$, encoding the appearance information of the target to track, and a search image $\mathbf{S}_{tn} \in \mathbb{R}^{H' \times W' \times C}$. The key idea of these trackers is to extract visual features from $\mathbf{R}_{tn}$, and use them as convolutional kernels for the visual features of $\mathbf{S}_{tn}$. In this way, these trackers are able to output an estimate of the bounding box of object n at time t, $\mathbf{X}_{tn}$. Importantly, the search region $\mathbf{S}_{tn}$ is obtained from the previous estimated object bounding box $\mathbf{X}_{(t-1)n}$, thus making the reasonable assumption that the target is not moving too fast but in the vicinity of its previous position, this assumption encodes the motion information of the target and it is commonly used for tracking pedestrians and common objects.

More formally, we represent the search of the reference image within the search image as a function, SOT of $\mathbf{R}_{tn}$, $\mathbf{S}_{tn}$ and the weights of the SOT network, W_{SOT}:

$$\mathbf{X}_{tn} = SOT(\mathbf{R}_{tn}, \mathbf{S}_{tn}, W_{\text{SOT}}), \quad \forall n. \tag{1}$$

At test time, we output all $\mathbf{X}_{tn}$ and check for new/idle tracks by means of the birth and death processes (see Section 4.1.2). At training time, we exploit the $\mathbf{X}_{tn}$'s (e-bb) together with the $\mathbf{O}_{tm}$'s (gt-bb) to compute $\overline{\text{MOTA}}$ and $\overline{\text{MOTP}}$ as a differentiable function of the $\mathbf{X}_{tn}$'s. The first step to do so, is to compute the pair-wise distance between the estimated and ground-truth bounding boxes. Generally speaking, we could use any (differentiable) distance. The commonly-used metric for measuring the similarity between two bounding boxes is intersection over union (**IoU**, also named Jaccard index). However, if two bounding boxes have no intersection, the distance $1 - \mathbf{IoU}$ will always be a constant value 1. The gradient from the loss will be 0 and W_{SOT} can not be updated through training. For this reason, we calculate D_{tnm}, the (n, m)-th element of the matrix $\mathbf{D}_t$, by using the combination of Euclidean center-point distance and $1 - \mathbf{IoU}$, formally written as:

$$D_{tnm} = \frac{L2(\boldsymbol{X}_{tn}, \boldsymbol{O}_{tm}) + (1 - \mathbf{IoU}(\boldsymbol{X}_{tn}, \boldsymbol{O}_{tm}))}{2}, \tag{2}$$

where the $L2(\cdot)$ term is the Euclidean distance normalized w.r.t. the image size and **IoU** itself is also within the range $[0, 1]$. For any distance D_{tnm}, we then have $D_{tnm} \in [0, 1]$.

The distance matrix $\mathbf{D}_t$ is the input of the DeepMOT loss module that is discussed in Section 4.2.

4.1.2 Birth and Death Processes

During test time, we use visual detections provided from MOT dataset [31] to initialize tracks and give birth to new identities. If the **IoU** of a detection $\boldsymbol{Y}_{tk}$ with any track bounding box $\mathbf{X}_{tn}$ is smaller than a threshold ξ, $\boldsymbol{Y}_{tk}$ is then considered as a birth candidate. To avoid false positive detections, if during L consecutive frames, a sequence of bounding boxes in birth candidates has an **IoU**, with each other, higher than a threshold τ, it is considered as a new track. Moreover, a track having no overlap with any detection is considered as invisible, and is removed from the tracking results. If a track is considered as invisible for Q frames, it will be removed from the tracking list. However, if the same track reappears, it will be recovered with the same identity as before. During training time, we replace the detections with ground-truth augmented with noise (see details in Section 6 Implementation details), and the same birth and death processes are applied but with different L and Q.

4.2. DeepMOT training framework

In order to train MOT methods to directly optimise MOTA ad MOTP, we first need to define two functions $\overline{\text{MOTA}}$ and $\overline{\text{MOTP}}$ that are differentiable and that approximate the original MOT metrics. As shown in Figure 1, these approximations are computed in two steps, inspired from the original MOT metrics. First, the deep Hungarian network estimates the optimal soft-assignment $\mathbf{A}_t$ from the distance matrix $\mathbf{D}_t$. Second, the MOT metrics are approximated from $\mathbf{A}_t$ and $\mathbf{D}_t$. The details of these two steps are presented in the following.

4.2.1 DHN: Deep Hungarian Network

The aim of the DHN is to compute the optimal soft-assignment matrix $\mathbf{A}_t$ from the input distance matrix $\mathbf{D}_t$. We recall that the original Hungarian algorithm takes $\mathbf{D}_t$ as the input and provides the optimal binary assignment matrix $\mathbf{A}_t^*$, i.e. the one that minimizes the sum distance cost. The element A_{tnm}^* represents the assignment decision made by the Hungarian algorithm for the n-th track and the m-th ground truth. If track n is assigned to ground truth m at time t, then $A_{tnm}^* = 1$ and 0 otherwise. Importantly, the assignment algorithm used for MOT [6] is inspired by, but not exactly, the original Hungarian algorithm. Indeed, two main differences hold: (i) the number of estimated and ground truth bounding boxes may not be the same and (ii) beyond a certain distance threshold τ_d, assignments should not be performed. This means that, in an extreme case, the optimal assignment for MOT could be empty (i.e. no assignments

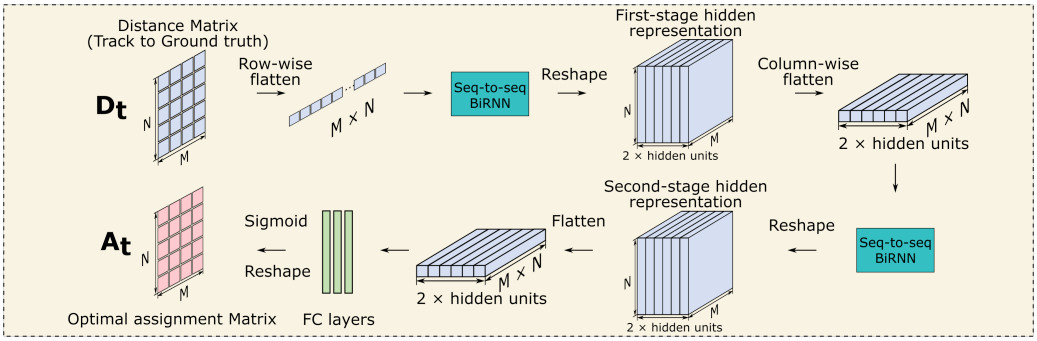

Figure 2. Overview of Deep Hungarian Net structure.

made), while for the original Hungarian algorithm, this solution is not possible. We will train the proposed DHN according to the MOT optimal assignment, rather than according to the one from the original Hungarian algorithm. However, the methodology presented remains generic and could be used to learn the behavior of the Hungarian algorithm.

The design of the proposed DHN is based on two ideas. First, the network should deal with distance matrices whose size varies over time. Second, the receptive field of all elements of $\mathbf{A}_t$ should be the whole $\mathbf{D}_t$, since the decision of the optimal assignment is global. While a fully convolutional approach could account for the first issue, large input matrices would imply partial receptive fields: the decision would be local and not global. The alternative is to use bi-directional recurrent neural networks (BiRNN).

More precisely, the structure of DHN is illustrated in Fig. 2. Since $\mathbf{D}_t$ is a two-dimensional matrix, two BiRNNs, with different weights and inputs, are sequentially applied in order to receive information from elements in $\mathbf{D}_t$ in a row-wise and in a column-wise direction. First, $\mathbf{D}_t$, flattened in the row-wise order, is input into the first BiRNN having hidden units of size h. and it outputs a sequence containing $N \times M$ elements. Each element is a vector of size $2 \times h$. Second, we reshape the output sequence into a tensor, named first-stage hidden representation of size $N \times M$, with depth of $2 \times h$. After, we perform a column-wise flatten operation before inputting to the second BiRNN, which produces the second-stage hidden representation, having same size as the first-stage representation. At that point, the second-stage hidden representation is flattened and given to three fully-connected (FC) layers. They operate independently for each of the $N \times M$ vectors of length $2 \times h$ (therefore they are independent to the size of $\mathbf{D}_t$). We apply the sigmoid function to the output after FC layers. Finally, after reshaping, we obtain $\mathbf{A}_t$.

This (now differentiable) operation is formalised as a function DHN:

$$\mathbf{A}_t = DHN(\mathbf{D}_t, W_{\text{DHN}}), \tag{3}$$

where W_{DHN} are the weights of the BiRNN and the channel-wise fully connected layers. Importantly, this formalism allows us to write the assignment matrix $\mathbf{A}_t$ as a differentiable function of the distance matrix $\mathbf{D}_t$. As described in [6], the input distance values larger than a threshold τ_d should not lead to an assignment, and are therefore multiplied by a large scaling factor inf before input to the DHN. The training procedure of the DHN is discussed later on (see Section 5). Once trained, the DHN is used for training MOT methods, but not updated anymore.

4.2.2 $\overline{\text{MOTA}}_t$ and $\overline{\text{MOTP}}_t$

Once the optimal assignment $\mathbf{A}_t$ is estimated, we need to compute (a differentiable approximation of) the MOT metrics, MOTA and MOTP. First of all, we quickly recall the definitions of MOTA and MOTP, see also [6]. Staying in the previous example of tracking pedestrians, if a track is matched to a ground truth (a match means that at time t, a ground-truth identity is assigned to a track. The matching criterion can vary according to applications, commonly, an **IoU** of two bounding boxes larger than 0.5 is considered as a match), the track is considered as a true positive (TP_t). Otherwise, it is a false positive (FP_t) and a missed ground-truth is considered as a false negative (FN_t). For a track marked as TP at time t and at the most recent previous time step, if it is assigned to different ground truth identifies, then it counts as identity switch (IDSw_t). Now MOTA is formally defined as:

$$\text{MOTA} = 1 - \frac{\sum_t (\text{FP}_t + \text{FN}_t + \text{IDSw}_t)}{\sum_t M_t}. \tag{4}$$

MOTP calculates the average of distances of bounding boxes among all the TP_t tracks and their associated ground truth bounding boxes, formally defined as:

$$\text{MOTP} = \frac{\sum_t \sum_{A^*_{tnm}=1} D_{tnm}}{\sum_t \text{TP}_t}. \tag{5}$$

We are now left with the task of computing FN_t, FP_t and IDSw_t as a differentiable function of $\mathbf{A}_t$ and $\mathbf{D}_t$. These operations are illustrated in Fig 3 ($N = M = 3$, as a simple example). FN_t and FP_t are computed with similar operations. To estimate FN_t, we append a row filled with a basis value δ to $\mathbf{A}_t$, and apply column-wise softmax, thus obtaining $\mathbf{A}_t^r$. Intuitively, if all elements of column m of $\mathbf{A}_t$ are smaller than δ, then $A^r_{t(N_t+1)m}$ will be close to one. Analogously for FP_t: a column filled with δ is appended and row-wise softmax is applied, obtaining $\mathbf{A}_t^c$. Then, the sum of the $N_t + 1$-th row of $\mathbf{A}_t^r$ and of the $M_t + 1$-th column of $\mathbf{A}_t^c$ provide an estimate of FN_t and FP_t respectively. We interpret this as expected values and write:

$$\overline{\text{FN}}_t = \sum_m A^r_{t(N_t+1)m}, \qquad \overline{\text{FP}}_t = \sum_n A^c_{tn(M_t+1)}. \quad (6)$$

To compute $\overline{\text{IDSw}}_t$ as well as $\overline{\text{MOTP}}_t$, we need a hard assignment to build binary assignment mask $\mathbf{A}_t^b$ and this can be done by positioning TPs knowing FNs and FPs. $\mathbf{A}_t^b$ is just served as a selective mask and has no gradient to be back-propagated but $\overline{\text{IDSw}}_t$ and $\overline{\text{MOTP}}_t$ do as shown below.

In order to compute the identity switches, we need to record and update historical track-ground truth identities associations knowing TP_t, named H_{IDt}. We first construct a binary hard assignment mask $\mathbf{A}^b_{t-1}$ with $\text{H}_{ID(t-1)}$ for all ground-truth objects at time t (new objects at time t can be filled the way as in $\mathbf{A}_t^b$). We then compute the element-wise product[2] of its complementary $\mathbf{1} - \mathbf{A}^b_{t-1}$ with $\mathbf{A}_t^r$, and sum up (except the last $N_t + 1$-th row), leading to:

$$\overline{\text{IDSw}}_t = \sum_{n,m}^{N_t,M_t} (1 - A^b_{(t-1)nm}) A^r_{tnm}. \quad (7)$$

We can now define the expected value of MOTA as:

$$\overline{\text{MOTA}}_t = 1 - \frac{\overline{\text{FP}}_t + \overline{\text{FN}}_t + \overline{\text{IDSw}}_t}{M_t}. \quad (8)$$

From $\mathbf{A}_t^b$, we compute the expected value of MOTP:

$$\overline{\text{MOTP}}_t = \frac{\sum_{n,m} A^b_{tnm} D_{tnm}}{|\text{TP}_t|}. \quad (9)$$

Since MOTA is to be maximized and MOTP to be minimized, the proposed DeepMOT loss at each time step t writes:

$$\mathcal{L}_{\text{DeepMOT}} = 1 - \overline{\text{MOTA}}_t + \lambda \overline{\text{MOTP}}_t, \quad (10)$$

[2]The necessary column and row additions and deletions must be performed accordingly to the birth and death processes. Importantly, the deletions and additions will never have an effect in the identity switches and the ordering of tracks and ground-truth objects in $\mathbf{A}^b_{t-1}$ should be consistent to $\mathbf{A}_t^r$.

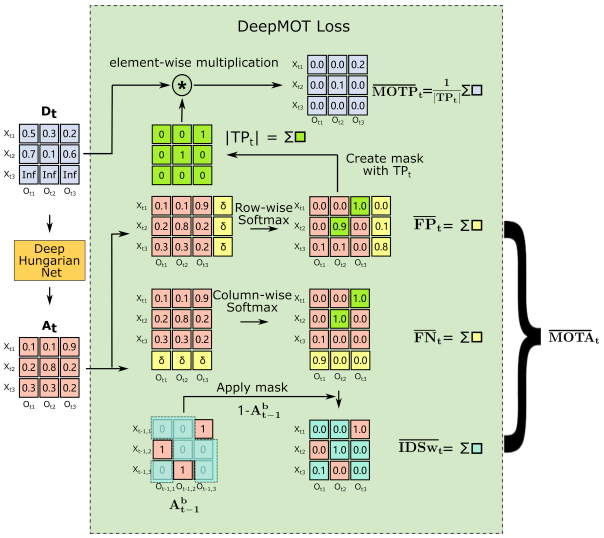

Figure 3. DeepMOT Loss.

where λ is a calibration factor.

By minimizing $\mathcal{L}_{\text{DeepMOT}}$, we are minimizing all artifacts associated to the MOT problem: false positives, missed objects, identity switches and inprecise predictions. In practice, we are explicitly telling to the MOT method in which respect the output bounding boxes should be modified to reduce all these artifacts.

5. Experiments on DHN

In this section, we first present the training details of DHN. Secondly, we compare different settings and structures of DHN and evaluate their performance in terms of the assignment accuracy. Finally, we further study the influence of different structures on MOT performance. Extensive tracking results are reported in Section 6.

Data simulation: To train the DHN, we generate distance matrices $\mathbf{D}$ of gts and detections bounding boxes provided by the MOT challenge datasets (MOT 15-17) [25, 31]. A threshold uniformly distributed between 0 and 1 is applied to $\mathbf{D}$ to augment data. $\mathbf{D}$ are then input to the Hungarian algorithm from [6] to get the corresponding $\mathbf{A}^*$. The latter is a binary matrix, where 1 represents a match, else 0. In other words, we create a data set with simulated pairs of matrices ($\mathbf{D}$ and $\mathbf{A}^*$), separated into 114,483 matrices for training and 17,880 for testing.

Network training: DHN is trained as a two-dimensional binary classification task. Focal loss [30] using a modulating factor $\gamma = 2$ is applied. Since in all $\mathbf{A}^*$, the number of zeros n_0 is much larger than that of ones, denoted as n_1, we balance the data by weighting the loss, w_1 for the positive

Structure	Recurrent unit	WA (%,↑)
RNN_A	GRU	**92.88**
RNN_B	GRU	89.56
RNN_A	LSTM	88.93
RNN_B	LSTM	83.65

Table 1. evaluation of weighted accuracy for different network configurations.

class (corresponding to ones in $\mathbf{A}^*$) and w_0 for the negative class (corresponding to zeros): $w_0 = n_1/(n_0 + n_1)$, and $w_1 = 1 - w_0$. The RMSprop optimizer [41] is used, with a learning rate of 0.0003, gradually decreased by 5% every 20,000 iterations. The training process lasts around 6 hours for 20 epochs on a GPU Titan XP.

Model comparison and selection: We study the impact of different network configurations in terms of prediction accuracy. Therefore, the weights w_1 and w_0 are also used when computing the accuracy, leading to the weighted accuracy (WA):

$$\text{WA} = \frac{w_1 n_1^* + w_0 n_0^*}{w_1 n_1 + w_0 n_0}, \tag{11}$$

where n_1^* and n_0^* are the number of true positives and true negatives respectively. Since the output of the DHN are between 0 and 1, only output values above 0.5 are considered as assignments.

Two candidate network structures are compared, namely RNN_A and RNN_B, where the RNN_A is illustrated in Fig. 2 and RNN_B is shown in Fig. 4. RNN_A is a sequential network structure while RNN_B processes $\mathbf{D}$ in a parallel way. Moreover, we try two different recurrent units (LSTM and GRU) on both network structures. The hidden size h is however fixed to 256. The result is shown in Table. 1. We observe that when using the structure RNN_B and LSTM, we achieved WA $= 83.65\%$. When switching to RNN_A, WA increases to 88.93%. The best performance is obtained by RNN_A using GRU units, with WA $= 92.88\%$.

We further study the impact of different network structures on MOT performance, plugged into DeepMOT. [29] is used as SOT. Three sequences in the training corpus of MOT17: MOT17-04, MOT17-05 and MOT17-09 are chosen for the experiment. Sequence MOT17-04 is used for training, while the rest is used for validation. During the validation, the provided SDP detections are used. All the three sequences are under the scenario with the crowded street view, where MOT17-04 and MOT17-09 are recorded with static cameras and MOT17-05 with a moving camera. For a fair comparison, all other parameters in the model are

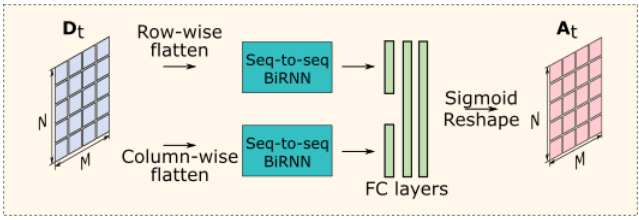

Figure 4. Alternative DHN structure (RNN_B). (1) $\mathbf{D}_t$ in the row-wise and the column-wise order. (2) The flattened vectors are then inputted to two different BiRNNs networks.The outputs are of shape $[M \times N, 2 \times h]$. (3) They then are respectively passed to a FC layer for reducing the number of channels by half. The outputs are then concatenated in the channel dimension, which provides an output of shape $[M \times N, 2 \times h]$. (4) After two FC layers followed by reshaping and sigmoid activation, we obtain $\mathbf{A}_t$, having the same size as the input.

fixed. The obtained quantitative results are reported in Table. 2.

Structure	Recurrent unit	**MOTA**(↑)	**MOTP**(↑)
RNN_A	GRU	**58.20**	**85.01**
RNN_B	GRU	53.76	84.49
RNN_A	LSTM	52.72	83.49
RNN_B	LSTM	52.42	84.82

Table 2. Comparison of MOTP and MOTA on the MOT17-05-SDP and MOT17-09-SDP sequences for models with different settings.

In this experiment, better performance of DHN will directly lead to a higher MOTA score. Since a good DHN structure will provide more accurate predictions and DeepMOT will produce more meaningful training gradient. The results in Table 2 are consistent with the the results in Table 1. The structure RNN_A with GRU outperforms the others in terms of MOTA score while keeping a similar MOTP [3].

6. Experiments on MOTs

In this section, we demonstrate the interest of our DeepMOT in three different ways. First, by comparing the performance of two different SOTs trained with the proposed DeepMOT or fine-tuned with bounding regression loss on the same data set. Second, by comparing to state-of-the-art MOT methods. Finally, by providing a graphical interpretation of the negative gradient produced by DeepMOT.

Datasets The MOT17 data set provides crowed pedestrian video sequences in real-world outdoor and indoor sce-

[3]MOTP here is calculated with **IoU**, it is thus the higher the better, some for the following evaluations.

Mode	Method	**MOTA** (↑)	**MOTP** (↑)	**IDF1** (↑)	**MT** (↑)	**ML** (↓)	**FP** (↓)	**FN** (↓)	**IDSw** (↓)
Offline	IOU [7]	45.5	76.9	39.4	15.7%	40.5%	19,993	281,643	5,988
	MHT_bLSTM [19]	47.5	77.5	51.9	18.2%	41.7%	25,981	268,042	2,069
	EDMT [8]	50.0	77.3	51.3	21.6%	36.3%	32,279	247,297	2,264
	MHT_DAM [18]	50.7	77.5	47.2	20.8%	36.9%	22,875	252,889	2,314
Online	GM_PHD[12]	36.4	76.2	33.9	4.1%	57.3%	23,723	330,767	4,607
	GMPHD_KCF[23]	39.6	74.5	36.6	8.8%	43.3%	50,903	284,228	5,811
	GMPHD_N1Tr [3]	42.1	77.7	33.9	11.9%	42.7%	18,214	297,646	10,698
	FPSN [26]	44.9	76.6	48.4	16.5%	35.8%	33,757	269,952	7,136
	DMAN [45]	48.2	75.7	55.7	19.3%	38.3%	26,218	263,608	2,194
	[15]-Pre	31.8	72.7	21.5	8.8%	46.1%	66,012	307,067	11,678
	[15]-Reg	38.6	75.0	34.5	9.5%	45.0%	41,985	297,338	6,989
	[15]-DeepMOT	44.3	76.0	38.5	13.0%	43.2%	30,302	279,144	4,861
	[29]-Pre	41.5	74.2	35.3	17.2%	38.6%	63,592	260,109	6,389
	[29]-Reg	47.2	75.3	41.5	17.3%	39.3%	31,533	262,945	3,269
	[29]-DeepMOT	48.1	76.5	43.0	17.6%	38.6%	26,490	262,578	3,696

Table 3. Tracking performance on the MOT17 dataset.

narios. It contains 14 sequences, 7 sequences for training and 7 for testing, including in total 17,757 frames and 564,228 manually annotated ground-truth bounding boxes. MOT17 provides pedestrian detections using different detectors: DPM [13], FRCNN [14], and SDP [44].

Evaluation metrics Different evaluation metrics are used. In addition to the standard MOTP and MOTA, IDF1[34] is the ratio of correctly identified detections over the average number of ground-truth and computed detections. Moreover, mostly tracked targets (MT), is the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span, while the mostly lost targets (ML) represents the ratio of the ground-truth trajectories are covered by less 20%. To notice that, as it is traditionally done, the bounding box used for evaluation is the **IoU** and not the distance used to compute $\mathcal{L}_{\text{DeepMOT}}$.

SOTs Two SOTs are selected: *GOTURN* [15] uses pre-trained CaffeNet to extract image features from both reference **R** and search images **S** ($H = W = H' = W' = 227$ and $C = 3$), followed by FC layers to predict the bounding box. *Siamese-RPN* [29], based on region proposal networks, also takes **R** and **S** as inputs($H = W = 127$, $H' = W' = 271$ and $C = 3$). It has two main branches: one provides classification scores, i.e. whether the proposal is an object or the background; the other outputs the distances among the proposed anchors and the ground-truth bounding box. The anchor with the highest classification score is selected to produce a predicted bounding box. A variant of AlexNet [21] is employed as the feature extraction backbone. All the pre-trained weights are loaded before training on different configurations.

Implementation details During training, values in **D** larger than $\tau_d = 0.5$ are multiplied by $inf = 1,000$; $Q = L = 1$, which means we give birth to new tracks and end old ones right away. The basis value is set to $\delta = 0.7$ and λ is set to 5. The Adam optimizer [20] with a learning rate of 0.0001 is used. The training procedure lasts around 72h for 15 epochs on a Titan XP GPU. In addition, to prevent overfitting during training, noise is added to the tracker bounding-box initialization. Bounding boxes are randomly re-scaled using the scaling factor ranging from 0.8 to 1.2. Also, they are randomly shifted both horizontally and vertically by a factor ranging from 0 to 0.25 of their width and height; With a probability of 0.2, standard Gaussian noise with is added to both **R** and **S**.

During test time, detections from three different detectors (DPM, SDP and FRCNN) are used to refine the tracking results **X** . If the **IoU** between X_{tn} and a detection is higher than $\zeta = 0.6$, we output their average. For the birth and death processes, the parameters are set as follows: $\tau = 0.3$, $\xi = 0.5$, $L = 3$ and $Q = 60$, see Section 4.1.2. All these details are common to all trackers and all configurations.

In the experiments, we compare training with the DeepMOT loss, to training with the standard loss, i.e. bounding box regression. For the latter, each track is initialized with the gt-bb. At each training step, MOT outputs the estimated bounding box for all tracks. The loss corresponds to the L1 distance between the estimated and the ground truth bounding boxes. Following the training settings in [15], SGD optimizer with a learning rate of 1e-5 is used. The training loss stabilises at around 20 epochs. An additional binary classification loss for the classification branch of Siamese-RPN is added to all training configurations.

Results and Comparisons We compare the results using different training configurations on MOT17 and they are reported in Table 3. We first recall the different configurations: (Pre) pre-trained trackers without training on

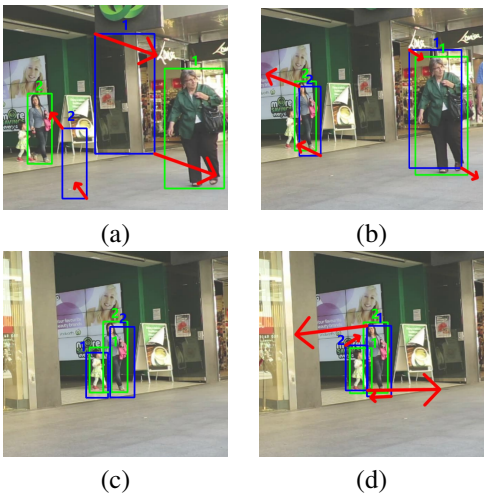

Figure 5. Visualization of negative gradient from different terms in the proposed loss (a) FP and FN (b) MOTP (c) the tracking status at $t-1$ (d) IDSw (compared with $t-1$); The tracking bounding-boxes are shown in blue; the ground-truth ones are shown in green; the gradient is in red arrow.

MOT17, (Reg) pre-trained trackers directly fine-tuned with the bounding-box regression loss on MOT17, and (DeepMOT) trackers trained with the proposed DeepMOT on MOT17.

From Table 3, we observe that compared the pre-trained configurations, the performance of both trackers is greatly improved by our proposed DeepMOT. For GOTURN, DeepMOT increases MOTA by 12.5% and MOTP by 3.3%. Compared with bounding box regression (Reg), our DeepMOT increases MOTA by 5.7% and MOTP by 1%, which means a decrease of 11,683 FPs, 18,194 FNs, and 2,128 IDs. For Siamese-RPN, DeepMOT also demonstrates largely improved performance. It outperforms bounding box regression (Reg) by improving MOTA to 48.1% from 47.2% and MOTP to 76.5% from 75.3%. Finally, our proposed [29]+DeepMOT is among the state-of-the-art methods. Importantly, [29]+DeepMOT is competitive with offline tracking methods, but without using future information.

Training Gradient The negative gradient should reflect the direction that minimizes the loss. We plot the negative gradient of different terms in DeepMOT loss w.r.t the position of each $\mathbf{X}_{tn}$ to demonstrate visually the effectiveness of our DeepMOT, illustrated in Fig. 5. In the example, we manually generated the cases which contain the FP, FN or identity switching scenarios. We see that the negative gradient does force the tracks to be close to their associated gts during training.

7. Conclusion

In this paper, we propose an end-to-end MOT training framework DeepMOT via the proxy DHN. The results reported on the MOT17 challenge dataset demonstrate that combined with the proposed framework, an SOT can achieve the nearly state-of-the-art online MOT performance.

References

[1] S. Ba, X. Alameda-Pineda, A. Xompero, and R. Horaud. An on-line variational bayesian model for multi-person tracking from cluttered scenes. *Computer Vision and Image Understanding*, 153:64–76, 2016. 2

[2] S.-H. Bae and K.-J. Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):595–610, 2018. 2

[3] N. L. Baisa and A. Wallace. Development of a n-type gm-phd filter for multiple target, multiple type visual tracking. *Journal of Visual Communication and Image Representation*, 2019. 8

[4] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud. Tracking multiple persons based on a variational bayesian model. In *European Conference on Computer Vision*, pages 52–67. Springer, 2016. 2

[5] Y. Ban, L. Girin, X. Alameda-Pineda, and R. Horaud. Exploiting the complementarity of audio and visual data in multi-speaker tracking. In *ICCV Workshop on Computer Vision for Audio-Visual Media*, 2017. 1

[6] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the CLEAR MOT metrics. *Journal on Image and Video Processing*, 2008. 1, 4, 5, 6

[7] E. Bochinski, V. Eiselein, and T. Sikora. High-speed tracking-by-detection without using image information. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2017. 8

[8] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong. Enhancing detection model for multiple hypothesis tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 18–27, 2017. 8

[9] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE international conference on computer vision*, pages 3029–3037, 2015. 2

[10] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6638–6646, 2017. 1, 2

[11] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488. Springer, 2016. 1, 2

[12] V. Eiselein, D. Arp, M. Pätzold, and T. Sikora. Real-time multi-human tracking using a probability hypothesis density filter and multiple detectors. In *2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, pages 325–330. IEEE, 2012. 8

[13] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. 2008. 8

[14] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 8

[15] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference Computer Vision (ECCV)*, 2016. 2, 4, 8

[16] R. Henschel, L. Leal-Taixé, D. Cremers, and B. Rosenhahn. Fusion of head and full-body detectors for multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1428–1437, 2018. 2

[17] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE transactions on pattern analysis and machine intelligence*, 2018. 2

[18] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *Computer Vision (ICCV), IEEE International Conference on*. IEEE, Dec. 2015. 8

[19] C. Kim, F. Li, and J. M. Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 200–215, 2018. 2, 8

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 8

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 8

[22] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 1

[23] T. Kutschbach, E. Bochinski, V. Eiselein, and T. Sikora. Sequential sensor fusion combining probability hypothesis density and kernelized correlation filters for multi-object tracking in video data. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–5. IEEE, 2017. 8

[24] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016. 2

[25] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942 [cs]*, Apr. 2015. arXiv: 1504.01942. 2, 6

[26] S. Lee and E. Kim. Multiple object tracking via feature pyramid siamese networks. *IEEE Access*, 7:8181–8194, 2019. 8

[27] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. *arXiv preprint arXiv:1812.11703*, 2018. 1, 2

[28] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018. 1, 2

[29] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 4, 7, 8, 9

[30] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 6

[31] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831. 2, 4, 6

[32] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 2

[33] A. Milan, S. H. Rezatofighi, R. Garg, A. Dick, and I. Reid. Data-driven approximations to np-hard problems. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 2

[34] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*, pages 17–35. Springer, 2016. 8

[35] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 300–311, 2017. 2, 3

[36] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker. Deep network flow for multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6951–6960, 2017. 2

[37] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, and J. Yu. Iterative multiple hypothesis tracking with tracklet-level association. *IEEE Transactions on Circuits and Systems for Video Technology*, 2018. 2

[38] J. Son, M. Baek, M. Cho, and B. Han. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5620–5629, 2017. 2

[39] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision*, pages 100–111. Springer, 2016. 2

[40] S. Tang, M. Andriluka, B. Andres, and B. 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

[41] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012. 7

[42] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2805–2813, 2017. 1, 2

[43] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE international conference on computer vision*, pages 4705–4713, 2015. 2

[44] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2129–2137, 2016. 8

[45] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382, 2018. 2, 3, 8

[46] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 101–117, 2018. 1, 2