FAMNet: Joint Learning of Feature, Affinity and Multi-dimensional Assignment for Online Multiple Object Tracking

Peng Chu and Haibin Ling
Temple University
Philadelphia, PA USA
{pchu, hbling}@temple.edu

Abstract

Data association-based multiple object tracking (MOT) involves multiple separated modules processed or optimized differently, which results in complex method design and requires non-trivial tuning of parameters. In this paper, we present an end-to-end model, named FAMNet*, where Feature extraction, Affinity estimation and Multi-dimensional assignment are refined in a single network. All layers in FAMNet are designed differentiable thus can be optimized jointly to learn the discriminative features and higher-order affinity model for robust MOT, which is supervised by the loss directly from the assignment ground truth. We also integrate single object tracking technique and a dedicated target management scheme into the FAMNet-based tracking system to further recover false negatives and inhibit noisy target candidates generated by the external detector. The proposed method is evaluated on a diverse set of benchmarks including MOT2015, MOT2017, KITTI-Car and UA-DETRAC, and achieves promising performance on all of them in comparison with state-of-the-arts.*

1. Introduction

Tracking multiple objects in video is critical for many applications, ranging from vision-based surveillance to autonomous driving. A current popular framework to solve multiple object tracking (MOT) uses the tracking-by-detection strategy where target candidates generated from an external detector are associated and connected to form the target trajectories across frames [1, 14, 21, 34, 37, 45, 49]. At the core of tracking-by-detection strategy lies the data association problem which is usually treated as three separate parts: *feature extraction* for candidate representation, *affinity metric* to evaluate the cost of each association hypothesis and *association algorithm* to find the optimal association. These parts involve multiple individual data-processing steps and are optimized differently from each other, which results in a complex method design and extensive tuning parameters to adapt different target categories and tracking scenarios.

Recently, deep neural network (DNN) has been investigated intensively to learn the association cost function in a unified architecture combining both feature extraction and affinity metric [10, 26, 42]. Through training, the task and scenario prior can be automatically adapted by the candidate representation and estimation metric without manually tuning the hyper-parameters. However, the association algorithm still stands outside the network, which requires dedicated affinity samples to be manually fabricated from ground truth association for the training process. It is not guaranteed that training and inference phases share the same data distribution; consequently it may lead to the degraded generalizability of the trained model. Moreover, crowded targets, similar appearance and fast motion impose great ambiguity for the association only considering pairs of neighboring frames. Successful association requires global optimization across multiple frames, where higher-order discriminative clues such as appearance changes over time and motion context could be included. Learning the robust representation and affinity criteria without the cooperation from the association procedure in this circumstance is even more complicated.

Our objective in this paper is to formulate an end-to-end model for MOT: the Feature representation, Affinity model and Multi-dimensional assignment (MDA) are refined in a single deep network named FAMNet, which is optimized jointly to learn the task prior. In particular, feature subnetwork is used to extract features for candidates on each frame, after which an affinity sub-network estimates the higher-order affinity for all association hypothesis. With the affinity, the MDA sub-network is to optimize globally and obtain the optimal assignments. By all layers in FAMNet designed differentiable, the feature and affinity sub-network can be trained directly referring to the assignment ground truth. To realize it, we make the following novelties to the FAMNet and its based tracking system:

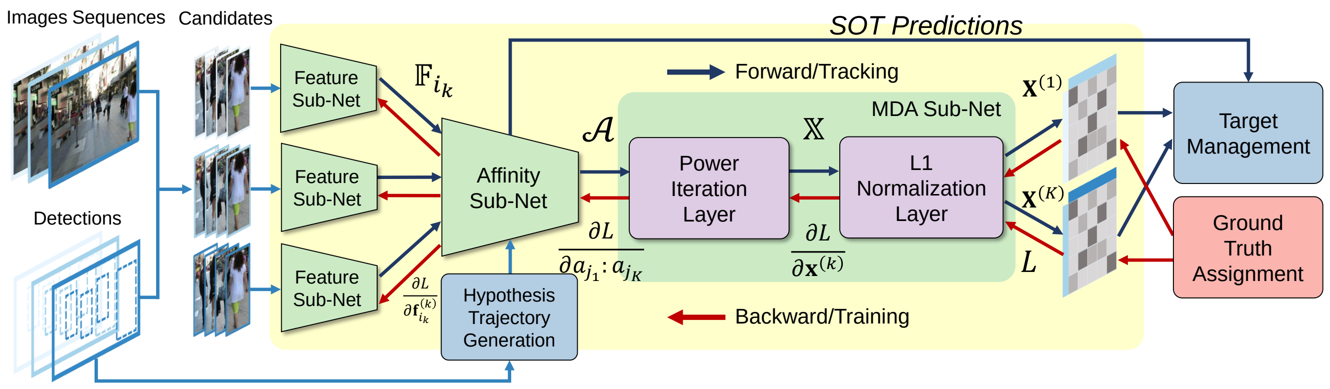

Figure 1. Overview of our FAMNet based tracking system. The sub-networks inside the yellow background consist the FAMNet. $\mathbb{F}_{i_k}$ is a set of features extracted from each frame as detailed in Sec. 4.1 and L is for the total loss.

- We design an affinity sub-network that fuses discriminative higher-order appearance and motion information into the affinity estimation.
- We propose an MDA sub-network, in which a modified rank-1 tensor approximation power iteration is designed differentiable and adapted for the deep learning architecture.
- We integrate single object tracking into the data association-based MOT. Detections and tracking predictions are merged and selected optimally through MDA to construct the target trajectories.
- We employ a target management scheme where a dedicated CNN network is used to refine the target bounding box to eliminate the noised candidates generated by external detector.

To show the effectiveness of the proposed approach, it is evaluated on the popular multiple pedestrian and vehicle tracking challenge benchmarks including MOT2015, MOT2017, KITTI-Car and UA-DETRAC. Our results show promising performance in comparison with other published works.

2. Related Work

Multiple object tracking (MOT) has been an active research area for decades, and many methods have been investigated for this topic. Recently, the most popular framework for MOT is the tracking-by-detection. Traditional methods primarily focus on solving the data association problem using such as Hungarian algorithm [3, 15, 19], network flow [12, 53, 55] and multiple hypotheses tracking [6, 23] on various of affinity estimation schemes. Higher-order affinity provides the global and discriminative information that is not available in pairwise association. In order to utilize it, MOT is usually treated as the MDA problem. Collins [11] proposes a block ICM-like method for the MDA to incorporate higher-order motion model. The method iteratively solves bipartite assignments alternatively while keeping other assignments fixed. In [41], MDA is formulated as the rank-1 tensor approximation problem where a dedicated power iteration with unite $\ell 1$ normalization is proposed to find the optimal solution. Our work is closely related to the MDA formulation, especially [41].

Recently, deep learning is explored with increasing popularity in MOT with great success. Most recent solutions rely on it as a powerful discriminative technique [1, 27, 42, 56]. Tang et al. [43] propose to use DNN based Re-ID techniques for affinity estimations. They include lift edges that connect two candidates spanning multiple frames to capture the long-term affinity. In [39], recurrent neural networks (RNN) and long short-term memory (LSTM) is adapted to model the higher-order discriminative clue. Those methods learn the networks in a separate process with the manually fabricated affinity training samples.

Some recent works have gone further to tentatively solve MOT in an entirely end-to-end fashion. Ondruska and Posner [35] introduce the RNN for the task to estimate the candidate state. Although this work is demonstrated on the synthetic sensor data and no explicit data association is applied, it firstly shows the efficacy of using RNN for an end-to-end solution. Milan et al. [31] propose an RNN-LSTM based online framework to integrate both motion affinity estimation and bipartite association into the deep learning network. They use LSTM to solve the data association target by target at each frame where the constrains in data association are not explicitly built into the network but learned from training data. For both works, only the occupancy status of targets are considered, the informative appearance clue is not utilized. Different from their methods, we propose an MDA sub-network which handles both the data association and the constrains, and our affinity fuses both the appearance and motion clue for better discriminability.

3. Overview

In this section, we first formulate the multiple object tracking (MOT) problem as a multi-dimensional assignment (MDA) form, and then provide an overview of our FAMNet-based tracking system (overview in Fig. 1).

3.1. Problem Formulation

Following the notation in [41], the input for MOT is denoted by $\mathbb{O} = \{\mathbb{O}^{(k)}\}_{k=0}^{K}$, which contains $K+1$ target candidate sets from $K+1$ frames. For frame k, $\mathbb{O}^{(k)} = \{\mathbf{o}_{i_k}^{(k)}\}_{i_k=1}^{I_k}$ is the set of I_k candidates to be matched or associated, where $\mathbf{o}_{i_k}^{(k)}$ represents the status of the candidate such as its center coordinate on the image frame.

With the input candidate set $\mathbb{O}$, MOT is to find a multi-dimensional association that maximizes the overall affinity subject to the association constrains. In detail, $c_{i_0:i_K} \doteq c_{i_0 i_1 \ldots i_K} \geq 0$ denotes affinity for one possible association, or in term of MOT, one hypothesis trajectory $\mathbf{t}_{i_0:i_K}$ composed by candidates $\{\mathbf{o}_{i_0}^{(0)}, \mathbf{o}_{i_1}^{(1)}, ..., \mathbf{o}_{i_K}^{(K)}\}$. We use $z_{i_0:i_K} \doteq z_{i_0 i_1 \ldots i_K}$ to indicate whether a hypothesis trajectory is true ($z_{i_0:i_K} = 1$) or not ($z_{i_0:i_K} = 0$). If we further denote tensor $\mathcal{C} = (c_{i_0:i_K})$ and $\mathcal{Z} = (z_{i_0:i_K})$, the MOT can be formulated as following MDA problem to solve $\mathcal{Z}$ based on $\mathcal{C}$:

$$\arg\max_{\{z_{i_0:i_K}\}} \sum_{i_0:i_K} c_{i_0:i_K} z_{i_0:i_K} = \arg\max_{\mathcal{Z}} \|\mathcal{C} \circ \mathcal{Z}\|_1, \qquad (1)$$

$$\text{s.t.} \begin{cases} \sum\limits_{i_0:i_K/\{i_k\}} z_{i_0:i_K} = 1, & \forall k = 0, 1, ..., K \\ z_{i_0:i_K} \in \{0, 1\}, & \forall i_k = 1, 2, ..., I_k \end{cases} \qquad (2)$$

where $\circ$ denotes the element-wise product, $\|\cdot\|_1$ is the matrix 1-norm, and $\sum_{i_0:i_K/\{i_k\}}$ stands for summation over all subscripts $i_0 : i_K$ except for i_k.

To solve Eq. 1, we follow the Rank-1 Tensor Approximation (R1TA) framework [41]. The multi-dimensional assignments $\mathcal{Z}$ are first decomposed as the product of a serials of local assignments $\mathbf{X}^{(k)} = (x_{i_{k-1}i_k}^{(k)})$ which represent the assignments between candidates in adjacent frames, *i.e.* $\mathbb{O}^{(k-1)}$ and $\mathbb{O}^{(k)}$. If we further rewrite the local assignment matrix into vector form $\mathbf{x}^{(k)} = (x_{j_k}^{(k)})$ where $j_k = (i_{k-1} - 1) \times I_k + i_k$,[1] optimization problem defined in Eq. 1 can be rewritten as follow:

$$\arg\max_{\mathbb{X}} \mathcal{A} \times_1 \mathbf{x}^{(1)} \times_2 \mathbf{x}^{(2)} \cdots \times_K \mathbf{x}^{(K)}, \qquad (3)$$

where $\mathcal{A} = (a_{j_1:j_K})$ is the reshaped affinity tensor from the $(K+1)$-th order $\mathcal{C}$ tensor to a K-th order tensor following the rules defined in [41] and $\times_k$ is the k-*mode* tensor product, $\mathbb{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(K)}\}$ is the set of local assignment vectors which we are optimizing for.

[1]For notational convenience, the same symbol is used for elements in $\mathbf{X}^{(k)}$ and $\mathbf{x}^{(k)}$ with double subscripts and a single subscript respectively.

3.2. Architecture Overview and Tracking Pipeline

For each association batch, the FAMNet based tracking system takes the $K+1$ image frames and corresponding detections provided by an external detector as input. Detection candidates are first used to generate the hypothesis trajectories. Image patches of target candidates together with the trajectory hypothesis are passed into FAMNet to compute the set of local assignments as shown in Fig. 1. Inside FAMNet, features of candidate patches are extracted through a feature sub-network. The affinity sub-network then calculates the affinity for all hypothesis trajectories on those features to form the affinity tensor as described in Sec. 4.1. With the affinity tensor, the optimal multi-dimension assignments are estimated by the MDA sub-network as explained in Sec. 4.2 and 4.3.

During training, the assignment ground truth is directly compared with the network output to compute the loss. The loss signal then back-propagates throughout the network to the feature and affinity sub-networks for learning, which is illustrated as the red paths in Fig. 1 and detailed in Sec. 4.4. In tracking phase, the output assignments together with single object tracking (SOT) predictions are used to update the trajectories of tracked targets through the target management scheme as described in Sec. 4.5 and 4.6.

We design our method in the online tracking framework intended for more casual applications. Under the constant velocity assumption, three frames are the minimum temporal span to calculate the motion affinity. Therefore, in the rest of paper, $K = 2$ with two frames overlapping between association batches is used as to balance the computation cost and the sufficient depth of association to include the higher-order discriminative clues.

4. FAMNet

4.1. Affinity Sub-Network

The affinity sub-network takes the features of candidates and hypothesis trajectories as input, and generates the affinity tensor as output.

For each association batch, the feature sub-network, which is a Siamese-style network, is first used to extract spatially aligned feature for the candidates from all frames in the batch. The candidates in the middle frame of the batch are treated as *anchor candidates*, and the middle frame is referred as *anchor frame*. E.g., for $K = 2$, the anchor frame refers to the frame $k = 1$. For each anchor candidate $\mathbf{o}_{i_1}^{(1)}$, $K+1$ features are extracted respectively from the $K+1$ frames, denoted as $\mathbb{F}_{i_1} = \{\mathbf{f}_{i_1}^{(0)}, \mathbf{f}_{i_1}^{(1)}, \mathbf{f}_{i_1}^{(2)}\}$. These features are all centered at the same location of $\mathbf{o}_{i_1}^{(1)}$ on frame $k = 1$, as illustrated on the left of Fig. 2. This way, features in $\mathbb{F}_{i_1}$ share the same coordinate origin thus can encode motion clue when concatenated along a channel. Spatial dimension

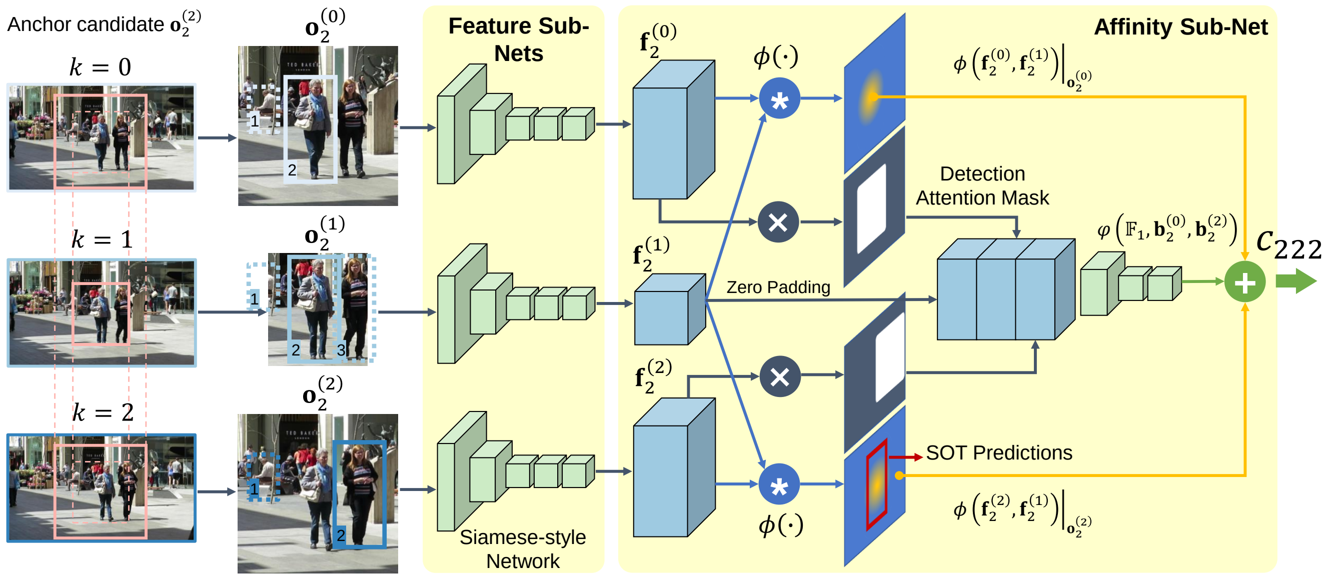
Anchor candidate $\mathbf{o}_2^{(2)}$
$k=0$
$k=1$
$k=2$
$\mathbf{o}_2^{(0)}$
$\mathbf{o}_2^{(1)}$
$\mathbf{o}_2^{(2)}$
Feature Sub-Nets
Siamese-style Network
Affinity Sub-Net
$\mathbf{f}_2^{(0)}$
$\mathbf{f}_2^{(1)}$
$\mathbf{f}_2^{(2)}$
$\phi(\cdot)$
$\phi\left(\mathbf{f}_2^{(0)},\mathbf{f}_2^{(1)}\right)\big|_{\mathbf{o}_2^{(0)}}$
Detection Attention Mask
Zero Padding
$\varphi\left(\mathbb{F}_1,\mathbf{b}_2^{(0)},\mathbf{b}_2^{(2)}\right)$
c_{222}
SOT Predictions
$\phi\left(\mathbf{f}_2^{(2)},\mathbf{f}_2^{(1)}\right)\big|_{\mathbf{o}_2^{(2)}}$

Figure 2. Illustration of the feature and affinity sub-networks for $K = 2$. It shows an example to calculate the affinity c_{222} for the hypothesis trajectory $\mathbf{t}_{222}$. The red bounding boxes in the first column of images indicate the locations on the image frame for the patch in the second column. The blue bounding boxes in the second column illustrate the detection candidates. Best viewed in color.

of $\mathbf{f}_{i_1}^{(1)}$ is determined by the bounding box of $\mathbf{o}_{i_1}^{(1)}$, while others are the multiples of $\mathbf{f}_{i_1}^{(1)}$ in order to include enough candidates on adjacent frames into the same view. Note that, hypothesis trajectories sharing the same anchor candidate have the same set of $\mathbb{F}_{i_k}$. Therefore, $(K+1)I_k$ features are extracted for each association batch.

Two levels of affinities are calculated for each hypothesis trajectory using the extracted feature set, as shown in Fig. 2. In detail, the affinity tensor is calculated as following:

$$c_{i_0i_1i_2} = \begin{cases} \quad\varphi\left(\mathbb{F}_{i_1}, \mathbf{b}_{i_0}^{(0)}, \mathbf{b}_{i_2}^{(2)}\right) & \\ +\phi\left(\mathbf{f}_{i_1}^{(0)}\big|_{\mathbf{o}_{i_0}^{(0)}}, \mathbf{f}_{i_1}^{(1)}\right) & \mathbf{t}_{i_0i_1i_2} \in \mathbb{T} \\ +\phi\left(\mathbf{f}_{i_1}^{(2)}\big|_{\mathbf{o}_{i_2}^{(2)}}, \mathbf{f}_{i_1}^{(1)}\right), & \\ 0, & \mathbf{t}_{i_0i_1i_2} \notin \mathbb{T}, \end{cases} \tag{4}$$

where $\mathbb{T}$ is the set of valid hypothesis trajectories, $\mathbf{b}_{i_k}^{(k)}$ is the bounding box associated with $\mathbf{o}_{i_k}^{(k)}$, $\phi(\cdot)$ calculates the pair-wise affinity and $\varphi(\cdot)$ evaluates the long-term affinity of a hypothesis trajectory, $\mathbf{f}_{i_1}^{(0)}\big|_{\mathbf{o}_{i_0}^{(0)}}$ is a spatial subset of $\mathbf{f}_{i_1}^{(0)}$ centered at $\mathbf{o}_{i_0}^{(0)}$ with the same spatial dimension as $\mathbf{f}_{i_1}^{(1)}$. The actual center coordinates of $\mathbf{o}_{i_0}^{(0)}$ in $\mathbf{f}_{i_1}^{(0)}$ need to be converted accordingly. We use $\mathbf{o}_{i_0}^{(0)}$ here for convenience.

For pair-wise affinity, the cross-correlation operation is used, such as

$$\phi\left(\mathbf{f}_{i_1}^{(0)}\big|_{\mathbf{o}_{i_0}^{(0)}}, \mathbf{f}_{i_1}^{(1)}\right) = \mathbf{f}_{i_1}^{(1)} * \mathbf{f}_{i_1}^{(0)}\big|_{\mathbf{o}_{i_0}^{(0)}} = \phi\left(\mathbf{f}_{i_1}^{(0)}, \mathbf{f}_{i_1}^{(1)}\right)\big|_{\mathbf{o}_{i_0}^{(0)}}, \tag{5}$$

where $*$ is the convolution operation. Due to the fact that hypothesis trajectories sharing the same anchor candidates have the same set of $\mathbb{F}_{i_1}$, we can calculate $\mathbf{f}_{i_1}^{(1)} * \mathbf{f}_{i_1}^{(0)}$ first then take the value at $\mathbf{o}_{i_0}^{(0)}$ from the cross-correlation result, which is referred as $\phi(\cdot)\big|_{\mathbf{o}_{i_0}^{(0)}}$.

We use convolutional neural network (CNN) with spatial attention to evaluate the higher-order affinity of hypothesis trajectory. For this purpose, features $\mathbf{f}_{i_1}^{(0)}$ and $\mathbf{f}_{i_1}^{(2)}$ are multiplied with spatial masks generated from $\mathbf{b}_{i_0}^{(0)}$ and $\mathbf{b}_{i_2}^{(2)}$ as shown in Fig. 2. In particular, we create a binary mask of the same spatial size with $\mathbf{f}_{i_1}^{(0)}$ or $\mathbf{f}_{i_1}^{(2)}$ for each candidate. Inside each mask, the region within $\mathbf{b}_{i_k}^{(k)}$ is set to 1 otherwise is 0. Each time, the actual position of $\mathbf{b}_{i_k}^{(k)}$ in the mask is converted from the image frame to the coordinates centered at anchor candidates. After encoding the spatial-temporal information, features in $\mathbb{F}_{i_1}$ are concatenated along channel to form the input of a CNN to estimate the long-term affinity. The final affinity for a hypothesis trajectory is the summation of two levels of affinity according to Eq. 4.

4.2. R1TA Power Iteration Layer

With the affinity tensor, we use R1TA power iteration to estimate the set of optimal assignments in Eq. 3. Solving the global optimum for MDA usually requires NP-hard probing. A sub-optimal approximation is usually guaranteed by a power iteration algorithm which can be expressed in the pure mathematical format.

In order to fit this process into the deep network framework, we adapt a different iteration scheme than the one in [41] where the row/column $\ell 1$ normalization is applied to $\mathbb{X}$ after each iteration to enforce the constrain defined in Eq. 2. The tensor power iteration and row/column normalization are separated into two independent layers in our design. It avoids cumulating too deep operations in a single layer and alleviate the potential gradient vanishing. Downside of this scheme is that we could not expect the same convergence property as in [41]. However, benefited from the end-to-end training, it can be compensated by the learned more discriminative feature and affinity metric.

In detail, the optimal solution to Eq. 3 subject to Eq. 2 is approximated iteratively by:

Forward pass. At the $(n+1)$-th iteration, the elements in $\mathbf{x}^{(1)(n+1)}$ is calculated from[2]

$$x_{j_1}^{(1)(n+1)} = \frac{x_{j_1}^{(1)(n)}}{C^{(n)}} \sum_{j_1:j_K/\{j_1\}} a_{j_1:j_K} x_{j_2}^{(2)(n)} \cdots x_{j_K}^{(K)(n)}, \quad (6)$$

where $C^{(n)} = \sum_{j_1:j_K} a_{j_1:j_K} x_{j_1}^{(1)(n)} x_{j_2}^{(2)(n)} \cdots x_{j_K}^{(K)(n)}$ is the $\ell 1$ normalization factor. At initialization, elements in all local assignment vectors $\mathbf{x}^{(k)(0)}$ are set to 1.

Backward pass. The R1TA power iteration layer computes the loss gradient of affinity tensor $\mathcal{A}$, denoted by $\partial L / \partial a_{j_1:j_K}$, as backward output. The input of the backward pass is the loss gradient of all local assignment vectors at the last iteration, e.g. $\partial L / \partial \mathbf{x}^{(k)(N)}$, where N is the total number of iterations performed in the forward pass. The gradient output is calculated as follow:

$$\begin{aligned} \frac{\partial L}{\partial a_{j_1:j_K}} = & \sum_n \frac{x_{j_1}^{(1)(n)} x_{j_2}^{(2)(n)} \cdots x_{j_K}^{(K)(n)}}{C^{(n)}} \\ & \times \sum_k \left(\mathbf{i}_{j_k}^{(k)} - \mathbf{x}^{(k)(n+1)}\right)^\top \frac{\partial L}{\partial \mathbf{x}^{(k)(n+1)}}, \end{aligned} \quad (7)$$

where $\mathbf{i}_{j_k}^{(k)}$ is a unit vector of the same dimension as $\mathbf{x}^{(k)}$ and has elements equal to 1 only at j_k and otherwise 0. In order to calculate Eq. 7 for all iterations, the loss gradients of assignment vectors at each iteration are also needed, which follows:

$$\begin{aligned} \frac{\partial L}{\partial x_{j_1}^{(1)(n)}} = & \frac{x_{j_1}^{(1)(n+1)}}{x_{j_1}^{(1)(n)}} \left[\left(\mathbf{i}_{j_1}^{(1)} - \mathbf{x}^{(1)(n+1)}\right)^\top \frac{\partial L}{\partial \mathbf{x}^{(1)(n+1)}} \right. \\ & \left. - \sum_{k \neq 1} \left(\mathbf{x}^{(1)(n+1)}\right)^\top \frac{\partial L}{\partial \mathbf{x}^{(1)(n+1)}} \right] + \frac{\sum_{k \neq 1} h_{j_1}^{(k)(n)}}{C^{(n)}}, \end{aligned} \quad (8)$$

where $h_{j_1}^{(k)(n)}$ can be calculated as, *e.g.* for $k=2$, $h_{j_1}^{(2)(n)} = \sum_{l_1:l_K/\{l_1\}} a_{j_1 l_2 \cdots l_k} x_{l_3}^{(3)(n)} \cdots x_{l_K}^{(K)(n)} \frac{\partial L}{\partial x_{l_2}^{(2)(n)}}$.

[2]The second superscript indicates the round of iteration. Moreover, derivation in this subsection is on $x_{j_1}^{(1)}$, but is the same for other $x_{j_k}^{(k)}$.

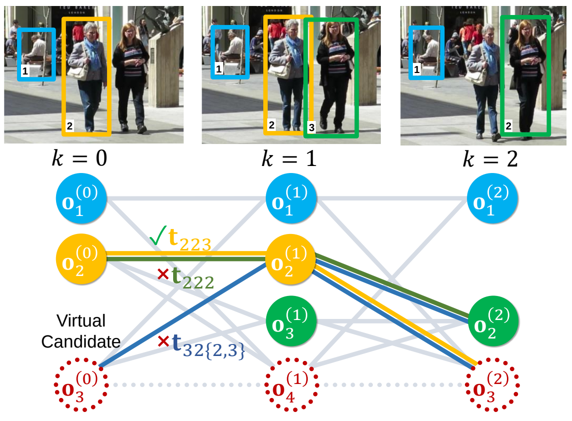

Figure 3. Hypothesis trajectory generation. It shows all hypothesis trajectories passing through candidate $\mathbf{o}_2^{(1)}$. The color of nodes indicates the ground truth associations.

4.3. $\ell 1$ Normalization Layer

To satisfy the constrains defined in Eq. 2 required by MDA, row/column $\ell 1$ normalization is applied to the result $\mathbb{X}$. The assignment vectors from the R1TA power iteration layer are reshaped back to their matrix form such that $\mathbf{X}^{(k)} = (x_{i_{k-1} i_k}^{(k)}) \in \mathbb{R}^{I_{k-1} \times I_k}$. Then, the $\ell 1$ normalization is performed row and column alternatively through multiple iterations.

Forward pass. For each pair of iterations, we start from row normalization. In the $(n+1)$ and $(n+2)$-th iteration:

$$\begin{aligned} \mathbf{X}^{(k)(n+1)} &= \left[\mathbf{X}^{(k)(n)} \mathbf{1}_{I_k}\right]^{-1} \mathbf{X}^{(k)(n)} \\ \mathbf{X}^{(k)(n+2)} &= \mathbf{X}^{(k)(n+1)} \left[\mathbf{1}_{I_{k-1}}^\top \mathbf{X}^{(k)(n+1)}\right]^{-1}, \end{aligned} \quad (9)$$

where $\mathbf{1}_{I_k}$ is a vector of $\mathbb{R}^{I_k}$ with all elements being 1, $[\mathbf{x}]$ here and below represents the diagonal matrix with $\mathbf{x}$ as diagonal elements.

Backward pass. Given a starting gradient $\partial L / \partial \mathbf{X}^{(k)(n+2)}$, we iteratively compute the gradients as:

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{X}^{(k)(n+1)}} = & \frac{\partial L}{\partial \mathbf{X}^{(k)(n+2)}} \left[\mathbf{1}_{I_{k-1}}^\top \mathbf{X}^{(k)(n+2)}\right]^{-1} - \mathbf{1}_{I_{k-1}} \\ & \cdot \operatorname{diag}\left(\left[\mathbf{1}_{I_{k-1}}^\top \mathbf{X}^{(k)(n+2)}\right]^{-2} \left(\mathbf{X}^{(k)(n+2)}\right)^\top \frac{\partial L}{\partial \mathbf{X}^{(k)(n+2)}} \right)^\top \\ \frac{\partial L}{\partial \mathbf{X}^{(k)(n)}} = & \left[\mathbf{X}^{(k)(n+1)} \mathbf{1}_{I_k}\right]^{-1} \frac{\partial L}{\partial \mathbf{X}^{(k)(n+1)}} \\ & - \operatorname{diag}\left(\left[\mathbf{X}^{(k)(n+1)} \mathbf{1}_{I_k}\right]^{-2} \frac{\partial L}{\partial \mathbf{X}^{(k)(n+1)}} \left(\mathbf{X}^{(k)(n+1)}\right)^\top \right) \mathbf{1}_{I_k}^\top. \end{aligned}$$

Our $\ell 1$ normalization layer is similar to but differs from that in [54], in that our implementation allows partial row/column normalization to handle real and virtual candidates differently as detailed in Sec. 4.5.

4.4. Training

During the training, the total loss L is measured by the binary cross entropy between all predicted assignments

$\left(x^{(k)}_{i_{k-1}i_k} \in [0,1]\right)$ and assignment ground truth $\left(\bar{x}^{(k)}_{i_{k-1}i_k} \in \{0,1\}\right)$, which is written as

$$L = \sum_k \sum_{i_{k-1}i_k} \bar{x}^{(k)}_{i_{k-1}i_k} \log x^{(k)}_{i_{k-1}i_k} + (1 - \bar{x}^{(k)}_{i_{k-1}i_k}) \log(1 - x^{(k)}_{i_{k-1}i_k}).$$

With the total loss, gradients are calculated throughout the network back to the affinity and feature sub-networks.

We use each association batch containing $K+1$ frames as one mini-batch during training and tracking. For each batch, $K+1$ consecutive frames and target candidates on each frame provided by an external detector serve as the input. The candidate set is first used to generate the hypothesis trajectories $\{\mathbf{t}_{i_0:i_K}\}$. We set a bound for the hypothesis trajectory generation where two candidates from two consecutive frames can be connected only when they are spatially close to each other and have the similar bounding box size. We set adaptive thresholds for this strategy. In detail, if a candidate cannot connect with any candidate at a threshold, it re-searches using degraded thresholds for the possible connection. This strategy allows the connections for both fast and slow movement targets, and meanwhile rejects naive false positive connections. Hypothesis trajectories are generated greedily by iterating through all valid connections to form trajectories starting with candidates in $\mathbb{O}^{(0)}$ and terminating in $\mathbb{O}^{(K)}$. Generated trajectories are sorted by their anchor candidates and together with image frames fed into our networks for training and tracking.

4.5. Tracking by Integrating Detection and SOT

In the tracking phase, predictions using SOT techniques are included to recover missing candidates from the external detector. We add a virtual candidate to each candidate set to represent missing candidates and allow it to connect with any candidate in consecutive frames as shown in Fig. 3. Both real and virtual candidates are used to generate trajectory hypothesis. When calculating affinity, we choose the location maximizing the affinity in Eq. 5 as the center of the virtual candidates for each anchor candidate such that

$$\arg\max_{\hat{\mathbf{o}}^{(2)}_{I_2+1}} \phi\left(\mathbf{f}^{(2)}_{i_1}, \mathbf{f}^{(1)}_{i_1}\right) = \left(\mathbf{f}^{(1)}_{i_1} * \mathbf{f}^{(2)}_{i_1}\right)\Big|_{\hat{\mathbf{o}}^{(2)}_{I_2+1}}. \quad (10)$$

Therefore, if an anchor candidate misses its detection in consecutive frame, it will connect with the virtual candidate which represents the location most similar to it in that consecutive frame, or, in terms of SOT, its tracking prediction. Each anchor candidate may have a different location predicted by SOT. We use $\hat{\mathbf{o}}^{(2)}_{I_2+1}$ in Eq. 10 to refer to the virtual candidate, its center coordinates may vary on different anchor candidates.

To prevent MDA from always choosing the virtual candidates since their affinity are no smaller than any real candidate, a coefficient $\alpha \in (0,1)$ is used to scale down their affinity. The virtual candidates are handled specially in the

Algorithm 1 Target Management

```
1: Input: Assignment matrix $\mathbf{X}^{(k)} = (x^{(k)}_{i_{k-1}i_k}) \in \mathbb{R}^{I_{k-1}\times I_k}$.
2: Output: Tracked target trajectories.
3: Discrete $\mathbf{X}^{(k)}$ using graph multicut [20].
4: for $i_{k-1} = 1, \ldots, I_{k-1}$ do
5:     if $\mathbf{o}^{(k-1)}_{i_{k-1}}$ not tracked
       and $\mathrm{CNN}_{\mathrm{BBE}}\big(\mathbf{b}^{(k-1)}_{i_{k-1}}\big) > 0.5$ then
6:         add new target trajectory starting with $\mathbf{b}^{(k-1)}_{i_{k-1}}$
7:     end if
8:     if $\mathbf{o}^{(k-1)}_{i_{k-1}}$ assigned to a real candidate, e.g. $\mathbf{o}^{(k)}_{j_k}$ then
9:         if $\mathrm{IoU}\big(\mathbf{p}^{(k-1)}_{i_{k-1}}, \mathbf{b}^{(k)}_{j_k}\big) < t_{\text{dif}}$
           and $\mathrm{CNN}_{\mathrm{BBE}}\big(\mathbf{b}^{(k)}_{j}\big) < 0.5$ then
10:            update target trajectory of $\mathbf{o}^{(k-1)}_{i_{k-1}}$ using $\mathbf{p}^{(k-1)}_{i_{k-1}}$
11:            continue
12:        end if
13:        update target trajectory of $\mathbf{o}^{(k-1)}_{i_{k-1}}$ using $\mathbf{b}^{(k)}_{j_k}$
14:    else
15:        ▷ $\mathbf{b}_F$ is the bounding box of image frame.
16:        if $\mathrm{IoU}\big(\mathbf{p}^{(k-1)}_{i_{k-1}}, \mathbf{b}_F\big) < t_{\text{exit}}$ then
17:            exit target trajectory of $\mathbf{o}^{(k-1)}_{i_{k-1}}$
18:        else
19:            update target trajectory of $\mathbf{o}^{(k-1)}_{i_{k-1}}$ using $\mathbf{p}^{(k-1)}_{i_{k-1}}$
20:        end if
21:    end if
22: end for
```

$\ell1$ normalization layer: for the row (column) in $\mathbf{X}^{(k)(n+1)}$ representing a virtual candidate, only column (row) $\ell1$ normalization is applied. This way, each real candidate can be assigned to only one candidate in consecutive frame including the virtual ones, while a virtual candidate can be assigned to multiple candidates. During optimization, if the affinities of real candidates are smaller than that of the virtual candidates, which represent tracking predictions, the anchor candidates will be automatically associated with the tracking predication. This way, our tracking system integrates the detection and SOT naturally.

4.6. Target Management

On receiving the assignment results, target management handles target entering, exiting and updating. In assignment results, if multiple anchor candidates choose to associate with a virtual candidate, new candidates will be added into candidate sets accordingly. For a virtual candidate not associated with any anchor candidate in assignment results, it will be dropped from the candidate set. Furthermore, if the virtual candidate associated with an anchor candidate in this batch appears as an anchor candidate in the next batch, the appearance feature of the anchor candidate is reused in the next batch in case that the missing detection is caused by occlusion. This SOT process will continue until a confident real candidate is associated.

Table 1. Tracking Performance on the MOT2015 benchmark test set. Best in bold.

	Method	MOTA	MOTP	MT	ML↓	FP↓	FN↓	IDS↓
Offline	CEM [32]	19.3	70.7	8.5%	46.5%	14180	34591	813
	R1TA [41]	24.3	68.2	5.5%	46.6%	6664	38582	1271
	SCNN [27]	29.0	71.2	8.5%	48.4%	**5160**	37798	639
	DAM [23]	32.4	71.8	16.0%	43.8%	9064	32060	**435**
	JMC [21]	**35.6**	**71.9**	**23.2%**	**39.3%**	10580	**28508**	457
Online	RNN [31]	19.0	71.0	5.5%	45.6%	11578	36706	1490
	oICF [22]	27.1	70.0	6.4%	48.7%	7594	36757	**454**
	SCEA [51]	29.1	71.1	8.9%	47.3%	6060	36912	604
	AP [7]	38.5	**71.3**	8.7%	37.4%	**4005**	33203	586
	proposed	**40.6**	71.1	**12.5%**	**34.4%**	4678	**31018**	778

Table 2. Tracking Performance on the MOT2017 benchmark test set. Best in bold.

	Method	MOTA	MOTP	MT	ML↓	FP↓	FN↓	IDS↓
Offline	IOU17 [4]	45.5	76.9	15.7%	40.5%	19993	281643	5988
	bLSTM [24]	47.5	77.5	18.2%	41.7%	25981	268042	2069
	TLMHT [40]	50.6	**77.6**	17.6 %	43.4%	**22213**	255030	**1407**
	jCC [43]	**51.2**	75.9	**20.9%**	**37.0%**	25937	**247822**	1802
Online	GMPHD [25]	39.6	74.5	8.8%	43.3 %	50903	284228	5811
	DMAN [56]	48.2	75.7	**19.3%**	38.3 %	26218	263608	**2194**
	MOTDT [8]	50.9	**76.6**	17.5%	35.7%	24069	**250768**	2474
	proposed	**52.0**	76.5	19.1%	**33.4%**	**14138**	253616	3072

For anchor candidates associated with real candidates, we train a CNN network to further refine their bounding boxes. During MDA, associations are made mainly based on the object center of each candidate. Most MOT tasks concern the actual bounding box enclosure of targets. When an anchor candidate is assigned to a real candidate, two bounding boxes will be associated with it: one from the real candidate itself denoted by $\mathbf{b}_{i_k}^{(k)}$ and the other from the SOT predication of the anchor candidate $\mathbf{p}_{i_{k-1}}^{(k-1)}$. If the bounding box Intersection over Union (IoU) between $\mathbf{b}_{i_k}^{(k)}$ and $\mathbf{p}_{i_{k-1}}^{(k-1)}$ is smaller than a threshold t_{dif}, a CNN is used to evaluate the quality of $\mathbf{b}_{i_k}^{(k)}$. In detail, a CNN-based binary classifier CNN_{BBE} is trained to decide whether a bounding box has IoU larger than a threshold, *e.g.* 0.5, with the category target. The detailed procedure of the target management is listed in Alg. 1.

5. Experiment

We conduct experiments on four popular MOT datasets: MOT2015 [28] and MOT2017 [30] for pedestrian tracking, KITTI-Car [17] and UA-DETRAC [47] for vehicle tracking. All datasets are provided with referred detections from real detectors.

5.1. Experiment Setting

The proposed approach is implemented in PyTorch and runs on a desktop with CPU of 6 cores@3.60GHz and a Titan X GPU. We adapt the SiamFC proposed in [39] as our feature sub-network and use their weights as pre-trained model which is trained on the ILSVRC15 dataset for object detection in video. The CNN $\varphi(\cdot)$ to estimate long-term affinity is constructed with three convolutional layers to map the concatenated spatial aligned feature into affinity score. A ResNet-101 with binary output is adapted for CNN_{BBE}, the pre-trained weights from MaskR-CNN [18] on COCO dataset is used for initialization. Proposed method runs average 0.6 fps on MOT2017 dataset in tracking phase.

For each test sequence in MOT2015 and MOT2017, following their protocol, one or more similar sequences in the training set are used to train a different set of FAMNet and CNN_{BBE} to best adapt the scenario prior. Sequences in the KITTI and UA-DETRAC dataset are all recorded in a similar setting, therefore all training sequences in their datasets are used together to train one set of networks for all test sequences. To train the FAMNet, ground truth bounding boxes are used as input target candidates. Therefore, no virtual candidate or SOT process is enabled during training. When training the CNN_{BBE}, the training samples are collected from both the external detection and ground truth bounding box after random shift and scale. The IoU of bounding boxes with ground truth larger than 0.5 are selected as positive samples while smaller than 0.4 are for negative samples.

To evaluate the performance of the proposed method, the widely accepted CLEAR MOT metrics [2] are reported, which include multiple object tracking precision (MOTP) and multiple object tracking accuracy (MOTA) that combines false positives (FP), false negatives (FN) and the identity switches (IDS). Additionally, we also report the percentage of mostly tracked targets (MT), the percentage of mostly lost targets (ML).

5.2. Evaluation Results

MOT2015. MOT2015 [28] contains 11 different indoor and outdoor scenes of public places with pedestrians as the objects of interest, where camera motion, camera angle and imaging condition vary greatly. The dataset provides detections generated by the ACF-based detector [13]. The numerical results on its test set are reported in Tab. 1. Our approach achieves clearly the state-of-the-art performance. In particular, our method achieves better performance in most metrics than the RNN based end-to-end online methods due to our discriminative higher-order affinity and the optimization method adapted. Our method also surpasses the same R1TA-based method which is with hand-crafted features and affinity metrics.

MOT2017. Similar to MOT2015, MOT2017 [30] contains seven different sequences in both training and test datasets but with higher average target density (31.8 vs 10.6 on the test set), thus is more challenging. MOT2017 also focuses

Table 3. Tracking Performance on the KITTI-Car benchmark test set. Best in bold.

	Method	MOTA	MOTP	MT	ML↓	FP↓	FN↓	IDS↓
Offline	DCO-X [33]	68.1	78.9	37.5%	14.1%	2588	8063	318
	R1TA [41]	71.2	79.2	47.9 %	11.7%	1915	7579	418
	LP-SSVM [44]	77.6	77.8	56.3%	**8.5%**	1239	**6393**	62
	NOMT [9]	**78.1**	**79.5**	**57.2%**	13.2%	**1061**	6421	**31**
Online	RMOT [52]	65.8	75.4	40.2%	9.7 %	4148	7396	209
	mbodSSP [29]	72.7	78.8	48.8%	**8.7%**	1918	7360	**114**
	CIWT [36]	75.4	**79.4**	49.9%	10.3 %	954	7345	165
	proposed	**77.1**	78.8	**51.4%**	8.9%	**760**	**6998**	123

Table 4. Tracking Performance on the UA-DETRAC benchmark test set. All results are averaged over different input detection confidence thresholds. Best in bold.

Method	MOTA	MOTP	MT	ML↓	FP↓	FN↓	IDS↓
CEM [32]	5.1	35.2	3.0%	35.3%	**12341**	260390	**267**
H^2T [48]	12.4	35.7	14.8 %	19.4%	51765	173899	852
CMOT [1]	12.6	36.1	16.1%	18.6%	57886	167111	285
GOG [37]	14.2	**37.0**	13.9%	19.9%	32093	180184	3335
†IOU [4]	19.4	28.9	**17.7%**	18.4 %	14796	171806	2311
proposed	**19.8**	36.7	17.1%	**18.2%**	14989	**164433**	617

† Private detector used.

on evaluating the tracker performance on different detection quality. It provides three different detection inputs from DPM [16], Faster-RCNN [38] and SDP [50], ranked in ascending order by AP. We train seven different sets of networks according to different scenes, without further fitting on the different detections. The numerical results are reported in Tab. 2. The performance of our method is better than or on par with other published state-of-the-art methods.

KITTI-Car. The KITTI dataset [17] contains 21 video sequences in the training set and 29 in the test set for multiple vehicle tracking in street view, where videos are recorded through a camera mounting in front of a moving vehicle. Referred detections from the regionlet [46] detector are used in our experiment. The numerical results on the dataset of our method along with other methods using the same detections are summarized in Tab. 3. Our method again surpasses the hand-crafted feature-based R1TA method, despite the fact that it uses a much larger association batch for off-line tracking. It is worth mentioning that motion affinity plays a more importance role in KITTI than in MOT2015 and MOT2017, since both targets and camera move faster and more regularly in KITTI.

UA-DETRAC. UA-DETRAC dataset [47] is another multiple vehicle tracking dataset with 60 sequences for training and 40 sequences for testing. All sequences are recorded with static camera at a lift-up position near different drive ways in various of weather conditions. We use referred detection from CompACT [5] detector in our experiment. UA-DETRAC reports the average of each MOT metric from a serials of results using different detection confidence thresholds (from 0 to 1.0 with 0.1 step). Comparison with other methods using the same detections are reported in Tab. 4. Proposed method achieves state-of-the-art performance among the published works. Our method also surpasses the IOU tracker which is an offline method and using a private detector.

5.3. Ablation Study

We justify the effectiveness of different modules in proposed method through ablation study as shown in Tab. 5. We conduct the study using the sequences ETH-Pedcross2 and ETH-Sunnyday for testing and ETH-Bahnhof for training. All sequences are from the training set of MOT2015. We start from FAMNet with randomly initialized weights whose tracking performance is referred as "No training" in Tab. 5. Then the network is trained on sequence ETH-Bahnhof. "Training from scratch" stands for the results in this scheme. Training with the limited MOT sequences may lead to overfitting of the feature and affinity sub-network. To increase the generalizability and further boost the performance, we use the weights trained on the ILSVRC15 dataset as initialization then perform fine-tuning on the MOT sequence, which is referred as "Fine-tuning" and is the scheme used in other experiments in this paper. "Without CNN_{BBE}" shows the configuration where detection score is used for bounding box quality estimation instead of the dedicated CNN_{BBE}. Target management without CNN_{BBE} cannot efficiently prevent FPs merging into the tracking results. "Without SOT" in Tab. 5 stands for the case that only detections from external detector are used for association, no SOT prediction is included. By contrast, in the proposed solution, though SOT predictions introduce some FPs, it recovers much more missing candidates and reduces FN greatly.

Table 5. Ablation study on sequences from MOT2015.

Method	MOTA	FP↓	FN↓	IDS↓
No training	35.5	240	4799	202
Training from scratch	44.1	281	4160	97
Without CNN_{BBE}	40.5	518	4227	87
Without SOT	42.0	**200**	4412	99
Fine-tuning (proposed)	**45.2**	259	**4105**	**87**

6. Conclusion

In this paper we proposed a novel deep architecture for MOT, which learns jointly, in an end-to-end fashion, features and high-order affinity directly from the ground truth trajectories. During tracking, predictions from SOT and a dedicated target management are include to further boost tracking robustness. Experiments on the MOT2015, MOT2017, KITTI-Car and UA-DETRAC datasets clearly show the effectiveness of proposed approach.

References

[1] S.-H. Bae and K.-J. Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *TPAMI*, 2018. 1, 2, 8

[2] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *JIVP*, 2008. 7

[3] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 2

[4] E. Bochinski, V. Eiselein, and T. Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, 2017. 7, 8

[5] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *CVPR*, 2015. 8

[6] J. Chen, H. Sheng, Y. Zhang, and Z. Xiong. Enhancing detection model for multiple hypothesis tracking. In *CVPRw*, 2017. 2

[7] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai. Online multi-object tracking with convolutional neural networks. In *ICIP*, 2017. 7

[8] L. Chen, H. Ai, Z. Zhuang, and C. Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *ICME*, 2018. 7

[9] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*, 2015. 8

[10] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. *ICCV*, 2017. 1

[11] R. T. Collins. Multitarget data association with higher-order motion models. In *CVPR*, 2012. 2

[12] A. Dehghan, Y. Tian, P. H. Torr, and M. Shah. Target identity-aware network flow for online multiple target tracking. In *CVPR*, 2015. 2

[13] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *TPAMI*, 2014. 7

[14] L. Fagot-Bouquet, R. Audigier, Y. Dhome, and F. Lerasle. Improving multi-frame data association with sparse representations for robust near-online multi-object tracking. In *ECCV*, 2016. 1

[15] K. Fang, Y. Xiang, X. Li, and S. Savarese. Recurrent autoregressive networks for online multi-object tracking. In *WACV*, 2018. 2

[16] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010. 8

[17] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 7, 8

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017. 7

[19] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008. 2

[20] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, and B. Andres. Efficient decomposition of image and mesh graphs by lifted multicuts. In *ICCV*, 2015. 6

[21] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *TPAMI*, 2018. 1, 7

[22] H. Kieritz, S. Becker, W. Hübner, and M. Arens. Online multi-person tracking using integral channel features. In *AVSS*, 2016. 7

[23] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *ICCV*, 2015. 2, 7

[24] C. Kim, F. Li, and J. M. Rehg. Multi-object tracking with neural gating using bilinear lstm. In *ECCV*, 2018. 7

[25] 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 *AVSS*, 2017. 7

[26] L. Lan, X. Wang, S. Zhang, D. Tao, W. Gao, and T. S. Huang. Interacting tracklets for multi-object tracking. *TIP*, 2018. 1

[27] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *CVPRw*, 2016. 2, 7

[28] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942*, 2015. 7

[29] P. Lenz, A. Geiger, and R. Urtasun. Followme: Efficient online min-cost flow tracking with bounded memory and computation. In *ICCV*, 2015. 8

[30] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831*, 2016. 7

[31] A. Milan, S. H. Rezatofighi, A. R. Dick, I. D. Reid, and K. Schindler. Online multi-target tracking using recurrent neural networks. In *AAAI*, 2017. 2, 7

[32] A. Milan, S. Roth, and K. Schindler. Continuous energy minimization for multitarget tracking. *TPAMI*, 2014. 7, 8

[33] A. Milan, K. Schindler, and S. Roth. Detection-and trajectory-level exclusion in multiple object tracking. In *CVPR*, 2013. 8

[34] A. Milan, K. Schindler, and S. Roth. Multi-target tracking by discrete-continuous energy minimization. *TPAMI*, 2016. 1

[35] P. Ondruska and I. Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. In *AAAI*, 2016. 2

[36] A. Osep, W. Mehner, M. Mathias, and B. Leibe. Combined image-and world-space tracking in traffic scenes. In *ICRA*, 2017. 8

[37] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011. 1, 8

[38] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: towards real-time object detection with region proposal networks. In *NIPS*, 2015. 8

[39] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv:1701.01909*, 2017. 2, 7

[40] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, and J. Yu. Iterative multiple hypothesis tracking with tracklet-level association. *CSVT*, 2018. 7

[41] X. Shi, H. Ling, Y. Pang, W. Hu, P. Chu, and J. Xing. Rank-1 tensor approximation for high-order association in multi-target tracking. *IJCV*, 2019. 2, 3, 5, 7, 8

[42] J. Son, M. Baek, M. Cho, and B. Han. Multi-object tracking with quadruplet convolutional neural networks. In *CVPR*, 2017. 1, 2

[43] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, 2017. 2, 7

[44] S. Wang and C. C. Fowlkes. Learning optimal parameters for multi-target tracking with contextual interactions. *IJCV*, 2017. 8

[45] X. Wang, E. Türetken, F. Fleuret, and P. Fua. Tracking interacting objects using intertwined flows. *TPAMI*, 2016. 1

[46] X. Wang, M. Yang, S. Zhu, and Y. Lin. Regionlets for generic object detection. In *ICCV*, 2013. 8

[47] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking. *arXiv:1511.04136*, 2015. 7, 8

[48] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, and S. Z. Li. Multiple target tracking based on undirected hierarchical relation hypergraph. In *CVPR*, 2014. 8

[49] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *ICCV*, 2015. 1

[50] 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 *CVPR*, 2016. 8

[51] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon. Online multi-object tracking via structural constraint event aggregation. In *CVPR*, 2016. 7

[52] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon. Bayesian multi-object tracking using motion context from multiple objects. In *WACV*, 2015. 8

[53] A. R. Zamir, A. Dehghan, and M. Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*. 2012. 2

[54] A. Zanfir and C. Sminchisescu. Deep learning of graph matching. In *CVPR*, 2018. 5

[55] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008. 2

[56] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online multi-object tracking with dual matching attention networks. In *ECCV*, 2018. 2, 7