# Near-Online Multi-pedestrian Tracking via Combining Multiple Consistent Appearance Cues

| | |
|---|---|
| Journal: | *IEEE Transactions on Circuits and Systems for Video Technology* |
| Manuscript ID | TCSVT-04266-2020 |
| Manuscript Type: | Transactions Papers - Regular Issue |
| Date Submitted by the Author: | 09-Mar-2020 |
| Complete List of Authors: | Feng, Weijiang; National University of Defense Technology, College of Computer<br>Lan, Long; National University of Defense Technology, College of Computer<br>Luo, Yong; Nanyang Technological University,<br>Yu, Yue; National University of Defense Technology, College of Computer<br>Zhang, Xiang; National University of Defense Technology, College of Computer<br>Luo, Zhigang; National University of Defense Technology, College of Computer |
| EDICS: | 2.5.3 □ Multi-Target Tracking < 2.5 □ Tracking < 2 □ IMAGE/VIDEO ANALYSIS AND COMPUTER VISION, 2.5.4 □ Human Tracking < 2.5 □ Tracking < 2 □ IMAGE/VIDEO ANALYSIS AND COMPUTER VISION |
| | |

SCHOLARONE™
Manuscripts

# Near-Online Multi-pedestrian Tracking via Combining Multiple Consistent Appearance Cues

Weijiang Feng, Long Lan, Yong Luo, Yue Yu, Xiang Zhang, and Zhigang Luo, *Member, IEEE*

***Abstract*—An important cue for multi-pedestrian tracking in video is the consistent appearance of an individual for quite a while. In this paper, we address multi-pedestrian tracking by learning a robust appearance model from the paradigm of tracking by detection. To separate detections of different pedestrians while assembling detections of the same pedestrian, we take advantage of the cue of consistent appearance and exploit three types of evidence from the recent, past and near-future. Existing online approaches only exploit the detection-to-detection and sequence-to-detection metrics, which focus on the recent and past appearance patterns respectively, while the future pedestrian appearance is simply ignored. This drawback is remedied in this paper by further considering the sequence-to-sequence metric, which resorts to near-future appearance presentation. Adaptive combination weights are learned to fuse these three different metrics. Moreover, we propose a novel Focal Triplet Loss to make the model focus more on hard examples than the easy ones. We demonstrate that this can significantly enhance the discriminating power of the model compared with treating every sample equally. Effectiveness and efficiency of the proposed method is verified by conducting comprehensive ablation studies and comparing with many competitive (offline/online/near-online) counterparts on the MOT16 and MOT17 Challenges.**

***Index Terms*—multi-pedestrian tracking, sequence-to-sequence metric, adaptive weights, Focal Triplet Loss**

## I. Introduction

MULTIPLE pedestrian tracking estimates the locations of all concerned pedestrians in a scene and maintain their identities across consecutive frames to produce their respective trajectories. Multiple pedestrian tracking in videos is one of the basic components in various computer vision applications, such as video surveillance, autonomous driving, human behavior analysis. Recent advances in object detection [1], [2] make it possible to generate high-quality detection responses of pedestrians in the form of bounding boxes. With these bounding boxes, the currently predominant approaches to multiple pedestrian tracking follow the paradigm of tracking-by-detection [3], which studies how to associate discrete bounding boxes distributed in different frames to produce a complete trajectory for each individual. Through this kind of data association formulation, the multiple pedestrian tracking community has made significant progress. However, multiple pedestrian tracking is still a challenging task due to the presence of frequent occlusions and interactions, sudden missing and appearing, abrupt appearance changes, pose variations, real-time requirements, etc. The tracking-by-detection paradigm contains two critical components: the *affinity model* and the *data association*. The first component measures how likely two concerned detection hypotheses share the same identity. Based on their similarity, the second component focuses on how to link these detections into longer tracklets and finally the full trajectories. To achieve a high-performance multiple pedestrian tracking algorithm, both the components should be carefully considered.

Data association is widely recognized as the backbone of multiple pedestrian tracking. Many works have been done in this field. Generally, according to different scenes, data association can be categorized into two types, namely, *offline* [4]–[8] and *online* approaches [9]–[11]. Online approaches, which associate detections by only looking back to the previous frames, are well-suited for time-critical applications. Due to the recursive nature, the association may be prone to many identity switches and is difficult to correct, especially for scenes where pedestrians frequently interact with each other. On the other hand, offline methods, which utilize all video frames, achieve a higher data association accuracy than their online counterparts. However, the shortage of offline methods is that they suffer a temporal delay. Some recent methods focus on near-online methods [12], [13], which consider previous frames and a few near-future frames, offering a compromise, and achieving promising results at the cost of only a slight temporal delay. Inspired by the benefits of near-online methods, we focus on near-online approaches in this paper.

As to affinity models, most methods emphasize the similarity between detection and detection [14]–[16] or detection and tracklet [10], [11], [17] (consisting of several already associated detections), as shown in Figure 1. These two settings are easy to achieve in the case of online tracking mode, as the robust detections or tracklets can be readily obtained. However, appearance modeling only utilizes the past frames is not enough, the unexpected occlusions cut off the link of pedestrians from successive frames and require the trackers to look into the future frames. Similarity measurement between sequences [18]–[20] commonly happens in the offline case where short tracklets can be constructed beforehand after given the whole frames. As the short tracklets are generated in advance and supposed to be robust, there is no rollback mechanism, even some unreasonable associations are

W. Feng and Z. Luo are with Science and Technology on Parallel and distributed Processing, National University of Defense Technology, Changsha 410073, China (email: fengweijiang14@nudt.edu.cn; zgluo@nudt.edu.cn).

L. Lan and X. Zhang are with Institute for Quantum Information & State Key Laboratory of High Performance Computing (HPCL), National University of Defense Technology, Changsha 410073, China (email: long.lan@nudt.edu.cn; zhangxiang08@nudt.edu.cn).

Y. Luo is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (email: yluo180@gmail.com).

Y. Yu is with College of Computer, National University of Defense Technology, Changsha 410073, China (email: yuyue@nudt.edu.cn).

*Corresponding author: Long Lan.*

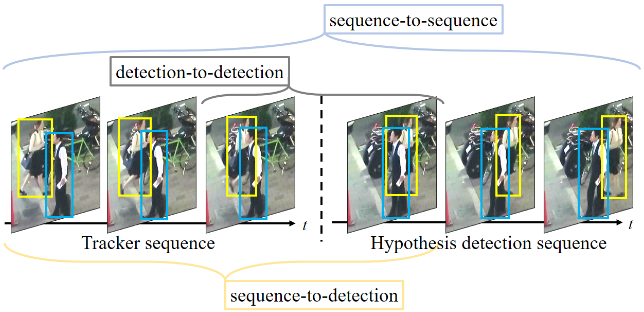

Fig. 1. Three different kinds of affinity models. Detections in the left of the vertical dotted line are from the past frames and form the tracker sequence. Detections in the right of the vertical dotted line are from current and near-future frames and form the hypothesis detection sequence. This figure shows the information used by three affinity models.

observed in the later step. An obvious defect is that these pre-constructed short tracklets may exist false positives, thus the measurement between tracklets are unreliable. A very recent work [19] exploits the cleaving of unreliable tracklets to obtain reliable shorter tracklets that contains the same identity when associating the tracklets into a much longer trajectory in the offline setting. Inspired by the recent advances, we study the near-online affinity modeling, which, on one hand, learns sequence to sequence affinity with several future frames, as shown by the light blue bracket in Figure 1, on the other hand, adopts a rollback mechanism to reconstruct the short tracklets if the detections contained in them take different pedestrian identities.

Specifically, in this paper, we firstly design a novel sequence-to-sequence appearance measurement in a near-online tracking model to handle the frequent occlusions of pedestrians. We assume that the first sequence is constructed from the past and is robustly assigned to a specific pedestrian, the second sequence is temporally generated from the near future and can rollback into detections after the measurement finished. By this setting, we thus can extend our affinity model to measure appearance consistency of occlusion in the online tracking case.

To construct a robust affinity model in the setting of near-online. Our final affinity model incorporates similarity from three different aspects. The detection-to-detection metric utilizes the most recent and informative evidence, the sequence-to-detection metric takes advantage of the historical information, and the sequence-to-sequence metric looks ahead to a few future frames. The explored three metrics work together to enhance the discriminative power of the affinity model. Specifically, the detection-to-detection metric measures the pedestrian similarity from two nearby frames, it is the simplest way of affinity model. The sequence-to-detection metric draws support from the previous frames to enhance the robustness as the appearance of an individual pedestrian hold consistent for quite a while. The sequence-to-sequence metric in our setting computes the similarity between tracked sequence and detection sequence. The tracked sequence is comprised of detection responses in the past frames, while the hypothesis detection sequence contains detection responses in the near future frames. We believe a specific pedestrian will also keep its appearance in the near future even after occlusion. Since detection-to-detection affinity model, sequence-to-detection affinity model, and sequence-to-sequence affinity model all provide cues from different perspectives for computing similarity between trackers and detection responses, we combine these three kinds of metrics into a synthesized affinity model. In order to find an optimal combination of these affinity models, we take advantage of Hedge algorithm [21] to adaptively learn the combination weights.

When considering associating a tracker $T$ with two detections $d_1$ and $d_2$ based on similarities $s(T, d_1)$ and $s(T, d_2)$ (assuming $T$ and $d_1$ share the same identity, while $T$ and $d_2$ have different identities), it is crucially important for the affinity models to let $s(T, d_1)$ be greater than $s(T, d_2)$. To achieve this, we utilize triplet loss [22] to train our affinity models. The triplet loss learns an embedding space for the pedestrian appearance in which the distances (inverse of similarities) between samples with the same identity are much shorter than those of samples with different identities. What's more, many works [11], [23] have shown that mining hard training triplets is beneficial to the triplet loss based models. For classification tasks, researchers proposed Focal Loss [24] that automatically down-weights the contribution of easy training samples and makes the model focus more on hard samples. In this paper, we apply the idea of Focal Loss to triplet loss and propose Focal Triplet Loss to adaptively improve the contribution of hard triplets training samples to the loss function.

Experimental results on challenging MOT16 and MOT17 benchmarks [25] demonstrate the effectiveness of the proposed affinity model in our near-online setting, even only using the simple greedy data association method [26].

Our main contributions in this paper can be listed:

1) Besides the conventional detection-to-detection and sequence-to-detection affinity metrics, we propose a sequence-to-sequence affinity metric that computes the similarity between known track sequences and hypothesis detection sequences comprised of detections in the near future frames. Based on them, we optimally combine the evidence provided by these three kinds of affinity metrics into a synthesized appearance model using the Hedge algorithm.
2) To assure the learned affinity metric has a better performance to distinguish appearance similar pedestrians, we extend Focal Loss to triplet loss and propose Focal Triplet Loss to learn our affinity models and prove the Focal Triplet loss really relieve the drifts of multi-pedestrian tracking.
3) A comprehensive ablation experiment is exerted to study the significance of each component of our appearance model. Our final near online tracker even with the simplest data association achieves a very promising performance.

The rest of the paper is organized as follows: In section 2, we review the related literature. In section 3, we present our tracking framework. Section 4 details the implementation, results of our approach on tracking benchmark challenge and ablation analysis. Conclusions are drawn in Section 5.

## II. Related Works

### A. Affinity Models

A proper choice of affinity measure is crucial for achieving promising tracking performance. Many works explicitly learn affinity metrics. Siamese network [27] and triplet network [28] are intuitive methods to measure the similarity between two objects. Leal-Taixé *et al.* [14] introduce a two-stage learning scheme to compare pairs of detections. They first train a Siamese convolutional neural network (CNN) to learn local spatial-temporal descriptors, then combine a set of contextual features with the CNN output in a gradient boosting framework to generate the final affinity metrics. Wang *et al.* [15] jointly learn Siamese CNNs and temporally constrained metrics to obtain an appearance-based tracklet affinity model. Son *et al.* [16] propose Quadruplet CNN by generalizing Siamese and triplet networks. Quad-CNN uses quadruplet losses to enforce an additional constraint that makes temporally adjacent detections more closely located than the ones with large temporal gaps. Ma *et al.* [19] propose a tracklet-to-tracklet based Siamese Bi-Gated Recurrent Unit (GRU) to compute the affinity between tracklets. Feng *et al.* [29] use a ReID network that takes a modified version of GoogLeNet Inception-v4 as its backbone CNN to compute detection-to-detection similarity. Most of the previous works address only one type of metric when measuring the similarity of appearance.

Recurrent Neural Networks (RNNs) are able to summarize the general characteristics of images from the same tracklet and have been successfully applied in multi-pedestrian tracking for tracklet-to-detection affinity measure. Milan *et al.* [30] first train a Long Short-Term Memory (LSTM) in a fully end-to-end manner for online multi-pedestrian tracking. Sadeghian *et al.* [17] combines appearance, motion and interaction cues into a unified RNN network. Kim *et al.* [10] propose a novel Bilinear LSTM to improve the learning of long-term appearance models.

In contrast to our approach, most of the aforementioned methods compute detection-to-detection or sequence-to-detection affinity metrics. The works [18]–[20] that compute the sequence-to-sequence affinity metric work in an offline setting. While we use the sequence-to-sequence affinity measure in a near online manner.

### B. Near-online Data Association

Data association is an important procedure of all multi-pedestrian tracking methods following the tracking-by-detection paradigm. The data association is usually formulated to various optimization problems. Most of offline approaches are variants of graph segmentation problem, and many online processing methods use Hungarian Algorithm [31] to solve a bipartite graph matching problem. The near online data association [12], [13] considers tracking between targets and detections in a temporal window, and performs this process repeatedly at every frame. In contrast, we use detections in the future frames to enhance the affinity measure between existing tracks and detections in the current frame, and conduct data association only for detections in the current frame. In practice, a greedy approach is often sufficient. In this paper, we simply use a greedy scheme [26] and focus on obtaining a promising affinity model.

### C. Triplet Loss and Its Variants

Weinberger *et al.* [22] propose the large margin nearest neighbor loss, and this is treated as the original triplet loss. FaceNet [32] formally defines margin based triplet ranking loss and its soft version. Balntas *et al.* [23] presents in-triplet hard negative mining to ensure the hardest negative inside the triplet is used for calculating the loss. Yi *et al.* [33] propose to generate the triplet with the hardest positive and hardest negative for each anchor sample in the min-batch. Zhang *et al.* [34] propose the most similar idea to ours. They combine focal loss with triplet loss for the task of person re-identification. Specifically, they map the original distance in the Euclidean space to an exponential kernel space, thus the hard triplets are penalized much more than the easy ones.

In this paper, we propose the Focal Triplet Loss to automatically increase the importance of the hard triplet samples without reducing the importance of simple ones. Different from [34], which has a large scale training datasets and thus increases the importance of hard examples meanwhile reduces the importance of easy examples for person re-identification, we apply Focal Triplet Loss to the task of efficiently training affinity models. Since the training dataset in our problem is limited, Focal Triplet Loss in our work does not reduce the importance of easy training samples compared with the original triplet loss.

### D. End-to-end Tracking Model

Recently, some researchers make efforts to model the object's affinities and the data association across frames by an end-to-end network, and obtain a promising tracking performance. Deep Affinity Network (DAN) [35] jointly learns targets' affinity and their association in a pair of frames in an end-to-end fashion, where the appearance modeling accounts for hierarchical feature learning of objects and their surroundings at multiple levels of abstraction, and association is estimated under exhaustive permutations of compact features. FAMNet [36] refines Feature extraction, Affinity estimation and Multi-dimensional assignment in a single deep network. The feature sub-network extracts features for detections, and the affinity sub-network estimates the higher-order affinity for all association hypothesis. All layers of FAMNet are designed differentiable and thus optimized jointly. Ma *et al.* [37] propose an end-to-end Deep Association Network by combining a CNN, a Motion Encoder, and a Graph Neural Network (GNN). The CNN is utilized to extract appearance features of bounding boxes, the Motion Encoder is designed to describe the bounding box information such as position, size, and shape, and the GNN replaces hand-crafted algorithm for graph optimization.

In this paper, the appearance feature extraction sub-model and affinity sub-model can be jointly designed by an end-to-end network. However, due to the limited training examples in the MOT sequences, we separate the feature extraction sub-model and the affinity sub-model by designing one feature extraction sub-network and three affinity sub-networks rather than three joint end-to-end "feature extraction - affinity" networks. The three independent affinity sub-models with a feature extraction sub-models obtain lower training costs than training them end to end. We note the separately learned affinity followed even by the simplest data association method could achieve a very promising tracking performance.

## III. The Proposed Framework

The performance of MOT algorithms relies heavily on the quality of affinity measure. However, affinity models in online approaches are prone to identity switches, because these models only consider information up to the current frame. On the other hand, affinity models in offline methods pose a major limitation for time-critical applications. In this paper, we devote to the near-online approach to make a compromise.

In our framework, we use trackers to record trajectories of tracked objects. Each tracker is an ordered set of detections, and we denote each tracker by $T = \{d_t\}$, where $d_t = [d_t(x), d_t(y), d_t(w), d_t(h), d_t(img)]$, $t$ is the frame index, $d_t(x), d_t(y), d_t(w), d_t(h)$ are the top-left coordinate, width and height of the corresponding bounding box of detection $d_t$, and $d_t(img)$ is its corresponding patch in image $I_t$.

### A. Overall Design

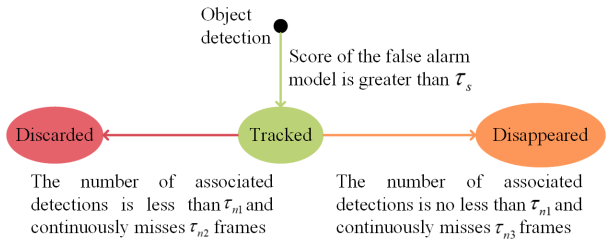
Object detection
Score of the false alarm model is greater than $\tau_s$
Discarded
Tracked
Disappeared
The number of associated detections is less than $\tau_{n1}$ and continuously misses $\tau_{n2}$ frames
The number of associated detections is no less than $\tau_{n1}$ and continuously misses $\tau_{n3}$ frames

Fig. 2. State transitions of trackers.

Each time a detection response $d$ is associated by a tracker $T$, the detection $d$ will be appended to the tracker $T$. Trackers may be in different states, as shown in Figure 2. Each object detection will firstly be scored by the false alarm model to determine whether this detection is a false alarm no not. We will explain the false alarm model in section III-B. If the score is greater than $\tau_s$, we will initialize a tracker using this object detection, and the state of the tracker is "tracked". Despite the false alarm model, false positive detections may still initialize trackers. Thus, we use heuristics to remove these false trackers. Specifically, if a tracker in "tracked" state continuously misses $\tau_{n2}$ frames, but the number of its associated detections is less than $\tau_{n1}$, then the state of the tracker will be converted to "discarded". Some objects may be occluded by other objects or may leave the scene. The transition from "tracked" state to "disappeared" state accounts for this situation. If the number of associated detections of a tracker is no less than $\tau_{n1}$, but the tracker continuously misses $\tau_{n3}$ frames, we will transition the tracker from "tracked" state to "disappeared" state. When conducting tracking, we only consider trackers in the "tracked" state, and ignore trackers in the "disappeared" state. While the final tracking results will include both the "tracked" trackers and the "disappeared" trackers.

Figure 3 shows the main parts of our proposed near-online multi-pedestrian tracking framework. We outline the whole framework in the following steps:

- Step 1. Initially, the set of trackers $\mathcal{T}$ is empty and $t = 0$.
- Step 2. At time $t+1$, the current detection set $\mathcal{D}_{t+1}$ contains all detection responses in image $I_{t+1}$. For the $j$-th detection $d_{t+1}^j, j \in [1, |\mathcal{D}_{t+1}|]$ in $\mathcal{D}_{t+1}$, a false alarm model calculates its score, and detections whose score is smaller than $\tau_s$ are removed from $\mathcal{D}_{t+1}$.
- Step 3. For the $j$-th remaining detection $d_{t+1}^j$ in $\mathcal{D}_{t+1}$ (Here, we reuse this symbol to denote detection sets after false alarms being removed), construct one hypothesis detection sequence $D_{t+1}^j$ by looking forward $K$ frames.
- Step 4. Use a Re-ID network to extract appearance features for detections in trackers and detections in hypothesis detection sequences.
- Step 5. Compute similarity $S_{dvd}\left(d_t^i, d_{t+1}^j\right)$ between the last detection $d_t^i$ of tracker $T^i$ in "tracked" state and detection $d_{t+1}^j$ using detection-to-detection affinity model; Compute similarity $S_{svd}\left(T^i, d_{t+1}^j\right)$ between $T^i$ and $d_{t+1}^j$ using sequence-to-detection affinity model; Compute similarity $S_{svs}\left(T^i, D_{t+1}^j\right)$ between $T^i$ and hypothesis detection sequence $D_{t+1}^j$ using sequence-to-sequence affinity model. Then compute the synthesised similarity score $S\left(T^i, d_{t+1}^j\right)$ between tracker $T^i$ and detection $d_{t+1}^j$ with the optimal combination weights learned by the Hedge algorithm.
- Step 6. Associate detections with trackers in "tracked" state using greedy data association algorithm based on similarity score $S$.
- Step 7. Append associated detections to their matched trackers, and update states of Kalman Filters of these trackers using the new matched detections. For trackers in "tracked" state that do not match, predict their position using their Kalman Filters. For isolated detection results, initialize new trackers and corresponding Kalman Filters. Then update the state of all trackers according to Figure 2.
- Step 8. Repeat steps from 2 to 7 for the next frame by setting $t = t+1$, until no more frames arrive.

For simplicity, we do not contain the false alarm model and Kalman Filters in Figure 3. Note that, we design the affinity model in a near-online manner to improve its robustness by introducing a sequence-to-sequence affinity model, but we conduct the data association procedure in an online manner.

### B. False Alarm and False Negative Model

False positive detection responses, or false alarms due to the inaccuracy of object detectors are detrimental to multi-pedestrian tracking in several ways. Firstly, false positives are not desired for some applications such as crowd analysis or

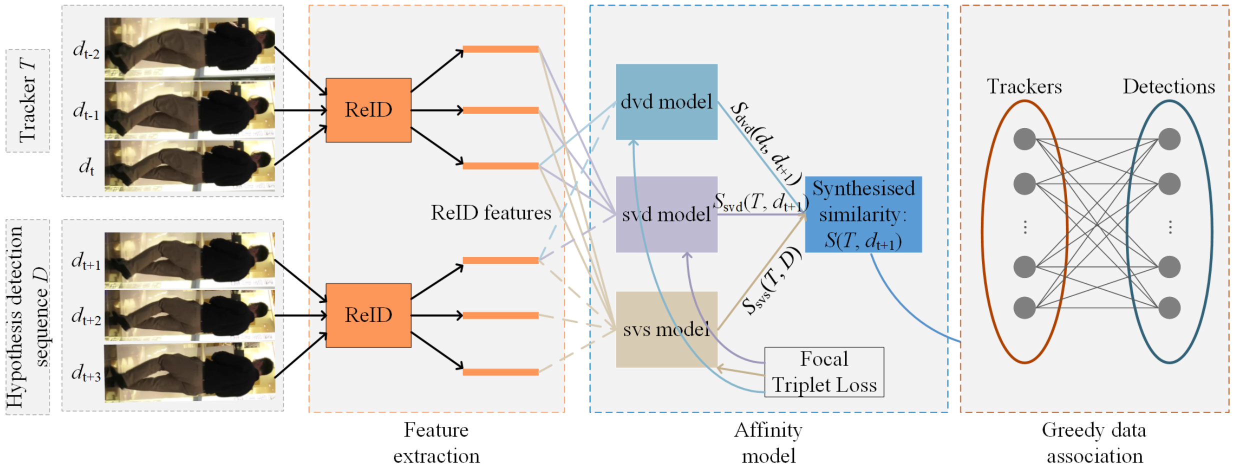

Fig. 3. The main parts of our proposed near-online multi-pedestrian tracking framework. Current time step is $t+1$. Based on detection $d_{t+1}$, we construct a hypothesis detection sequence $D$. Firstly, we extract ReID features for all detections in the tracker $T$ and the hypothesis sequence $D$. Secondly, we use the detection-to-detection affinity model, sequence-to-detection affinity model, and sequence-to-sequence affinity model to compute the synthesised similarity score between tracker $T$ and detection $d_{t+1}$. All these three affinity models are trained with the novel Focal Triplet Loss. Thirdly, we conduct a greedy data association based on the synthesized similarity score.

people motion analysis, where very reliable trajectories are desirable. Secondly, trackers may be wrongly associated with false alarms, and this causes undesirable identity switches, false positives, and false negatives. Thirdly, it costs extra computation and memory resources for both affinity models and data association algorithms. Thus, it is a desirable property of multi-pedestrian tracking approaches to remove false alarms. Many researchers remove false alarms using heuristics, such as dropping detections whose confidence scores are lower than a threshold, or dropping tracked targets that do not associate detection responses for a period of time, according to the intuition that false positive objects cannot be consistently detected. Xiang *et al.* [38] train a binary Support Vector Machine (SVM) to predict whether a detection response is a false alarm or not. The inputs of the SVM model include 2D coordinates, width, height and score of detection responses. Though these heuristics or SVM model do contribute to false alarms' removal, these methods are not robust enough to handle complex scenes, since these methods neglect the spatial information and appearance features of objects.

In this paper, to remove false alarms, we adopt the object classification model proposed by Long *et al.* [39]. Their object classifier uses a region-based fully CNN (R-FCN) [40] as the backbone architecture, and employs the position-sensitive region of interest pooling layer to explicitly encode spatial information. For each detection response, the false alarm model calculates a score, and we discard detection response whose score is less than a threshold $\tau_s$.

On the other hand, despite the advances of detectors, they may still miss some detections, resulting in false negatives. To reduce the number of false negatives caused by missed detections, we propose to maintain a Kalman Filter for each tracked pedestrian. For a pedestrian tracker without assigning a detection at the current time step, it will predict a bounding box using its corresponding Kalman Filter. In this way, we significantly reduce the number of false negatives.

### *C. Appearance Representation with ReID Features*

Appearance information is an important cue to build similarity scores. Convolutional neural networks (CNN) have been well studied and employed to encode appearance cue. These deeply learned features by a data-driven approach typically outperform traditional hand-crafted features. In this paper, to learn the similarity function, we employ a CNN to extract feature vectors from RGB images.

Multi-pedestrian tracking has a very close relationship with the person re-identification. Generally, person re-identification addresses the person matching cross different cameras while most multi-pedestrian tracking methods only focus on a single camera. Thus, appearance modeling in person re-identification faces more challenges as many inconsistencies exist in different cameras.

To better discriminate different pedestrians, we utilize the ReID network [41] to extract appearance features for all detection responses. The ReID network $H_{reid}$ consists of a sub-network of the first version of GoogLeNet [42] and several branches of part-aligned fully connected layers. We refer to [41] for more details on the network architecture. Given an RGB image $d\left(img\right)$ of detection $d$, the appearance representation is formulated as $f_{reid} = H_{reid}\left(d\left(img\right)\right)$. In implementation, we extract appearance features from all detections using the ReID network $H_{reid}$ as a pre-processing step, and store the appearance feature, rather than the RGB image for each detection.

### *D. Affinity Models with Focal Triplet Loss*

*1) Triplet Loss Revisit:* Triplet loss aims to train a model as an embedding function $g_\theta : \mathbb{R}^{d1} \rightarrow \mathbb{R}^{d2}$. During the training procedure, samples in training datasets are formed as a set of triplets $\left\{\left\langle a^i, p^i, n^i\right\rangle\right\}$, where $a$ is called anchor sample, $p$ is called positive sample, and $n$ is called negative sample respectively. $\left\langle a^i, p^i\right\rangle$ is a positive pair with the same identity, and $\left\langle a^i, n^i\right\rangle$ is a negative pair with different identities. Let

$d_{s,t}$ denote the Euclidean distance of sample $s$ and $t$ in the embedding space, i.e., $d_{s,t} = \sqrt{\|g_\theta(s) - g_\theta(t)\|^2}$, then the triplet loss for triplet $\langle a, p, n\rangle$ is formulated as:

$$L_{Tri} = \max\left(d_{a,p} - d_{a,n} + m, 0\right), \tag{1}$$

where $m > 0$ is a predefined margin. Triplet loss tries to make the distance in the embedding space between the anchor sample $a$ and positive sample $p$ closer than that of the anchor sample $a$ and negative sample $n$, by at least margin $m$.

*2) Focal Triplet Loss:* One shortcoming of triplet loss is that the number of triplets increase cubically with the number of training dataset, making the training of all possible triplets impractical. Thus, it is desirable to focus on hard examples. Motivated by Focal loss [24] which automatically focuses the model on hard examples and down-weights the contribution of easy examples, we propose Focal Triplet Loss to automatically up-weight the hard triplets without down-weight the easy triplets. The formulation of Focal Triplet Loss for triplet $\langle a, p, n\rangle$ is as following:

$$L_{FTL} = \max\left(d_{a,p} - d_{a,n} + m, 0\right) \cdot \max\left(d_{a,p} - d_{a,n}, 1\right)^{\lambda}, \tag{2}$$

where $\lambda \geq 0$ is the focusing parameter.

Similar to Focal Loss, we adds a scaling factor $\max\left(d_{a^i,p^i} - d_{a^i,n^i}, 1\right)^{\lambda}$ to the standard triplet loss. Hard triplets mean that larger $d(a,p)$ and smaller $d(a,n)$. With the help of the scaling factor, the loss of hard triplets will be enlarged, while the loss of easy triplets keeps unchanged. The harder the input triplets are, the more penalty they will get relatively. As a result, Focal Triplet Loss can automatically focus on "hard" triplets. Different from Focal Loss which takes effect by reducing the relative loss of easy samples, the proposed Focal Triplet Loss takes effect via increasing the relative loss of hard samples.

Based on Focal Triplet Loss, we train our detection-to-detection, sequence-to-detection, and sequence-to-sequence affinity models.

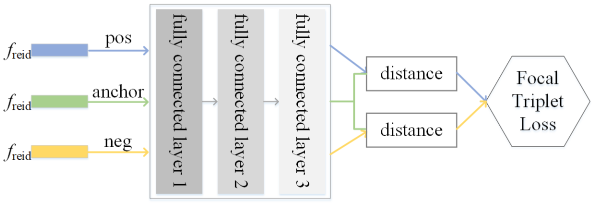

Fig. 4. Network architecture for the detection-to-detection affinity model.

*3) Detection-to-detection Affinity Model:* This affinity model uses information up to the current frame $t+1$. Assume a tracker $T^i$ in "tracked" state with the latest detection $d^i_t$, and two detection responses $d^j_{t+1}, d^k_{t+1}$ at current frame $t+1$, whose identities are the same as and different from that of the tracker, we use the network architecture shown in Figure 4 to compute the detection-to-detection similarity score $S_{dvd}\left(d^i_t, d^j_{t+1}\right)$, where we take $d^i_t$ as a proxy to $T^i$. During the training phase, inputs to the network are ReID features of detections $d^i_t, d^j_{t+1}, d^k_{t+1}$, and we consider these features as anchor sample, positive sample and negative sample of a triplet respectively. We use a three-layers multi-layer perceptron (MLP) as the embedding function $g_\theta$.

Since the goal of the network is to compute similarity score between each pair of tracker $T^i$ and detection $d^j_{t+1}$, after getting the distance $d_{d^i_t, d^j_{t+1}}$ between the latest detection $d^i_t$ of tracker $T^i$ and the detection $d^j_{t+1}$ in the embedding space during the evaluation phase, we calculate the similarity score between $d^i_t$ and $d^j_{t+1}$ as follows,

$$S_{dvd}\left(d^i_t, d^j_{t+1}\right) = \exp\left(-d_{d^i_t, d^j_{t+1}}\right). \tag{3}$$

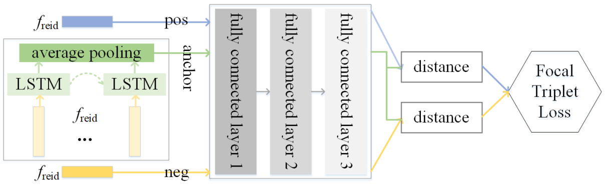

Fig. 5. Network architecture for the sequence-to-detection affinity model.

*4) Sequence-to-detection Affinity Model:* This affinity model also uses information up to the current frame $t+1$. Assume a tracker $T^i$ in "tracked" state, and two detection responses $d^j_{t+1}, d^k_{t+1}$ at current frame $t+1$, whose identities are the same as and different from that of the tracker, we use the network architecture shown in Figure 5 to compute the sequence-to-detection similarity score $S_{svd}\left(T^i, d^j_{t+1}\right)$. During the training phase, inputs to the network are ReID features of the latest $K$ detections $d^i_{t-K+1}, \cdots, d^i_t$ of tracker $T^i$ and ReID features of detections $d^j_{t+1}, d^k_{t+1}$. To deal with the length $K$ sequence of the tracker, we use a LSTM RNN followed by an average pooling layer, and we consider its output as the anchor sample of a triplet. The ReID features of $d^j_{t+1}, d^k_{t+1}$ are considered as positive sample and negative sample of a triplet respectively. Similar to the detection-to-detection network, we use a three-layers MLP as the embedding function $g_\theta$. After getting the distance $d_{T^i, d^j_{t+1}}$ between the tracker $T^i$ and the detection $d^j_{t+1}$ in the embedding space during the evaluation phase, we calculate the similarity score between the tracker $T^i$ and detection $d^j_{t+1}$ as follows,

$$S_{svd}\left(T^i, d^j_{t+1}\right) = \exp\left(-d_{T^i, d^j_{t+1}}\right). \tag{4}$$

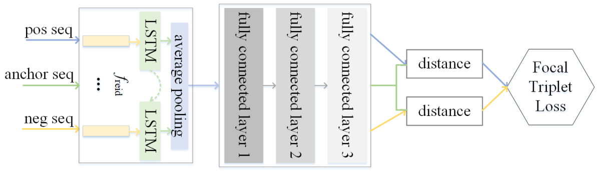

Fig. 6. Network architecture for the sequence-to-sequence affinity model.

*5) Sequence-to-sequence Affinity Model:* This affinity model uses near future information up to the frame $t+K$. Assume a tracker $T^i$ in "tracked" state, and two detection responses $d^j_{t+1}, d^k_{t+1}$ at current frame $t+1$, whose identities are the same as and different from that of the tracker, we use the network architecture shown in Figure 6 to compute the sequence-to-sequence similarity score $S_{svs}\left(T^i, D^j_{t+1}\right)$. Here we use the sequence $D^j_{t+1}$ as a proxy to detection $d^j_{t+1}$.

By looking forward $K$ frames, we first construct hypothesis detection sequence $D^j_{t+1}$and $D^k_{t+1}$ based on detections $d^j_{t+1}$ and $d^k_{t+1}$ respectively, this process is shown in **Algorithm 1**. During the training phase, inputs to the network are ReID features of the latest $K$ detections $d^i_{t-K+1}, \cdots, d^i_t$ of tracker $T^i$, ReID features of detections in sequence $D^j_{t+1}$, and ReID features of detections in sequence $D^k_{t+1}$, which can be considered as anchor sample, positive sample, and negative sample respectively. The embedding function $g_\theta$contains a LSTM RNN followed by an average pooling layer, and a three-layers MLP. After getting the distance $d_{T^i, D^j_{t+1}}$ between the tracker $T^i$ and the hypothesis detection sequence $D^j_{t+1}$ in the embedding space during the evaluation phase, we calculate the similarity score between $T^i$ and $D^j_{t+1}$ as follows,

$$S_{svs}\left(T^i, D^j_{t+1}\right) = \exp\left(-d_{T^i, D^j_{t+1}}\right). \tag{5}$$

**Algorithm 1** Hypothesis detection sequence construction algorithm.

```
Require: Object detection sets $\{\mathcal{D}_{t+k}\}_{k=1}^{K}$ after removing false alarms, and the number of frames of current video $N$.
Ensure: Hypothesis detection sequences $\left\{D^i_{t+1}\right\}_{i=1}^{|\mathcal{D}_{t+1}|}$.
Initialize trackers $T^i = \left\{d^i_{t+1}\right\}$, null detection $d_\phi$ with zero appearance features
for $k = 2$ to $K$ do
    if $t + k > N$ then
        break
    end if
    for $i = 1$ to $|\mathcal{D}_{t+1}|$ do
        Compute similarity scores between $T^i$ and all detections in $\mathcal{D}_{t+k}$ using the detection-to-detection affinity model
    end for
    Conduct greedy data association based on similarity scores
end for
for $i = 1$ to $|\mathcal{D}_{t+1}|$ do
    Denote the number of $T^i$ as $n$
    if $n < K$ then
        Append $(K - n)$ $d_\phi$ to $T^i$
    end if
    $D^i_{t+1} = T^i$;
end for
```

### E. Synthesized Similarity Score

With the detection-to-detection similarity score $S_{dvd}\left(d^i_t, d^j_{t+1}\right)$, the sequence-to-detection similarity score $S_{svd}\left(T^i, d^j_{t+1}\right)$, and the sequence-to-sequence similarity score $S_{svs}\left(T^i, D^j_{t+1}\right)$, we define the synthesised similarity score between the tracker $T^i$ and the detection $d^j_{t+1}$ as follows:

$$\begin{array}{r}S\left(T^i, d^j_{t+1}\right) = \alpha S_{dvd}\left(d^i_t, d^j_{t+1}\right) + \beta S_{svd}\left(T^i, d^j_{t+1}\right) \\ +\gamma S_{svs}\left(T^i, D^j_{t+1}\right)\end{array}, \tag{6}$$

where $0 \leq \alpha, \beta, \gamma \leq 1$ and $\alpha + \beta + \gamma = 1$. The first two terms in the right hand side of equation (6) uses online affinity model, while the third term utilizes near-online model. If $\alpha = 1$, equation (6) degrades to the common detection-to-detection similarity model. If $\beta = 1$, equation (6) degrades to the sequence-to-detection similarity model. By setting $\alpha = 0$ or $\beta = 0$, we switch off the detection-to-detection similarity model or the sequence-to-detection similarity model. By setting $\gamma = 0$, we switch off the sequence-to-sequence similarity model.

To learn the optimal combination weights, that are $\alpha, \beta, \gamma$, we apply the well-known Hedge algorithm [21]. It is an online learning algorithm, and the key idea is to maintain a dynamic weight distribution over a set of strategies. During the online learning process, the distribution is updated by exponentially decreasing the weight of every strategy with respect to its suffered loss. The learning process is shown in **Algorithm 2**, in which $\mathbb{I}$ is an indicator function.

**Algorithm 2** Hedge algorithm for learning the combination weights.

```
Require: Discount weight $\eta \in (0, 1)$, initial combination weights $\alpha = \beta = \gamma = \frac{1}{3}$, and training data of size $T$.
Ensure: Optimal combination weights $\alpha, \beta, \gamma$.
for $i = 1$ to $T$ do
    Receive: $\left(a^i, p^i, n^i\right)$
    $f_i^{(\alpha)} = d_{a^i, p^i} - d_{a^i, n^i}$ based on detection-to-detection affinity model
    $f_i^{(\beta)} = d_{a^i, p^i} - d_{a^i, n^i}$ based on sequence-to-detection affinity model
    $f_i^{(\gamma)} = d_{a^i, p^i} - d_{a^i, n^i}$ based on sequence-to-sequence affinity model
    Update $\alpha \leftarrow \alpha\eta^{\mathbb{I}\left(f_i^{(\alpha)} > 0\right)}$, $\beta \leftarrow \beta\eta^{\mathbb{I}\left(f_i^{(\beta)} > 0\right)}$, $\gamma \leftarrow \gamma\eta^{\mathbb{I}\left(f_i^{(\gamma)} > 0\right)}$
    $s = \alpha + \beta + \gamma$
    $\alpha \leftarrow \alpha/s$, $\beta \leftarrow \beta/s$, $\gamma \leftarrow \gamma/s$
end for
```

### F. Greedy Data Association Algorithm

This paper focuses on developing a robust affinity model, and we just use the simple greedy data association algorithm to conduct data association. This algorithm works as follows: First, a similarity matrix $S$ for each pair $(tr, d)$ of tracker $tr$ and detection $d$ is computed. Then, the pair $(tr^*, d^*)$ with the maximum similarity score is iteratively selected, and the rows and columns belonging to tracker $tr^*$ and detection $d^*$ in $S$ are deleted. This process is repeated until no further valid pair is available. Finally, only the associated detections with a similarity score above a threshold are used, ensuring that an associated detection actually is a good match to a tracker.

### G. Kalman Filter Status Update

For each tracker, we maintain a Kalman filter. The state space for these Kalman Filters is 8-dimensional vectors $[x, y, a, h, vx, vy, va, vh]^{\mathrm{T}}$ (These elements mean center position, aspect ratio, height, and their respective velocities of a bounding box). Each time we initialize a new tracker using detection $d_t = [d_t(x), d_t(y), d_t(w), d_t(h), d_t(img)]$, we initialize the state of its corresponding Kalman Filter as $\left[d_t(x) + \frac{d_t(w)}{2}, d_t(y) + \frac{d_t(h)}{2}, \frac{d_t(w)}{d_t(h)}, d_t(h), 0, 0, 0, 0\right]$. For a matched tracker at frame $t+k$, we update the state of its Kalman Filter by running a Kalman Filter correction step using its associated detection; while for an unassociated tracker, we estimate the location of the target by running a Kalman Filter prediction step.

## IV. Experiments

### A. Implementation Details

This framework is written in Python with TensorFlow [43] support. For the feature extraction ReID network, we use the learned weights of [39]. With the feature extraction network, we only use the public tracking dataset to train our affinity models. The public data we use is the training video sequences from the MOT16 and MOT17 benchmarks. We set parameters of our method empirically as follows: detection score threshold $\tau_s = 0.3$, number of detections of the shortest tracker $\tau_{n1} = 3$, number of missing frames before being converted to "discareded" state $\tau_{n2} = 5$, number of missing frames before being converted to "disappeared" state $\tau_{n3} = 30$, number of frames being looked forward $K = 6$, pre-defined margin of Focal Triplet Loss $m = 0.2$, focusing parameter $\lambda = 0.2$. The numbers of units of fully connected layers are 1024, 512 and 256 respectively. Rectified Linear Unit (ReLU) activation function is applied for the first two fully connected layers. The number of hidden units of the LSTM cell is 512.

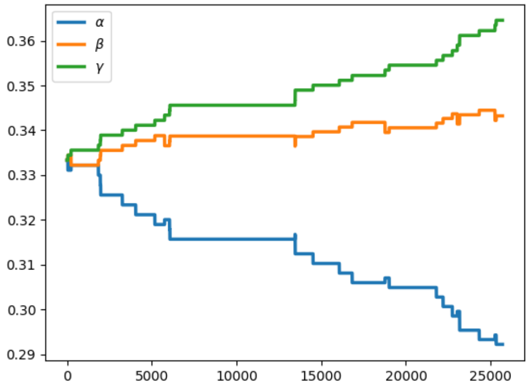

Fig. 7. The curve of the combination weights of three affinity models with the learning process.

The learning result of the Hedge algorithm is shown in Figure 7. We set the initial value of these weights to 1/3. As expected, the weights of the sequence-to-detection and sequence-to-sequence models $\beta, \gamma$ gradually increased, and the weight of the detection-to-detection model $\alpha$ decreased. In the end, the sequence-to-sequence model has the largest weight, the sequence-to-detection model has the second, and the detection-to-detection model has the smallest. According to the results of the Hedge algorithm, we set the combination weights: $\alpha$= 0.292,$\beta$= 0.343,$\gamma$= 0.365.

**Training data generation.** In order to generate training samples, we first assign identities to all detections. The identity of detection is the same as that of a ground truth bounding box which has the largest IOU with the detection. Assume a training video $v$ contains $N$ frames and $M$ objects, and we denote the detection whose identity is $i$ at frame $t$ as $d_t^i$. For the detection-to-detection affinity model, we construct its training triplets as all possible:

$$\begin{aligned} &\left\langle d_t^i, d_{t+gap}^i, d_{t+gap}^j \right\rangle \\ s.t., \quad &i \neq j, 1 \leq gap \leq 10, IOU\left(d_{t+gap}^i, d_{t+gap}^j\right) > 0 \end{aligned} \tag{7}$$

The IOU constraint makes the training samples focus on hard examples, and greatly reduces the number of possible triplets. For the sequence-to-detection affinity model, we construct its training triplets as all possible:

$$\begin{aligned} &\left\langle \left\{ d_{t_K}^i, d_{t_{K-1}}^i \cdots, d_{t_2}^i, d_{t_1}^i \right\}, d_{t_1+gap}^i, d_{t_1+gap}^j \right\rangle \\ s.t., \quad &t_K < t_{K-1} < \cdots < t_2 < t_1, t_k - t_{k+1} \leq \tau_{n3} \\ &i \neq j, 1 \leq gap \leq 10, IOU\left(d_{t_1+gap}^i, d_{t_1+gap}^j\right) > 0 \end{aligned} \tag{8}$$

where $\tau_{n3}$ is defined in Figure 2, which is the threshold to change a "tracked" tracker into a "disappeared" one. For the sequence-to-sequence affinity model, we construct its training triplets as all possible:

$$\begin{aligned} &\left\langle \begin{array}{l} \left\{ d_{t_K}^i, d_{t_{K-1}}^i \cdots, d_{t_2}^i, d_{t_1}^i \right\}, \\ \left\{ d_{t_1+gap}^i, \cdots, d_{t_1+gap+K-1}^i \right\}, \\ \left\{ d_{t_1+gap}^j, \cdots, d_{t_1+gap+K-1}^j \right\} \end{array} \right\rangle . \\ s.t., \quad &t_K < t_{K-1} < \cdots < t_2 < t_1, t_k - t_{k+1} \leq \tau_{n3} \\ &i \neq j, 1 \leq gap \leq 10, IOU\left(d_{t_1+gap}^i, d_{t_1+gap}^j\right) > 0 \end{aligned} \tag{9}$$

Since we only look forward $K$ frames during the tracking process, we construct the positive sample sequence and the negative sample sequence in a frame window of size $K$. If the number of detections of these two sequences is less than $K$, we append zero bounding boxes with zero appearance features to these sequences until the length of all sequences is $K$.

**Training.** When training the similarity models with the new proposed Focal Triplet Loss, we use the Adam [44] optimizer, and set the initial learning rate to $1e^{-4}$, decay by 0.9 every 20 epochs for 100 epochs in total. Mini-batch size is set to 4096. To learn the combination weights by the Hedge algorithm, we split part of the training data for the sequence-to-sequence affinity model as its online learning data.

### B. Evaluation on MOT16 and MOT17

The proposed framework is evaluated on the MOT16 and MOT17 benchmark. The MOT17 dataset comprises 7 training sequences and 7 test sequences in different scenarios, and these sequences are captured with various types of cameras. Each sequence is provided three kinds of public detections: DPM, FRCNN, and SDP. MOT16 shares the same video sequences as MOT17, while sequences in MOT16 are only provided DPM detection input. What's more, MOT17 has fixed the

TABLE I
TRACKING RESULTS ON THE MOT16 TEST DATASET. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED BY RED AND BLUE, RESPECTIVELY.

| Tracker | Type | MOTA↑ | IDF1↑ | MOTP↑ | MT↑ | ML↓ | FP↓ | FN↓ | IDS↓ |
|---|---|---|---|---|---|---|---|---|---|
| GMMCP [45] | offline | 38.1 | 35.5 | 75.8 | 8.6 | 50.9 | 6607 | 105315 | 937 |
| MHT_bLSTM6 [10] | offline | 42.1 | 47.8 | 75.9 | 14.9 | 44.4 | 11637 | 93172 | 753 |
| QuadMOT16 [16] | offline | 44.1 | 38.3 | 76.4 | 14.6 | 44.9 | 6388 | 94775 | 745 |
| EDMT [46] | offline | 45.3 | 47.9 | 75.9 | 17.0 | 39.9 | 11122 | 87890 | 639 |
| MHT_DAM [47] | offline | 45.8 | 46.1 | 76.3 | 16.2 | 43.2 | 6412 | 91758 | 590 |
| FWT [48] | offline | 47.8 | 44.3 | 75.5 | 19.1 | 38.2 | 8886 | 85487 | 852 |
| PHD_GSDL16 [9] | online | 41.0 | 43.1 | 75.9 | 11.3 | 41.5 | 6498 | 99257 | 1810 |
| DMAN [11] | online | 46.1 | 54.8 | 73.8 | 17.4 | 42.7 | 7909 | 89874 | 532 |
| MOTDT [39] | online | 47.6 | 50.9 | 74.8 | 15.2 | 38.3 | 9253 | 85431 | 792 |
| LSST16O [29] | online | 49.2 | 56.5 | 74.0 | 13.4 | 41.4 | 7187 | 84875 | 606 |
| LINF1_16 [13] | near-online | 41.0 | 45.7 | 74.8 | 11.6 | 51.3 | 7896 | 99224 | 430 |
| NOMT_16 [12] | near-online | 46.4 | 53.3 | 76.6 | 18.3 | 41.4 | 9753 | 87565 | 359 |
| ours | near-online | 49.8 | 44.6 | 77.0 | 15.0 | 40.6 | 2835 | 87813 | 868 |

TABLE II
TRACKING RESULTS ON THE MOT17 TEST DATASET. THE BEST AND SECOND BEST RESULTS ARE HIGHLIGHTED BY RED AND BLUE, RESPECTIVELY.

| Tracker | Type | MOTA↑ | IDF1↑ | MOTP↑ | MT↑ | ML↓ | FP↓ | FN↓ | IDS↓ |
|---|---|---|---|---|---|---|---|---|---|
| IOU17 [49] | offline | 45.5 | 39.4 | 76.9 | 15.7 | 40.5 | 19993 | 281643 | 5988 |
| MHT_bLSTM [10] | offline | 47.5 | 51.9 | 77.5 | 18.2 | 41.7 | 25981 | 268042 | 2069 |
| EDMT17 [46] | offline | 50.0 | 51.3 | 77.3 | 21.6 | 36.3 | 32279 | 247297 | 2264 |
| MHT_DAM [47] | offline | 50.7 | 47.2 | 77.5 | 20.8 | 36.9 | 22875 | 252889 | 2314 |
| jCC [50] | offline | 51.2 | 54.5 | 75.9 | 20.9 | 37.0 | 25937 | 247822 | 1802 |
| FWT_17 [48] | offline | 51.3 | 47.6 | 77.0 | 21.4 | 35.3 | 24101 | 247921 | 2648 |
| LSST17 [29] | offline | 54.7 | 62.3 | 75.9 | 20.4 | 40.1 | 26091 | 228434 | 1243 |
| GMPHD_KCF [51] | online | 39.6 | 36.6 | 74.5 | 8.8 | 43.3 | 50903 | 284228 | 5811 |
| EAMTT [52] | online | 42.6 | 41.8 | 76.0 | 12.7 | 42.7 | 30711 | 288474 | 4488 |
| PHD_GSDL17 [9] | online | 48.0 | 49.6 | 77.2 | 17.1 | 35.6 | 23199 | 265954 | 3998 |
| DMAN [11] | online | 48.2 | 55.7 | 75.7 | 19.3 | 38.3 | 26218 | 263608 | 2194 |
| HAM_SADF17 [53] | online | 48.3 | 51.1 | 77.2 | 17.1 | 41.7 | 20967 | 269038 | 1871 |
| MOTDT17 [39] | online | 50.9 | 52.7 | 76.6 | 17.5 | 35.7 | 24069 | 250768 | 2474 |
| LSST17O [29] | online | 52.7 | 57.9 | 76.2 | 20.4 | 40.1 | 22512 | 241936 | 2167 |
| ours | near-online | 52.7 | 49.4 | 77.0 | 17.7 | 38.2 | 10819 | 253890 | 2396 |

ground truth of MOT16 and make them more accurate. These dataset provide a wide range of challenges including occlusion, crowded scenarios, and moving backgrounds. Many related works have reported their results on this dataset, allowing the straightforward comparison of our approach with other state-of-the-art methods.

For quantitative evaluation, we use the widely adopted CLEAR MOT metrics [54] and trajectory-based metrics (TB-M) [55]. MOTA evaluates accuracy in the presence of false positives (FP), false negatives (FN), and identity switches (IDS). MOTP evaluates the intersecting area of the tracking output and the ground truth. IDF1 is the ratio of correctly identified detections over the average number of ground-truth and computed detections, and it indicates the average maximum consistent tracking rate. TBM is used to estimate the completeness of each trajectory, and it is an important supplemental metric for better evaluation of multi-pedestrian tracking methods. Specifically, MT evaluates the mostly tracked trajectories that are successfully tracked at least 80%. ML evaluates the mostly lost trajectories that are successfully tracked at most 20%. IDS counts the total number of identity switches. Frag counts the total number of times a trajectory in the ground truth is interrupted during tracking. Hz indicates the processing speed (in frames per second). Among these metrics, MOTA and IDF1 are usually considered the most important.

We compare our method with state-of-the-art methods, and show the results of MOT16 and MOT17 in Table I and Table II respectively. As evident from Table I and Table II, our tracker performs competitively with other trackers on both the MOT16 and MOT17 challenges in terms of MOTA and IDF1. Our tracker achieves the best MOTA on the MOT16 benchmark, and ranks the second place on the MOT17 benchmark in terms of MOTA, where the best tracker uses an offline method. In terms of IDF1, there is an area of improvement for our tracker.

Our hypothesis is that the proposed Kalman Filter is not accurate enough to predict missing detections of an external detector, and the proposed false alarm model may wrongly remove some true positives, thus our tracker has the smallest number of False Positives, but has a relative large number of False Negatives, as shown in both Table I and Table II, and this results in small ratio of correctly identified detections over the average number of ground-truth. We leave the more accurate motion model and false alarm removal model as our future work.

We show in Figure 8 the qualitative results of our tracker on the MOT17 challenge using SDP detections, three representative images per sequence. Consistency of the estimated trajectories is indicated by bounding boxes of the same color and the same ID number over time. As can be seen from this figure, our tracker yields visually plausible results even on challenging scenarios with many pedestrians or occlusions.

### *C. Ablation Study*

To inspect how different components influence the tracking quality, we conducted an ablation study on an evaluation dataset, which contains the following sequences: "MOT17-02-SDP", "MOT17-05-SDP", "MOT17-09-SDP" and "MOT17-10-SDP". For this section, the training dataset contains the following sequences: "MOT17-04-SDP", "MOT17-11-SDP" and "MOT17-13-SDP".

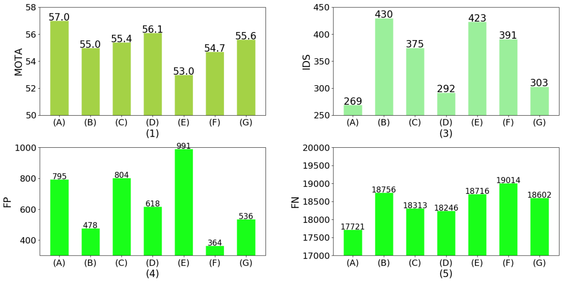

Fig. 9. Analysis of the impact of different components in terms of MOTA (1), IDS (2), FP (3), and FN (4). (A) Full model with the Focal Triplet Loss. (B) Only detection-to-detection similarity model. (C) Only sequence-to-detection similarity model. (D) Only sequence-to-sequence similarity model. (E) Without the false alarm model. (F) Without Kalman Filter. (G) Full model with triplet loss.

Figure 9 shows the impact of different components in terms of MOTA, IDS, FP, and FN respectively. The full model, (A) and (G) in Figure 9, includes all the three similarity models, the false alarm model, and Kalman Filter. The similarity models of (B)-(F) are trained using the Focal Triplet Loss. We will discuss their performance following.

**Impact of the similarity models.** The sequence-to-sequence similarity model utilizes more information than the sequence-to-detection similarity model, which, however, uses more information than the detection-to-detection similarity model. As a consequence, the robustness decreases gradually from the sequence-to-sequence similarity model to the detection-to-detection similarity model. The change of IDS in Figure 9 (2) explicitly depicts this trend. However, these three similarity models are not completely redundant. By combining all the three similarity models, we achieved the highest MOTA as shown in Figure 9 (1).

TABLE III
RANK1, RANK5, RANK10, RANK20 RECOGNITION RATE (IN %) AND MAP OF DIFFERENT SIMILARITY MODELS ON AN EXTRACTED DATASET FROM MOT16 TRAINING SETS.

| Similarity model | Rank1 | Rank5 | Rank10 | Rank20 | mAP |
|---|---|---|---|---|---|
| detection-to-detection | 81.5 | 87.5 | 89.5 | 90.3 | 68.6 |
| sequence-to-detection | 89.8 | 94.3 | 95.5 | 96.0 | 69.9 |
| sequence-to-sequence | **96.9** | **97.7** | **98.3** | **98.9** | **72.4** |

To validate the assertion that the robustness increases gradually from the detection-to-detection similarity model to the sequence-to-sequence similarity model, we also conducted person re-identification experiments using the MOT16 dataset. For each pedestrian trajectory in the MOT16 training sets, we extracted one query image and at most 30 gallery images, and in total obtained 352 query images and 10121 gallery images as a result. Person re-identification can be evaluated based on the Cumulative Matching Characteristics by treating person re-identification as a ranking problem or based on the mean average precision (mAP) by treating it as a retrieval problem. Here we report both rank$m$ and mAP. The results are shown in Table III. It's obvious that all the studied performance metrics improved when comparing the three similarity models, which confirmed our assertion that the robustness increases correspondingly.

**Impact of the false alarm model.** It can be seen from the figure that removing the false alarm model drop the MOTA significantly, meanwhile, it causes the worst FP when compared with the full model, which together demonstrate the effectiveness of the proposed false alarm model. What's more, reducing the number of false positives can also reduce the possibility of associating trackers with false alarms, thus reducing IDS and FN.

**Impact of Kalman Filters.** After we removed the Kalman Filters from the full model, FN increased as shown in Figure 9 (4). This demonstrates that these Kalman Filters are capable of predicting bounding boxes for missed detection responses. However, these Kalman Filters may wrongly predict bounding boxes, since incorporating Kalman Filters also increased FP as shown in Figure 9 (3).

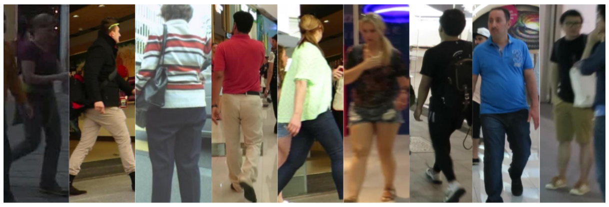

Fig. 10. Predicted bounding boxes by Kalman Filters.

Figure 10 shows some predicted bounding boxes predicted by Kalman Filters. It can be seen that the predicted bounding boxes are highly confident "detection responses". Thus the

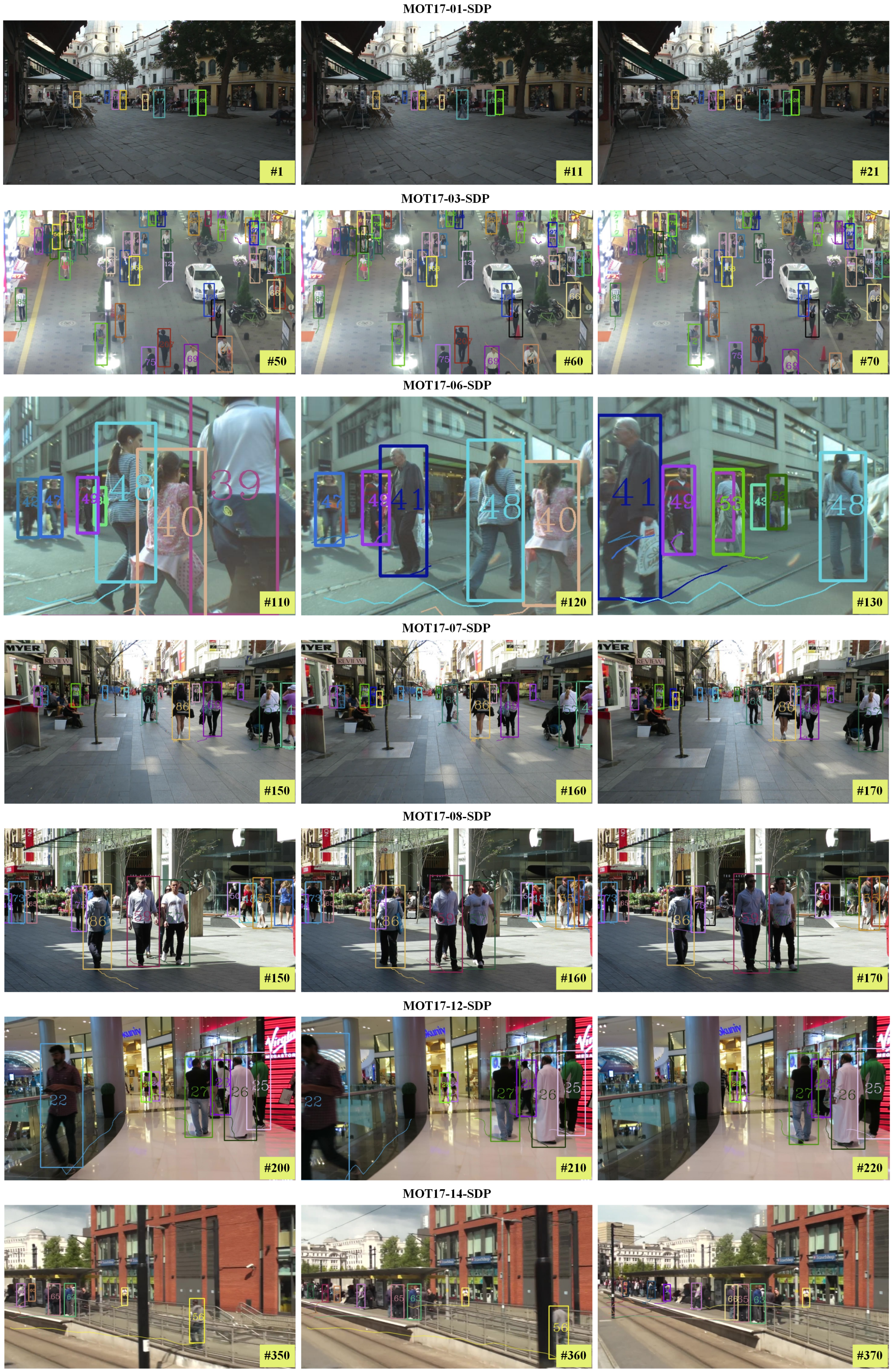

Fig. 8. Qualitative tracking results on different test sequences of the MOT17 dataset, featuring complex scenarios including many pedestrians, luminance changes and low contrasts among many other challenges.

number of false negatives can be effectively reduced by introducing Kalman Filters.

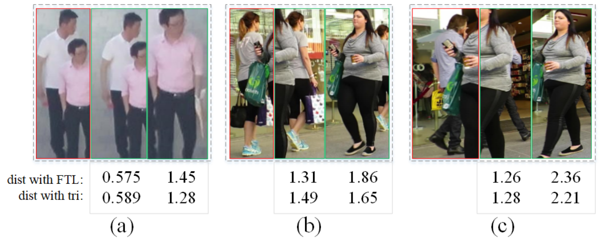

Fig. 11. Hard samples for data association where two detection responses compete for one tracker. The red bounding box of each sample indicates the newest detection response of a tracker, and the green bounding boxes indicate two detection responses to be associated at the current frame.

**Impact of Focal Triplet Loss.** The full model with the Focal Triplet Loss outperforms the full model with the original triplet loss in terms of all considered metrics, demonstrating the superiority of our proposed Focal Triplet Loss.

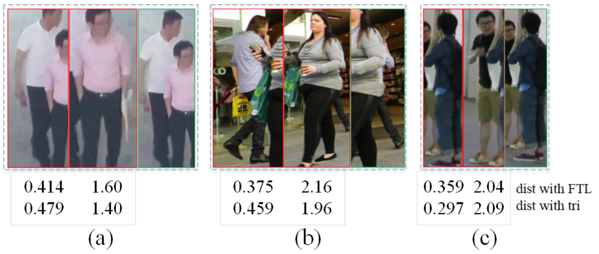

Fig. 12. Hard samples for data association where two trackers compete for one detection response. The red bounding boxes of each sample indicate the newest detection response of two trackers, and the green bounding box indicates one detection response to be associated at the current frame.

To investigate the reason for the superiority of Focal Triplet Loss over triplet loss, we show some hard examples for data association in Figure 11 and Figure 12. These examples are challenging since two detection responses compete for one tracker (Figure 11) or two trackers compete for one detection response (Figure 12). For all these samples, the distance between a tracker and a detection response coming from the same trajectory is smaller by similarity models trained with focal triplet loss (dist with FTL) than that by similarity models with triplet loss (dist with tri), and the distance between a tracker and a detection response coming from different trajectories is larger by similarity models trained with focal triplet loss than that by similarity models with triplet loss. Thus it is easier for data association using similarity models trained with focal triplet loss than using those trained with triplet loss.

### D. Time Analysis

In table IV, we show the running time of our tracker on different test sets of detections given the detections and the extracted ReID features. All experiments were conducted on a server with a Tesla V100 GPU. Note that the running time varies between these sets of detections. This is mainly due to the fact that different detections feature different numbers of detection bounding boxes, which leads to the varying time for both the computation of the similarity matrix and the process of greedy data association. The quality of detection bounding boxes also affects the running time. For example, the number of original DPM detection bounding boxes is much larger than that of original SDP detection. However, many DPM detection bounding boxes are false alarms, and these false alarms are removed by our false alarm model. The valid number of DPM detection bounding boxes is smaller than the valid number of SDP detection, and thus processing speed of DPM detection (15.25 fps) is larger than that of SDP detection (11.65 fps).

TABLE IV
RUNNING TIME AND SPEED OF OUR TRACKER ON DIFFERENT TEST SETS OF DETECTIONS.

| Detections | Frames | Boxes | Density | Running time(s) | Speed(fps) |
|---|---|---|---|---|---|
| DPM | 5919 | 135376 | 22.87 | 388.1 | 15.25 |
| FRCNN | 5919 | 110141 | 18.61 | 397.7 | 14.88 |
| SDP | 5919 | 128653 | 21.74 | 508.2 | 11.65 |
| total | 17757 | 374170 | 21.07 | 1264 | 14.05 |

## V. Conclusion

In this paper, we have presented a robust affinity model for multi-pedestrian tracking in the paradigm of tracking by detection. Based on the appearance consistent assumption, we fully utilize information from the current, past and future, and form three different types of metrics respectively to measure the appearance similarity. As these three metrics complement to each other, they provide comprehensive evidence to the data association of multi-pedestrian tracking.

To make our designed metrics have better performance to distinguish the appearance-similar pedestrians when they come close, which frequently happens but is the most challenging in multi-pedestrian tracking, we extend the triplet loss into a Focal Triplet Loss by automatically emphasize the hard examples.

The experiments conducted on the MOT16 and MOT17 Challenges show that our proposed affinity model achieves a very competitive performance even using a simple data association strategy.

## References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 91–99.

[2] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv:1804.02767*, 2018.

[3] A. A. Mekonnen and F. Lerasle, "Comparative evaluations of selected tracking-by-detection approaches," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 996–1010, 2018.

[4] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[5] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1201–1208.

[6] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Subgraph decomposition for multi-target tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5033–5041.

[7] H. Sheng, Y. Zhang, J. Chen, Z. Xiong, and J. Zhang, "Heterogeneous association graph fusion for target association in multiple object tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 11, pp. 3269–3280, 2018.

[8] 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*, vol. 29, no. 12, pp. 3660–3672, 2018.

[9] Z. Fu, P. Feng, F. Angelini, J. Chambers, and S. M. Naqvi, "Particle phd filter based multiple human tracking using online group-structured dictionary learning," *IEEE Access*, vol. 6, pp. 14 764–14 778, 2018.

[10] 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*, 2018, pp. 200–215.

[11] 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*, 2018, pp. 366–382.

[12] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3029–3037.

[13] 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 *Proceedings of the European Conference on Computer Vision*, 2016, pp. 774–790.

[14] 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 Workshop*, 2016, pp. 33–40.

[15] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. Luk Chan, and G. Wang, "Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, 2016, pp. 1–8.

[16] 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*, 2017, pp. 5620–5629.

[17] 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*, 2017, pp. 300–311.

[18] B. Wang, G. Wang, K. L. Chan, and L. Wang, "Tracklet association by online target-specific metric learning and coherent dynamics estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 3, pp. 589–602, 2016.

[19] C. Ma, C. Yang, F. Yang, Y. Zhuang, Z. Zhang, H. Jia, and X. Xie, "Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for multiple object tracking," in *IEEE International Conference on Multimedia and Expo*, 2018, pp. 1–6.

[20] G. Wang, Y. Wang, H. Zhang, R. Gu, and J.-N. Hwang, "Exploit the connectivity: Multi-object tracking with trackletnet," in *Proceedings of the 27th ACM International Conference on Multimedia*. ACM, 2019, pp. 482–490.

[21] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[22] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *Journal of Machine Learning Research*, vol. 10, no. Feb, pp. 207–244, 2009.

[23] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks." in *Proceedings of the British Machine Vision Conference*, 2016, pp. 119.1–119.11.

[24] 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*, 2017, pp. 2980–2988.

[25] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv:1603.00831*, 2016.

[26] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Online multiperson tracking-by-detection from a single, uncalibrated camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1820–1833, 2011.

[27] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 539–546.

[28] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[29] W. Feng, Z. Hu, W. Wu, J. Yan, and W. Ouyang, "Multi-object tracking with multiple cues and switcher-aware classification," *arXiv:1901.06129*, 2019.

[30] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017, pp. 4225–4232.

[31] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[32] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[33] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Deep metric learning for person re-identification," in *International Conference on Pattern Recognition*, 2014, pp. 34–39.

[34] S. Zhang, Q. Zhang, X. Wei, Y. Zhang, and Y. Xia, "Person re-identification with triplet focal loss," *IEEE Access*, vol. 6, pp. 78 092–78 099, 2018.

[35] S. Sun, N. Akhtar, H. Song, A. S. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[36] P. Chu and H. Ling, "Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking," *arXiv:1904.04989*, 2019.

[37] C. Ma, Y. Li, F. Yang, Z. Zhang, Y. Zhuang, H. Jia, and X. Xie, "Deep association: End-to-end graph-based learning for multiple object tracking with conv-graph neural network," in *Proceedings of the 2019 on International Conference on Multimedia Retrieval*. ACM, 2019, pp. 253–261.

[38] 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*, 2015, pp. 4705–4713.

[39] C. Long, A. Haizhou, Z. Zijie, and S. Chong, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *IEEE International Conference on Multimedia and Expo*, 2018, pp. 1–6.

[40] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 379–387.

[41] L. Zhao, X. Li, Y. Zhuang, and J. Wang, "Deeply-learned part-aligned representations for person re-identification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3219–3228.

[42] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[43] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation*, 2016, pp. 265–283.

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

[45] A. Dehghan, S. Modiri Assari, and M. Shah, "Gmmcp tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 4091–4099.

[46] 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*. IEEE, 2017, pp. 18–27.

[47] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2015, pp. 4696–4704.

[48] R. Henschel, L. Leal-Taixe, 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*. IEEE, 2018, pp. 1428–1437.

[49] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2017, pp. 1–6.

[50] 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*, vol. 42, no. 1, pp. 140–153, 2018.

[51] 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 *IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, 2017, pp. 1–5.

[52] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, "Online multi-target tracking with strong and weak detections," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 84–99.

[53] Y.-c. Yoon, A. Boragule, Y.-m. Song, K. Yoon, and M. Jeon, "Online multi-object tracking with historical appearance matching and scene adaptive detection filtering," in *IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2018, pp. 1–6.

[54] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *Journal on Image and Video Processing*, vol. 2008, p. 1, 2008.

[55] B. Yang and R. Nevatia, "Multi-target tracking by online learning a crf model of appearance and motion patterns," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 203–217, 2014.

**Weijiang Feng** is currently a Ph.D. student with the College of Computer, National University of Defense Technology. He received the M.S. degree in computer science from the National University of Defense Technology in 2016. His research interests include multi-object tracking, computer vision, and reinforcement learning.

**Long Lan** is currently a lecturer with College of Computer, National University of Defense Technology. He received the Ph.D. degree in computer science from National University of Defense Technology 2017. He was a visiting Ph.D. student in University of Technology, Sydney from 2015 to 2017. His research interests include multi-object tracking, computer vision and discrete optimization.

**Yong Luo** received the B.E. degree from the Northwestern Polytechnical University, and the D.Sc. degree from the Peking University. He is currently a Research Fellow with the School of Computer Science and Engineering, Nanyang Technological University. His research interests are primarily on machine learning and data mining with applications to visual information understanding and analysis.

**Yue Yu** is an assistance professor of National University of Defense Technology. He received his Ph.D. degree in Computer Science from National University of Defense Technology in 2016, and won Outstanding Ph.D. Thesis Award from Hunan Province. His current research interests include software engineering and Artificial Intelligence.

**Xiang Zhang** received the M.S., and Ph.D. degrees from the National University of Defense Technology in 2010 and 2015, respectively. He is currently a research assistant with the Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer, National University of Defense Technology. His current research interests include computer vision and machine learning.

**Zhigang Luo** received the B.S., M.S., and Ph.D. degrees from the National University of Defense Technology in 1981, 1993, and 2000, respectively. He is currently a Professor with the College of Computer, National University of Defense Technology. His current research interests include machine learning, computer vision, and bioinformatics.