Exploit the Connectivity: Multi-Object Tracking with TrackletNet

Gaoang Wang*
gaoang@uw.edu
University of Washington
Seattle, Washington

Yizhou Wang
ywang26@uw.edu
University of Washington
Seattle, Washington

Haotian Zhang
haotiz@uw.edu
University of Washington
Seattle, Washington

Renshu Gu
renshugu@uw.edu
University of Washington
Seattle, Washington

Jenq-Neng Hwang
hwang@uw.edu
University of Washington
Seattle, Washington

ABSTRACT

Multi-object tracking (MOT) is an important topic and critical task related to both static and moving camera applications, such as traffic flow analysis, autonomous driving and robotic vision. However, due to unreliable detection, occlusion and fast camera motion, tracked targets can be easily lost, which makes MOT very challenging. Most recent works exploit spatial and temporal information for MOT, but how to combine appearance and temporal features is still not well addressed. In this paper, we propose an innovative and effective tracking method called *TrackletNet Tracker (TNT)* that combines temporal and appearance information together as a unified framework. First, we define a graph model which treats each tracklet as a vertex. The tracklets are generated by associating detection results frame by frame with the help of the appearance similarity and the spatial consistency. To compensate camera movement, epipolar constraints are taken into consideration in the association. Then, for every pair of two tracklets, the similarity, called the *connectivity* in the paper, is measured by our designed multi-scale TrackletNet. Afterwards, the tracklets are clustered into groups and each group represents a unique object ID. Our proposed TNT has the ability to handle most of the challenges in MOT, and achieves promising results on MOT16 and MOT17 benchmark datasets compared with other state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies → Tracking**; **Neural networks**; *Epipolar geometry*.

KEYWORDS

multi-object tracking; TrackletNet; epipolar geometry; connectivity; tracklet

ACM Reference Format:
Gaoang Wang, Yizhou Wang, Haotian Zhang, Renshu Gu, and Jenq-Neng Hwang. 2019. Exploit the Connectivity: Multi-Object Tracking with TrackletNet. In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3350853

1 INTRODUCTION

Multi-object tracking (MOT) is an important topic in computer vision and machine learning field. This technique is critically needed in many tasks, such as traffic flow analysis from static cameras, human behavior prediction and autonomous driving assistance [7, 10, 37, 38, 40]. However, due to the noisy visual object detection and occlusion in the crowds, tracking multiple objects over long time is very challenging. To address such problems, many methods follow the tracking-by-detection framework, i.e., tracking is applied as an association approach given the detection results. Built upon the tracking-by-detection framework, multiple cues can be combined together into the tracking scheme. 1) Designing robust appearance features of detected object [29, 36, 42, 46] is the most widely used cue in the tracking. With a well-embedded appearance, features should be similar if they are from the same object, while they can be very different if they are from distinct objects. 2) Exploiting temporal relation for locations among frames in a trajectory [23] can serve as another important cue. With slow motion and high frame rate of cameras, we can assume that the trajectories of objects follow some smoothing functions in the time domain. 3) Interaction cue among different target objects considers the relationship among neighboring targets in a crowd or a group [30]. As a result, we should take into account all these cues in the tracking system.

In this paper, the proposed TrackletNet Tracker (TNT) takes advantages of the above useful cues together into a unified framework based on an undirected graph model [24]. Each vertex in our graph model represents one tracklet and the edge between two vertices measures the similarity between the two connected tracklets. Here, the tracklet is defined as a small piece of consecutive detections of an object by detection association, which will be discussed in detail in Section 4.1. Due to the unreliable detection and occlusion among objects, the entire trajectory of an object may be divided into several distinct tracklets. Given the graph representation, tracking can be regarded as a clustering approach that can merge the tracklets into big clusters.

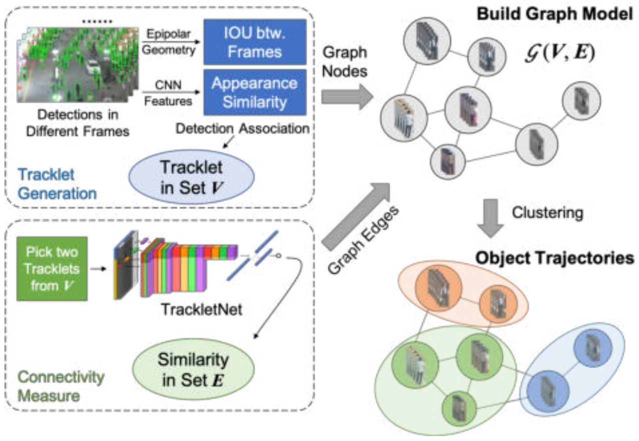

Figure 1: Our TNT framework for multi-object tracking. Given the detections in different frames, association is conducted to generate tracklets as the Vertex Set V. After that, every pair of two neighboring tracklets are sent into the TrackletNet to measure the connectivity, which forms the similarity on the Edge Set E. Finally, the tracklets with the same ID are grouped into one cluster using the graph partition approach on the defined graph $\mathcal{G}(V, E)$.

To generate the tracklets, i.e., the vertices of the graph, we associate detections among consecutive frames based on the spatial consistency, i.e., the intersection-over-union (IOU) between bounding boxes and the similarity of appearance features. However, the IOU criterion becomes unreliable when the camera is moving or revolving because the position of detection may shift a lot on the 2D image plane. In such a situation, epipolar geometry is adopted to compensate camera movement and predict the position of bounding boxes in the next frame. To estimate the connectivity on the edge of the graph between two vertices, multi-scale TrackletNet based on convolutional neural networks (CNN) is designed for measuring the similarity and continuity of each pair of two input tracklets. Specifically, we propose the following contributions:

1) To the best of our knowledge, this is the first work to adopt epipolar geometry in the tracklet generation to compensate camera movement.

2) A CNN architecture, called multi-scale TrackletNet, is designed to measure the connectivity between two neighboring tracklets. This network combines temporal consistency and appearance information into a unified system.

3) Our model outperforms many state-of-the-art methods in MOT for both MOT16 and MOT17 benchmarks, and it can be also easily applied to other different scenarios and tasks.

The outline of the paper is as follows: In Section 2, we review some related works of state-of-the-art methods in MOT. Section 3 describes the proposed tracklet-based graph modeling. Section 4 presents the multi-scale TrackletNet for measuring the connectivity between two tracklets. The simulations and discussions are provided in Section 5, followed by the conclusions and future works in Section 6.

2 RELATED WORK

Graph Model based Tracking. Most of the recent multi-object tracking approaches are based on tracking-by-detection schemes [6, 45]. Given detection results, tracking is treated as a detection association task. Many tracking methods are based on graph models [2, 14, 18, 24, 34–36, 38, 41] and solve the tracking problem by minimizing the total cost. Generally, there are two categories of graph models. One treats the individual detections as the vertices [18, 24, 35, 36], while the other using tracklets as vertices [2, 34, 38, 41]. For detection-based graph models, there are two major disadvantages. First, one of the important assumptions in graph models is the conditional independence of the vertices. However, detections are not conditional independent from frame to frame if we want to track an object in a long run. Hence, the temporal information is not well utilized. Second, detection-based graph usually comes with a very high-dimensional affinity matrix, which is very time consuming to find a good solution in the optimization. On the contrary, tracklet-based graph models can better utilize the information from a short trajectory to measure the relationship between vertices, but the mis-association should be carefully handled in the tracklet generation step.

Tracking by RNNs. Besides graph models, recurrent neural networks (RNNs)-based tracking also plays an important role in recent years [17, 20, 21, 23, 30]. For example, [23] first time proposes an end-to-end learning approach which uses RNN to model the target motion. However, drawbacks are also obvious. It can be easily affected by the camera motion and does not utilize appearance information in the association. Similarly, [20] minimizes the regression and association error in a unified framework by using long short-term memory (LSTM) blocks. One advantage of RNN-based tracking is the ability of online prediction. However, along with the propagation of RNN blocks, the relation between two far-away detections becomes very weak, especially for high-dimensional appearance features [17]. As a result, the drift error will be easily accumulated during the long-time occlusion. The performance of RNN-based methods degrades in the long run and sometimes can be easily affected by unreliable detections.

Tracking by Feature Fusion. Features are very important in the tracking-by-detection framework. There are two types of features that are used in common, i.e., appearance features and temporal features. For appearance features, many works adopt CNN-based features and treat tracking as the re-identification (Re-ID) task [29, 36, 46, 47]. For example, [29] proposes an adaptive weighted triplet loss for training and a novel technique for hard-identity mining. [46] adopts the re-ranking technique [47] in calculating the feature similarity. Besides CNN-based features, histogram-based features, like color histograms, histogram of oriented gradients (HOG), and local binary patterns (LBP), are still powerful if no labeled training data are available [38]. For temporal features, the location, size, and motion of bounding boxes are commonly used. Given the appearance features and temporal features, the tracker can fuse them together, like [24, 38, 46]. However, it is still empirical and difficult to determine the weights of different types of features.

End-to-End Tracking. Another category of tracking is based on end-to-end frameworks [3, 11, 12], where we input raw video sequences and output object trajectories. In other words, the detection and tracking are trained jointly in a single-stage network. One major advantage of this framework is that the errors will not be accumulated from detection to tracking. The temporal information across frames can help improve the detection performance, while reliable detections can also further improve the tracking performance. However, such a framework requires a lot of training data with a large diversity of different scenarios. Without enough training data, over-fitting becomes a severe problem. Unlike detection based training, tracking annotations for video sequences are usually expensive to be obtained, which becomes the major limitation of the end-to-end tracking framework.

3 TRACKLET-BASED GRAPH MODEL

Unlike the detection-based graph models, which are computational expensive and not well utilizing temporal information, our proposed TrackletNet Tracker (TNT), as shown in Figure 1, uses tracklets as the vertices in our graph model with edges measuring the similarities between tracklets. From the tracklets, we can infer objects' moving trajectories for a longer time, and we can also measure how the embedded features of the detections change along the time. Moreover, the number of tracklets is much less than the number of detections, which makes the optimization more efficient. In the following section, we will discuss in detail about the designed graph model and formulate the tracking as an optimization problem.

3.1 Graph Definition $\mathcal{G}(V, E)$

Vertex Set. A finite set V in which every element $u \in V$ represents a tracklet across multiple frames, i.e., a set of consecutive detections of the same object along time. For each detection, we define the bounding box with five parameters, i.e., the center of the bounding box (x_t, y_t), the width and height (w_t, h_t), and the frame index t. Besides the bounding box of the detection, we also extract an appearance feature [31] for each detected object at frame t. Note that because of unreliable detections, an entire trajectory of an object may be divided into multiple pieces of tracklets. The tracklet generation is explained in detail in Section 4.1.

Edge Set. A finite set E in which every element $e \in E$ represents an edge between a pair of two neighboring tracklets $u, w \in V$ in the time domain, i.e., $\min_{t_u \in T(u), t_w \in T(w)} |t_u - t_w| \leq \delta_t$, where $T(u)$ is the set of frame indices of the tracklet u. For tracklets that are far away from each other, the edge is not considered between them since not enough information can be utilized for measuring their relationship.

Then, a connectivity measure p_e is defined to represent the similarity of the two tracklets connected by the edge $e \in E$. The edge cost is defined as

$$c_e = \log\left(\frac{p_e}{1 - p_e}\right), \tag{1}$$

which represents the cost of cutting the edge. Moreover, the connectivity is defined to be 0 if two tracklets have overlap in the time domain since they must belong to distinct objects. This is because an object cannot appear in two tracklets at the same time. The connectivity is measured by our designed TrackletNet, which will be introduced in Section 4.2.

3.2 Tracklet Clustering

After the tracklet graph is built, we acquire the object trajectories by clustering the graph into different sub-graphs. The tracklets in each sub-graph can represent the same object. Given a tracklet graph $\mathcal{G}(V, E)$, for every edge $e \in E$, a cost or reward c_e is to be payed based on the predicted clustering labels. Let u and w be arbitrary neighboring vertices connected by the edge e. Let $\pi_e = -1$ if u and w are clustered in the same track and $\pi_e = 1$ if they are clustered in distinct tracks. Then the cost will be payed if the similarity $p_e > 0.5$ and $\pi_e = 1$, or the similarity $p_e < 0.5$ and $\pi_e = -1$. Therefore, the clustering cost on the edge e can be defined as $\pi_e \cdot c_e$. As a result, the objective function can be defined to minimize the total clustering cost on all graph edges as follows,

$$\begin{aligned} \min_{\pi_e \in \{+1,-1\}} \quad & \sum_{e \in E} \pi_e \cdot c_e, \\ \text{subject to} \quad & \max(\pi_e, 0) \leq \sum_{e' \in C \setminus \{e\}} \max(\pi_{e'}, 0), \\ & \forall C \in \textsc{Cycles}(\mathcal{G}), \forall e \in C. \end{aligned} \tag{2}$$

Here, $\textsc{Cycles}(\cdot)$ returns all cycles in a graph.

Note that the costs c_e can be both positive or negative. For tracklets $u, w \in V$ connected by an edge $e = \{u, w\}$, the assignment $\pi_e = -1$ indicates that u and w belong to the same track. Thus, the constraint in the objective function can be understood as follows: If, for any neighboring vertices u and w, there exists a path in $\mathcal{G}$ from u to w along which all edges are labeled -1 (indicating that u and w belong to the same track), then the edge u, w cannot be labeled 1 (which would indicate the opposite), where the max function in the equation is to convert the label from -1 to 0. In fact, the constraints from Equation (2) are generalized transitivity constraints which guarantee that a feasible solution π well-defines a decomposition of the graph $\mathcal{G}$ into tracks.

Based on the objective function, the graph partition is formulated as a clustering problem. However, the minimum cost of graph cut problem defined by Equation (2) is APX-hard [26]. Besides, the number of clusters is unknown in advance. In this work, we adopt a graph clustering method proposed by [38] to minimize the cost with five clustering operations, i.e., assign, merge, split, switch, and break. The advantage of adopting different types of clustering operations is to avoid being stuck at a bad local minimum as much as we can in the optimization.

4 PROPOSED TRACKLETNET TRACKER

4.1 Tracklet Generation with Epipolar Geometry Constraints

As defined in Section 3, a tracklet contains consecutively detected objects with bounding box information and embedded appearance features. To simplify the generation of tracklets, we associate two consecutive detections based on IOU and appearance similarity in adjacent frames with a high association threshold to make the mis-association as small as possible [39, 46].

However, the association accuracy can still be affected by the fast motion of the camera. For example, as shown in Figure 2(a)(b), the target detection in the t-th frame has a large IOU with another

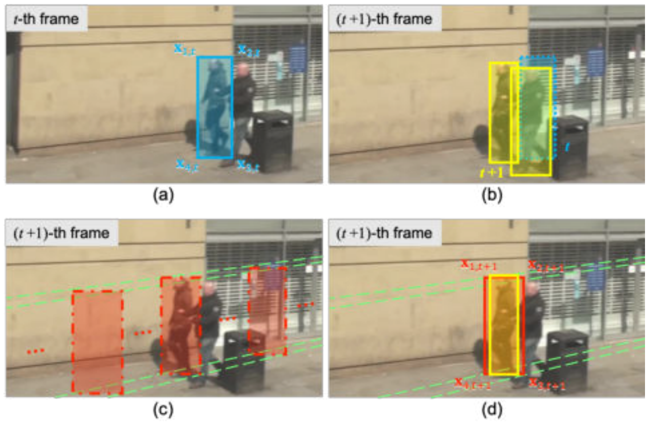

Figure 2: An example of EG-based detection association. (a) The t-th frame with the target detection (blue). (b) The (t+1)-th frame with new detections (yellow). The target detection from the t-th frame (blue dash-box) has a larger IOU with a different candidate detection in the (t+1)-th frame (right yellow box). (c) Some examples of candidate predicted bounding boxes (red dash-boxes) intersected with epipolar lines (green dash-lines) corresponding to the four virtual corners of the bounding box of the t-th frame. (d) The predicted bounding box (red) in the (t+1)-th frame overlapped with the correct detection (yellow).

detection in the $(t+1)$-th frame. As a result, the detection may easily get mis-associated.

As we know, the epipolar geometry (EG) [8], i.e., $\mathbf{x}_t^\top \mathbf{F} \mathbf{x}_{t+1} = 0$ holds for each pair of static points in two frames, where $\mathbf{F}$ is the fundamental matrix, $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$ are the matched points in two frames. Due to the imperfect matching or slow motion of the matched points, $\mathbf{x}_t^\top \mathbf{F} \mathbf{x}_{t+1}$ is usually close to zero but does not exactly equal to zero. Here we propose two assumptions. First, we assume the target is static or has slow motion, then the four virtual corner points $\{\mathbf{x}_{i,t}\}, i \in \{1,2,3,4\}$ of the target detection bounding box in the t-th frame should lie on the corresponding epipolar lines in the $(t+1)$-th frame, i.e., the predicted target bounding box in the $(t+1)$-th frame should intersect with the four epipolar lines as much as possible as shown in Figure 2(c). Second, we also assume the size of the bounding box does not have much change in adjacent frames, then the optimal predicted bounding box can be obtained, which is shown in red in Figure 2(d).

Followed by the above two assumptions, we can predict the target bounding box location in the $(t+1)$-th frame by formulating an optimization problem. Define four corner points of the target bounding box in the t-th frame as $\{\mathbf{x}_{i,t}\}, i \in \{1,2,3,4\}$, like the example shown in Figure 2(a). Similarly, we define $\{\mathbf{x}_{i,t+1}\}, i \in \{1,2,3,4\}$, as the bounding box in the $(t+1)$-th frame. Then we can define the cost function as follows,

$$f(\mathbf{x}_{i,t+1}) = \sum_{i=1}^{4} \|\mathbf{x}_{i,t}^\top \mathbf{F} \mathbf{x}_{i,t+1}\|_2^2 + \|(\mathbf{x}_{3,t+1} - \mathbf{x}_{1,t+1}) - (\mathbf{x}_{3,t} - \mathbf{x}_{1,t})\|_2^2, \tag{3}$$

where the first term guarantees the predicted bounding box should intersect with four corresponding epipolar lines as much as possible, while the second term is the target size constraint in adjacent frames. One example of predicted bounding box, as shown in Figure 2(d), is well aligned with the true target in the $(t+1)$-th frame. Then, in the detection association, IOU is calculated between predicted bounding boxes and detection bounding boxes in the $(t+1)$-th frame.

The optimization of the cost function in Equation (3) can be reformulated into a Least Square problem and solved efficiently. Here, we write the fundamental matrix $\mathbf{F}$ as column vectors, i.e., $[\mathbf{f}_1, \mathbf{f}_2, \mathbf{f}_3]$, then Equation (3) can be reformulated as

$$f = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2^2, \tag{4}$$

where

$$\mathbf{A} = \begin{bmatrix} \mathbf{x}_{1,t}^\top \mathbf{f}_1 & \mathbf{x}_{1,t}^\top \mathbf{f}_2 & 0 & 0 \\ 0 & \mathbf{x}_{2,t}^\top \mathbf{f}_2 & \mathbf{x}_{2,t}^\top \mathbf{f}_1 & 0 \\ 0 & 0 & \mathbf{x}_{3,t}^\top \mathbf{f}_1 & \mathbf{x}_{3,t}^\top \mathbf{f}_2 \\ \mathbf{x}_{4,t}^\top \mathbf{f}_1 & 0 & 0 & \mathbf{x}_{4,t}^\top \mathbf{f}_2 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}, \tag{5}$$

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_{1,t+1}(x) \\ \mathbf{x}_{1,t+1}(y) \\ \mathbf{x}_{3,t+1}(x) \\ \mathbf{x}_{3,t+1}(y) \end{bmatrix}, \mathbf{b} = \begin{bmatrix} -\mathbf{x}_{1,t}^\top \mathbf{f}_3 \\ -\mathbf{x}_{2,t}^\top \mathbf{f}_3 \\ -\mathbf{x}_{3,t}^\top \mathbf{f}_3 \\ -\mathbf{x}_{4,t}^\top \mathbf{f}_3 \\ \mathbf{x}_{3,t}(x) - \mathbf{x}_{1,t}(x) \\ \mathbf{x}_{3,t}(y) - \mathbf{x}_{1,t}(y) \end{bmatrix}. \tag{6}$$

In our experiments, the fundamental matrix $\mathbf{F}$ is efficiently estimated using ORB-SLAM [25].

4.2 Multi-Scale TrackletNet

To measure the connectivity between two tracklets, we aggregate different types of information, including temporal and appearance features via the designed multi-scale TrackletNet. The architecture of the proposed TrackletNet is shown in Figure 3.

For each frame t, a vector consisting of the bounding box parameters, i.e., (x_t, y_t, w_t, h_t), concatenated by an embedded appearance feature extracted from the FaceNet [31], is used to represent an individual detection from a tracklet. Considering two connected tracklets with a shared edge in the graph, we concatenate the embedded appearance feature, with dimension d_{ap}, of each detection from these two tracklets inside a time window with a fixed size T. Then the feature space in the time window of the two tracklets is $(4 + d_{ap}) \times T$. For frames with missing detections between the two target tracklets, we use $(4 + d_{ap})$ dimensional interpolated vectors to fill out the blank space. Besides, zero-padding is used for frames after the second tracklet. To better represent the time occupancy of input tracklets, two binary masks are used as individual channels with $(4+d_{ap}) \times T$ dimension for each input tracklet. For each frame t, if the detection exists, then we set the t-th column of the corresponding binary mask to be all $\mathbf{1}$ vector; otherwise we set $\mathbf{0}$ vector instead. As a result, the size of the input tensor of the TrackletNet is $B \times (4 + d_{ap}) \times T \times 3$, where B is the batch size and 3 indicates the number of channels, one for the embedded feature map and the other two for the binary masks.

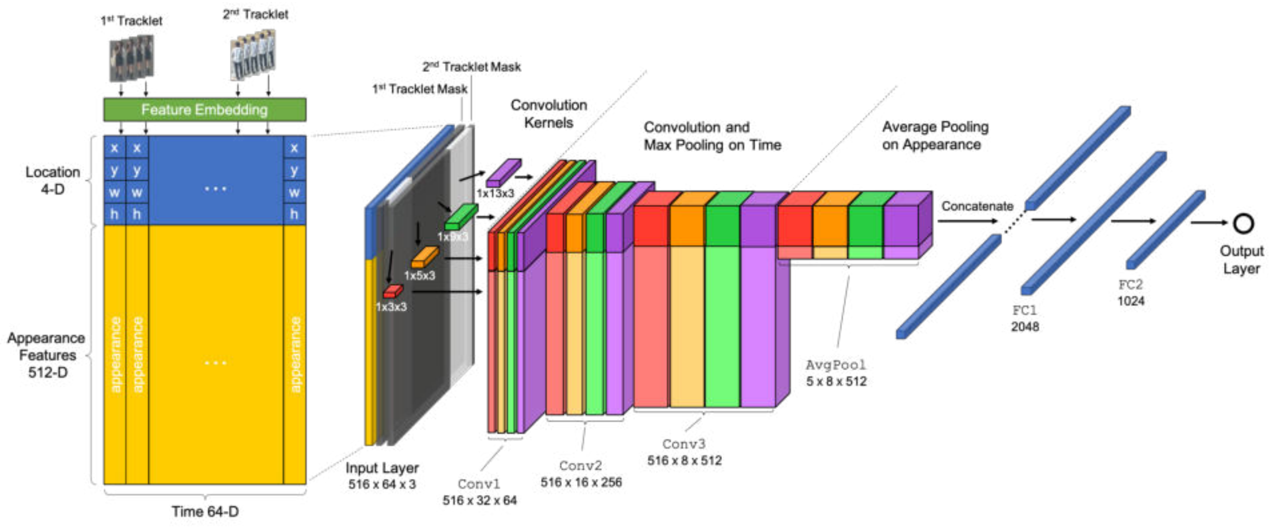

Figure 3: The architecture of the proposed multi-scale TrackletNet.

The TrackletNet contains three convolution layers followed by max pooling `Conv1`, `Conv2`, `Conv3`, one average pooling layer `AvgPool`, and two fully connected layers `FC1`, `FC2`. For each convolution layer, inception-like [33] block is used with four different sizes of kernels, i.e., 1×3, 1×5, 1×9, 1×13, which can extract useful information in different scales. Note that our convolution is only in the time domain, which can measure the continuity for each dimension of the feature. Different sizes of kernels will look for feature changes in different scales. Large kernels have the ability to measure the continuity of two tracklets even if they are far away in the time domain, while small kernels can focus on appearance difference if input tracklets are in small pieces. Each convolution is followed by one max pooling layer which down-samples by 2 in the time domain. After `Conv3`, we take the average pooling on appearance feature dimensions. The average pooling plays a role of weighted majority vote to measure the discontinuity of all appearance dimensions. Then we concatenate all features and use two fully connected layers for the final output. The output is defined as the similarity between the two input tracklets, which ranges from zero to one.

We use the binary cross entropy loss as the objective function in the training, which is

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \log p_i + (1 - y_i)\log(1 - p_i), \tag{7}$$

where y_i is the ground truth label, which indicates whether the two input tracklets are from the same object, p_i is the output probability, and N is the number of training samples in a batch.

There are some important properties of the TrackletNet, which are listed as follows.

1) The TrackletNet focuses on the continuity of the embedded features along the time. Because of the independence among different feature dimensions, no convolution is conducted across the dimensions of the embedded features. In other words, the convolution kernels only capture the dependency along time.

2) Binary masks of the input tensor play a role of the tracklet indicator, telling the temporal occupancy of the tracklets. They help the network learn if the discontinuity of two tracklets is caused by frames without detection or the abrupt changes of the tracklets. These two types of discontinuity should have different activation results.

3) The network integrates object Re-ID, temporal and spatial dependency as one unified framework.

5 EXPERIMENTS

5.1 Dataset

We use MOT16 and MOT17 [22] datasets to train and evaluate our tracking performance. For MOT16 dataset, there are 7 training video sequences and 7 testing video sequences. The benchmark provides deformable part models (DPM) [4] detections for both training and testing sequences. With the public detection provided for all the participants, the tracking performance can be fairly compared in the evaluation. MOT17 has the same video sequences as MOT16 but provides more accurate ground truth in the evaluation. In addition to DPM, Faster-RCNN [27] and scale dependent pooling (SDP) [43] detection results are also provided. The number of trajectories in the training data is 546 and the number of total frames is 5,316.

5.2 Implementation Details

Our proposed multi-scale TrackletNet is purely trained on MOT dataset. The extracted appearance feature has 512 dimensions, i.e., $d_{ap} = 512$. The time window T is set to 64 and the batch size B is set to 32. We use Adam optimizer with a learning rate of 10^{-3} at the beginning. We decrease the learning rate by 10 every 2,000 steps until it reaches 10^{-5}. As mentioned above, the MOT dataset is quite a small dataset for training a complex neural network. However, the architecture of our proposed TNT is carefully designed to avoid over-fitting. In addition, augmentation approaches are used for generating the training samples, i.e., tracklets, as follows.

Bounding Box Randomization. Instead of using the ground truth bounding boxes for training, we randomly disturb the size and location of bounding boxes by a factor α sampled from the normal distribution $\mathcal{N}(0, 0.05^2)$. Since the detection results could be very noisy, this randomization will make sure the data from training and testing are as similar as possible. For each embedded detection before TrackletNet, the four parameters, i.e., (x, y, w, h), are normalized by the size of the frame image to ensure the input of TrackletNet keeps the same scale in different datasets.

Tracklet Generation. We randomly divide the trajectory of each object into small pieces of tracklets as follows. For each frame of a trajectory, we sample a random number from the uniform distribution, if it is smaller than a threshold, then we set this frame as a breaking point. Then we split the entire trajectory based on the breaking points into tracklets.

In the training stage, we randomly generate tracklets with augmentations mentioned above. For each training data, two neighboring tracklets are randomly selected as the input. If they are from the same object, the training label is set to be 1; otherwise, 0 is assigned as the label. To avoid bias, positive and negative pairs are sampled equally.

5.3 Feature Map Visualization

To better understand the effectiveness of our proposed TrackletNet, we also plot two examples of feature maps (a) and (b), as shown in Figure 4. In these two examples, the top figures show the spatial locations of the two input tracklets in the 64-frame time window. Blue and green colors represent two tracklets respectively. The bottom figures show the corresponding feature map along the time domain after the max pooling of `Conv3` with kernel size 5. The horizontal axis represents the time domain which aligns with the figures in the top row, while the vertical axis represents different channels in the feature map. For the example shown in (a), much higher values of the feature map are on the left side since the connection between the two tracklets is on the left part of the time window. As for (b), higher values in the feature map are on the right side of the time window, which also matches the situation of the two input tracklets. From the feature map, we can see that the connection part of the input tracklets has strong activation, which is critical for the connectivity measurement.

5.4 Tracking Performance

Quantitative Results on MOT16 and MOT17 Datasets. We also provide our quantitative results on MOT16 and MOT17 benchmark datasets compared with other state-of-the-art methods, which are shown in Table 1 and Table 2. Note that we use IDF1 [28] and MOTA as major performance metrics to evaluate the reliability of a tracker. As mentioned in [28], there are several weaknesses of MOTA metric, which is very sensitive to the detection threshold. Instead, IDF1 score compares ground truth trajectory and computes trajectory by a bipartite graph, which reflects how long of an object has been correctly tracked. As a result, IDF1 is used as the first metric in the comparison. From the table, we can see that our IDF1 score is much higher than most state-of-the-art methods. For other metrics shown in the table, we are also among the top rankings. Notice that although LSST17 [5] has better performance than our

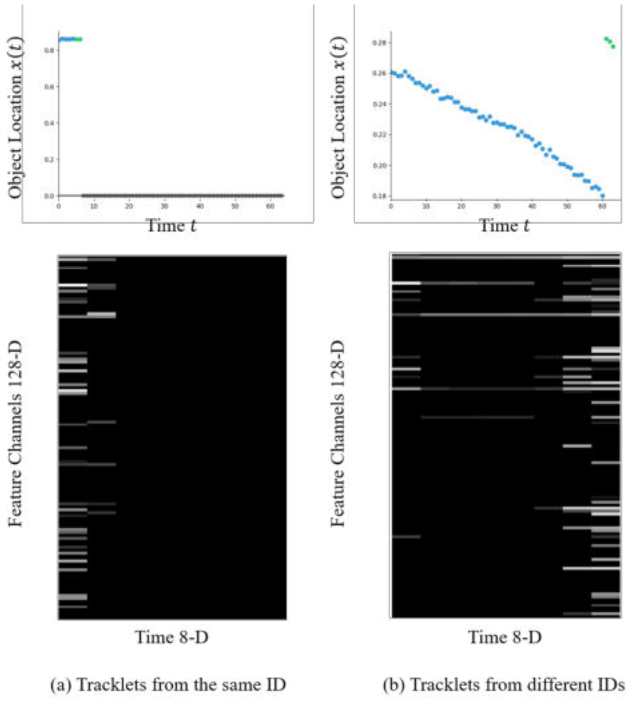

Figure 4: Two examples of the feature maps. For each example, the top figure shows the spatial locations of the two input tracklets in the 64-frame time window. The bottom figure is the corresponding feature map after the max pooling of `Conv3` with the kernel size 5, which aligns with the figure in the top row in the time domain. We can see that the connection part of the input tracklets in the time domain have strong activation.

method, it uses extra private datasets for training the SOT sub-net and Re-ID sub-net, which is not a fair comparison. The trackers with extra datasets for training are listed in the top half of each table while the trackers in the bottom half are trained purely on MOT datasets. As shown in the table, our proposed TNT has promising results on both MOT16 and MOT17 datasets compared with other state-of-the-art methods.

Qualitative Results for Different Scenarios. With the trained model on the MOT dataset, we also test our tracker on other scenarios without any fine-tuning. Promising results are also achieved. Figure 5 shows some qualitative tracking results using our tracker on other applications, like 3D pose estimation and UAV applications.

5.5 Ablation Study

Occlusion Handling. Occlusion is one of the major challenges in MOT. Our framework can easily handle both partial and full occlusions even with a long-time duration. As we know, when a person is occluded, the detection as well as appearance features are unreliable. If we discover a large change in appearance during generation of the tracklets, we just stop detection association, even if the detection result is available. After several or tens of frames,

Tracker	IDF1 ↑	MOTA ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDsw. ↓	Frag ↓
GCRA [21]	48.6	48.2	12.9%	41.1%	5,104*	88,586	821	1,117
MOTDT [19]	50.9	47.6	15.2%	38.3%*	9,253	85,431*	792	1,858
LMP [36]	51.3	48.8*	18.2%	40.1%	6,654	86,245	481	595*
MCjoint [14]	52.3*	47.1	20.4%*	46.9%	6,703	89,368	370*	598
oICF [15]	49.3	43.2	11.3%	48.5%	6,651	96,515	381	1,404
NOMT [2]	53.3	46.4	**18.3%**	41.4%	9,753	87,565	**359**	**504**
DMMOT [48]	54.8	46.1	17.4%	42.7%	7,909	89,874	532	1,616
TLMHT [32]	55.3	48.7	15.7%	44.5%	**6,632**	86,504	413	642
TNT (Ours)	**56.1**	**49.2**	17.3%	**40.3%**	8,400	**83,702**	606	882

Table 1: Tracking performance on the MOT16 testing set. The first four trackers are trained with extra training data, and the best results are marked with stars. For the rest trackers, best in bold, second best in blue.

Tracker	IDF1 ↑	MOTA ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDsw. ↓	Frag ↓
HAM_SADF17 [44]	51.1	48.3	17.1%	41.7%	20,967*	269,038	1,871	3,020*
EDMT17 [1]	51.3	50.0	21.6%*	36.3%	32,279	247,297	2,264	3,260
MOTDT17 [19]	52.7	50.9	17.5%	35.7%*	24,069	250,768	2,474	5,317
LSST17 [5]	62.3*	54.7*	20.4%	40.1%	26,091	228,434*	1,243*	3,726
MHT_DAM [16]	47.2	50.7	20.8%	36.9%	**22,875**	252,889	2,314	**2,865**
FWT [9]	47.6	51.3	21.4%	**35.2%**	24,101	247,921	2,648	4,279
jCC [13]	54.5	51.2	20.9%	37.0%	25,937	247,822	**1,802**	2,984
DMAN [48]	55.7	48.2	19.3%	38.3%	26,218	263,608	2,194	5,378
TNT (Ours)	**58.0**	**51.9**	**23.5%**	35.5%	37,311	**231,658**	2,294	2,917

Table 2: Tracking performance on the MOT17 testing set. The first four trackers are trained with extra training data, and the best results are marked with stars. For the rest trackers, best in bold, second best in blue.

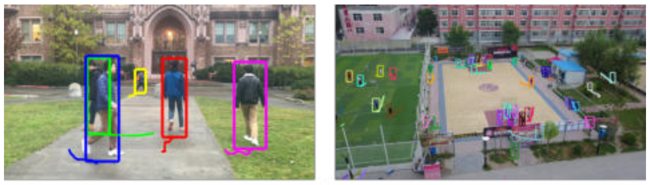
(a) Campus Scenario (b) UAV Scenario

Figure 5: Tracking in other scenarios. (a) Tracking on campus pose estimation dataset. 3D human pose can be further estimated using the tracking results. (b) Tracking for UAV applications.

when a person re-appears from occlusion, a new tracklet will be assigned to the person. Afterwards, the connectivity between these two tracklets will be measured based on the TrackletNet to distinguish whether they are from the same person. Once they are confirmed with the same ID, we can easily fill out the missing detections with linear interpolation. Figure 6 shows qualitative results for handling occlusions. The first row of Figure 6 is from MOT17-08 sequence. At frame 566, the person with a red bounding box is fully occluded by a statue. But it can be correctly tracked after it appears again at frame 604. The second row is one example from MOT17-01 sequence, the person with the red bounding box goes across five other pedestrians, but the IDs of all targets keep consistent along the time. The last row shows the person with a yellow bounding box is crossing the street from MOT17-06 sequence captured with a moving camera. Although it is occluded by several other pedestrians, it can be still effectively tracked in a long run.

Effectiveness of Tracklet Generation with Epipolar Geometry. To check the effectiveness of EG in the tracklet generation, we run detection association on MOT17-10 and MOT17-13 because these two sequences have large camera motion. Table 3 shows the results with/without epipolar geometry. Two types of error rates are evaluated, i.e., false discovery rate (FDR) and false negative rate (FNR), which are defined as follows,

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{TP} + \mathrm{FP}}, \quad \mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}}, \tag{8}$$

where TP, FP and FN represent true positive, false positive and false negative, respectively.

From Table 3, we can see that FDR is quite small in both cases. This means only a small portion of incorrect associations is involved in the tracklet generation. On the other hand, FNR largely drops with epipolar geometry adopted, especially for MOT17-13, which means more detections are correctly associated and also reflects the effectiveness of the proposed tracklet generation strategy.

Robustness to Appearance Features. Another major advantage of our TrackletNet is the ability to address the over-fitting

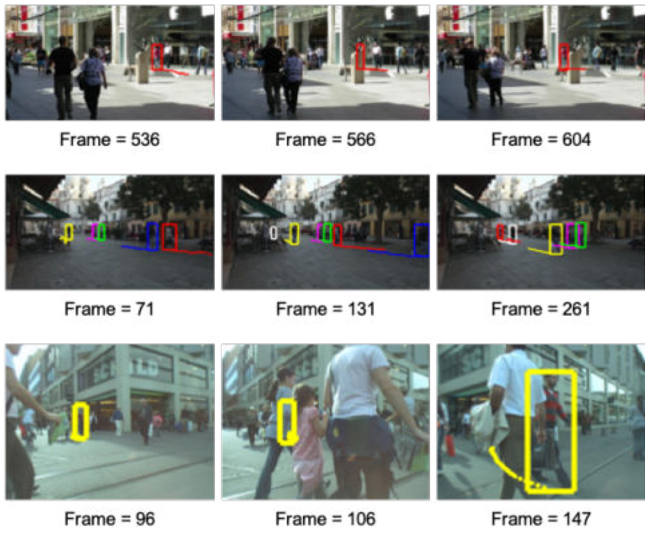

Figure 6: Occlusion handling in different MOT sequences.

Video Seq.	EG Involved	FDR (%)	FNR (%)
MOT17-10	×	2.4	6.5
	✓	**2.4**	**5.9**
MOT17-13	×	3.6	12.4
	✓	**3.4**	**9.7**

Table 3: The effectiveness of tracklet generation with EG.

Noise (Std)	Method	IDF1	MOTA	IDsw.
$\sigma = 0.05$	Baseline	31.7	22.4	23
	TNT	**34.1**	**22.5**	**20**
$\sigma = 0.1$	Baseline	31.1	22.1	26
	TNT	**34.1**	**22.3**	**21**
$\sigma = 0.2$	Baseline	20.6	19.0	80
	TNT	**34.0**	**22.5**	**20**

Table 4: The robustness of TNT compared with the baseline method to disturbed appearance features.

issue with a small dataset in the training. Different from [21], our TrackletNet is trained only on MOT dataset without using additional tracking datasets, but we can still achieve very good performance on other testing datasets. This is because of the dimension independence of appearance features in training the network with convolutions only conducted in the time domain. As a result, the complexity of the network is largely reduced, which also decreases the effect of the over-fitting.

To test the model robustness to appearance features, we disturb the appearance features with Gaussian noise on MOT17-02 sequence. The compared baseline method is using the Bhattacharyya distance of appearance features between the input pair of tracklets as the edge cost, which is commonly used in person Re-ID tasks.

Architecture	IDF1	MOTA
TNT w/o appearance features	38.8	27.6
TNT w/o location features	41.1	28.2
Original TNT	43.7	28.6

Table 5: The ablation study on the TNT architecture.

The comparison results are shown in Table 4 with Gaussian noise using different standard deviations (Std). From the table, we can see the baseline method degrades largely with the increasing of noise level, while the tracking performance is not affected much for TNT. This is because TNT measures the temporal continuity of features as the similarity rather than using feature distance itself, which can largely suppress unreliable detections or noise.

Ablation Study on the TNT Architecture. One major advantage of the TNT architecture is to combine the location and appearance features in one unified framework and extract useful features along the temporal domain. To test the effectiveness, we change the architecture with two variants, i.e., 1) input appearance features without location features; 2) input location features without appearance features and remove the `AvgPool` layer. In both variants, the size of the fully connected layers is changing accordingly. Table 5 shows the results on MOT17-02 sequence with Faster-RCNN detection as inputs. The results show that the performance degrades compared with the original TNT, which further proves that the combination of the location and appearance features is very important in the MOT.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a novel multi-object tracking method, called TrackletNet Tracker (TNT), based on a tracklet-based graph model, including the tracklet generation with epipolar geometry and the connectivity measurement by a multi-scale TrackletNet. Our TNT outperforms most state-of-the-art methods on MOT16 and MOT17 benchmarks. We also show some qualitative results on different scenarios and applications using TNT. Robustness of TNT is further discussed with handling occlusions.

However, fast camera motion is still a challenge in 2D tracking. In our future work, we are going to convert 2D tracking to 3D tracking with the help of visual odometry. Once the 3D location of the object in the world coordinate can be estimated, the trajectory of the object should be much more smooth than the 2D case, resulting in more reliable tracking.

REFERENCES

[1] Jiahui Chen, Hao Sheng, Yang Zhang, and Zhang Xiong. 2017. Enhancing detection model for multiple hypothesis tracking. In *Conf. on Computer Vision and Pattern Recognition Workshops*. 2143–2152.

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

[3] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. 2017. Detect to track and track to detect. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 3038–3046.

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

[5] Weitao Feng, Zhihao Hu, Wei Wu, Junjie Yan, and Wanli Ouyang. 2019. Multi-Object Tracking with Multiple Cues and Switcher-Aware Classification. *arXiv preprint arXiv:1901.06129* (2019).

[6] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 2014. 3d traffic scene understanding from movable platforms. *IEEE transactions on pattern analysis and machine intelligence* 36, 5 (2014), 1012–1025.

[7] Renshu Gu, Gaoang Wang, and Jenq-Neng Hwang. 2019. Efficient Multi-person Hierarchical 3D Pose Estimation for Autonomous Driving. In *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. IEEE, 163–168.

[8] Richard Hartley and Andrew Zisserman. 2003. *Multiple view geometry in computer vision.* Cambridge university press.

[9] Roberto Henschel, Laura Leal-Taixé, Daniel Cremers, and Bodo Rosenhahn. 2018. Fusion of head and full-body detectors for multi-object tracking. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*.

[10] Hung-Min Hsu, Tsung-Wei Huang, Gaoang Wang, Jiarui Cai, Zhichao Lei, and Jenq-Neng Hwang. 2019. Multi-Camera Tracking of Vehicles based on Deep Features Re-ID and Trajectory-Based Camera Link Models. In *AI City Challenge Workshop, IEEE/CVF Computer Vision and Pattern Recognition (CVPR) Conference, Long Beach, California.*

[11] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. 2017. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology* (2017).

[12] Kai Kang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. 2016. Object detection from video tubelets with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 817–825.

[13] Margret Keuper, Siyu Tang, Bjorn Andres, Thomas Brox, and Bernt Schiele. 2018. Motion Segmentation & Multiple Object Tracking by Correlation Co-Clustering. *IEEE transactions on pattern analysis and machine intelligence* (2018).

[14] Margret Keuper, Siyu Tang, Yu Zhongjie, Bjoern Andres, Thomas Brox, and Bernt Schiele. 2016. A multi-cut formulation for joint segmentation and tracking of multiple objects. *arXiv preprint arXiv:1607.06317* (2016).

[15] Hilke Kieritz, Stefan Becker, Wolfgang Hübner, and Michael Arens. 2016. Online multi-person tracking using integral channel features. In *Advanced Video and Signal Based Surveillance (AVSS), 2016 13th IEEE International Conference on.* IEEE, 122–130.

[16] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. 2015. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE International Conference on Computer Vision.* 4696–4704.

[17] Chanho Kim, Fuxin Li, and James M Rehg. 2018. Multi-object Tracking with Neural Gating Using Bilinear LSTM. In *Proceedings of the European Conference on Computer Vision (ECCV).* 200–215.

[18] Ratnesh Kumar, Guillaume Charpiat, and Monique Thonnat. 2014. Multiple object tracking by efficient graph partitioning. In *Asian Conference on Computer Vision.* Springer, 445–460.

[19] C Long, A Haizhou, Z Zijie, and S Chong. 2018. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. ICME.

[20] Yongyi Lu, Cewu Lu, and Chi-Keung Tang. 2017. Online video object detection using association LSTM. In *Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy.* 22–29.

[21] Cong Ma, Changshui Yang, Fan Yang, Yueqing Zhuang, Ziwei Zhang, Huizhu Jia, and Xiaodong Xie. 2018. Trajectory Factory: Tracklet Cleaving and Re-connection by Deep Siamese Bi-GRU for Multiple Object Tracking. *arXiv preprint arXiv:1804.04555* (2018).

[22] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. 2016. MOT16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831* (2016).

[23] Anton Milan, Seyed Hamid Rezatofighi, Anthony R Dick, Ian D Reid, and Konrad Schindler. 2017. Online Multi-Target Tracking Using Recurrent Neural Networks.. In *AAAI*, Vol. 2. 4.

[24] Anton Milan, Konrad Schindler, and Stefan Roth. 2016. Multi-target tracking by discrete-continuous energy minimization. *IEEE transactions on pattern analysis and machine intelligence* 38, 10 (2016), 2054–2068.

[25] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE transactions on robotics* 31, 5 (2015), 1147–1163.

[26] Christos H Papadimitriou and Mihalis Yannakakis. 1991. Optimization, approximation, and complexity classes. *Journal of computer and system sciences* 43, 3 (1991), 425–440.

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

[28] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. 2016. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision.* Springer, 17–35.

[29] Ergys Ristani and Carlo Tomasi. 2018. Features for Multi-Target Multi-Camera Tracking and Re-Identification. *arXiv preprint arXiv:1803.10859* (2018).

[30] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. 2017. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv preprint arXiv:1701.01909* 4, 5 (2017), 6.

[31] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 815–823.

[32] Hao Sheng, Jiahui Chen, Yang Zhang, Wei Ke, Zhang Xiong, and Jingyi Yu. 2018. Iterative Multiple Hypothesis Tracking with Tracklet-level Association. *IEEE Transactions on Circuits and Systems for Video Technology* (2018).

[33] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 1–9.

[34] Siyu Tang, Bjoern Andres, Miykhaylo Andriluka, and Bernt Schiele. 2015. Subgraph decomposition for multi-target tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 5033–5041.

[35] Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. 2016. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision.* Springer, 100–111.

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

[37] Zheng Tang, Milind Naphade, Ming-Yu Liu, Xiaodong Yang, Stan Birchfield, Shuo Wang, Ratnesh Kumar, David Anastasiu, and Jenq-Neng Hwang. 2019. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 8797–8806.

[38] Zheng Tang, Gaoang Wang, Hao Xiao, Aotian Zheng, and Jenq-Neng Hwang. 2018. Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features. In *CVPR Workshop (CVPRW) on the AI City Challenge.*

[39] Gaoang Wang, Jenq-Neng Hwang, Kresimir Williams, and George Cutter. 2016. Closed-Loop Tracking-by-Detection for ROV-Based Multiple Fish Tracking. In *Computer Vision for Analysis of Underwater Imagery (CVAUI), 2016 ICPR 2nd Workshop on.* IEEE, 7–12.

[40] Gaoang Wang, Xinyu Yuan, Aotian Zhang, Hung-Min Hsu, and Jenq-Neng Hwang. 2019. Anomaly Candidate Identification and Starting Time Estimation of Vehicles from Traffic Videos. In *AI City Challenge Workshop, IEEE/CVF Computer Vision and Pattern Recognition (CVPR) Conference, Long Beach, California.*

[41] Longyin Wen, Wenbo Li, Junjie Yan, Zhen Lei, Dong Yi, and Stan Z Li. 2014. Multiple target tracking based on undirected hierarchical relation hypergraph. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 1282–1289.

[42] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. 2017. Simple online and realtime tracking with a deep association metric. In *Image Processing (ICIP), 2017 IEEE International Conference on.* IEEE, 3645–3649.

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

[44] Young-chul Yoon, Abhijeet Boragule, Kwangjin Yoon, and Moongu Jeon. 2018. Online Multi-Object Tracking with Historical Appearance Matching and Scene Adaptive Detection Filtering. *arXiv preprint arXiv:1805.10916* (2018).

[45] Hongyi Zhang, Andreas Geiger, and Raquel Urtasun. 2013. Understanding high-level semantics by modeling traffic patterns. In *Proceedings of the IEEE international conference on computer vision.* 3056–3063.

[46] Zhimeng Zhang, Jianan Wu, Xuan Zhang, and Chi Zhang. 2017. Multi-Target, Multi-Camera Tracking by Hierarchical Clustering: Recent Progress on DukeMTMC Project. *arXiv preprint arXiv:1712.09531* (2017).

[47] Zhun Zhong, Liang Zheng, Donglin Cao, and Shaozi Li. 2017. Re-ranking person re-identification with k-reciprocal encoding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 1318–1327.

[48] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. 2018. Online Multi-Object Tracking with Dual Matching Attention Networks. In *Proceedings of the European Conference on Computer Vision (ECCV).* 366–382.