

Joint Object Detection and Multi-Object Tracking with Graph Neural Networks

Yongxin Wang, Kris Kitani, Xinshuo Weng

Abstract—Object detection and data association are critical components in multi-object tracking (MOT) systems. Despite the fact that the two components are dependent on each other, prior works often design detection and data association modules separately which are trained with separate objectives. As a result, one cannot back-propagate the gradients and optimize the entire MOT system, which leads to sub-optimal performance. To address this issue, recent works simultaneously optimize detection and data association modules under a joint MOT framework, which has shown improved performance in both modules. In this work, we propose a new instance of joint MOT approach based on Graph Neural Networks (GNNs). The key idea is that GNNs can model relations between variable-sized objects in both the spatial and temporal domains, which is essential for learning discriminative features for detection and data association. Through extensive experiments on the MOT15/16/17/20 datasets, we demonstrate the effectiveness of our GNN-based joint MOT approach and show state-of-the-art performance for both detection and MOT tasks. Our code is available at: <https://github.com/yongxinw/GSDT>.

I. INTRODUCTION

Object detection [1]–[8] and data association [9]–[16] are two components of multi-object tracking (MOT), which is essential to perception in robotic systems such as autonomous driving [17]–[21]. Prior work [10], [14] often approaches MOT in an online fashion using a tracking-by-detection pipeline, where a detector outputs detections followed by a data association module matching the detections with past tracklets to form new tracklets up to the current frame. Oftentimes, the detector and data association modules are trained separately in prior work. However, with this separate optimization procedure, we cannot back-propagate errors through the entire MOT system. In other words, each module is optimized only towards its own local optimum, but not towards the objective of MOT. As a result, this separate optimization procedure used in prior work often yields sub-optimal performance.

To improve performance, we investigate 1) joint optimization of object detection and data association, which we refer to as the joint MOT framework; 2) within the joint MOT framework, how to learn more discriminative features. First, to address the joint MOT problem, prior work [22]–[31] has explored different directions. [22]–[25] proposed to unify object detector with a model-free single-object tracker, where the tracker directly regresses the location of each object detected in the previous frame to the current frame. As each object is tracked independently, the data association problem is naturally resolved. [26]–[29] proposed to extend an object detector by adding a re-identification (Re-ID) [32]

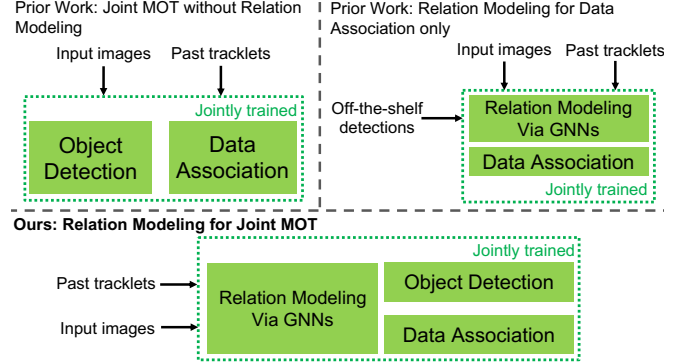


Fig. 1. (Top left): Although jointly training detection and data association, prior work does not take into account object-object relations. (Top right): Prior work leverages object-object relations but only adopts it for data association. By using off-the-shelf detections that cannot be optimized jointly, such disjoint MOT paradigm can lead to sub-optimal MOT performance. (Bottom): Our method leverages spatial-temporal object relations for both detection and data association under a joint MOT framework.

or ID verification branch which extracts features of objects for matching across frames. Also, [30], [31] proposed an anchor tube. Different from anchor box used in anchor-based detectors, anchor tube represents a sequence of bounding boxes in a list of frames (one box per frame). Given video clips as inputs, true positive tubes can be found and used as tracklet outputs, which solves the joint MOT problem in a single shot.

Although prior joint MOT methods achieved impressive performance, we observed that the feature extracted for individual object/tracklet/tube is independent of one another and object-object relations are ignored. We argue that such object relations are useful to both object detection and data association. For example, for object detection in MOT, if related objects (*e.g.*, two pedestrians that walk together) co-occur in the last frame, it is likely that they will also co-occur in the current frame at a nearby location. For data association, if the similarity score of two objects across frames increases with a high confidence (*i.e.*, it is likely that these two objects have the same identity), then the similarity score between any of these two objects and other objects should be suppressed to avoid confusion in data association. As a result, to achieve better performance for the joint MOT framework, we should design our method to leverage object-object relations. While recent works [33]–[35] have exploited object-object relations in data association, they are limited to disjoint MOT where the detector is optimized separately. Therefore, the object relations are not used for the detector which is sub-optimal.

To deal with the above issue, we propose a new instance of joint MOT approach that can model object-object relations for both object detection and data association. Specifically,

to obtain more discriminative features, we use Graph Neural Networks (GNNs) to exploit object-object relations. Then, the obtained features that consider object relations are used for both tasks of detection and data association. With GNNs, the feature extracted for each object is not isolated anymore, but instead can be adapted via features of its related objects in both spatial and temporal domains. To the best of our knowledge, our work is the first to model object relations via GNNs in the joint MOT framework. Our work overcomes the limit of prior work that 1) can perform joint MOT while ignoring object relations, or 2) can model object relations in association but cannot address joint MOT and use object relations to improve detection. Through experiments on MOTChallenges, we show S.O.T.A. performance on MOT15/20 and competitive performance on MOT16/17 datasets. Our contributions are summarized as follows.

- 1) A joint MOT approach that models object relations via GNNs to improve both detection and data association;
- 2) Strong empirical performance on MOT 15/16/17/20 datasets for both detection and MOT tasks.

II. RELATED WORKS

Object Detection. There have been tremendous advancements in image-based detection since large datasets such as COCO [36] have been introduced. In past few years, anchor-based object detectors [2], [4], [37], [38] have been dominant in the field of object detection. Recently, a new type of detector that models objects as points has been proposed [39], [40] and achieved impressive performance. However, image-based methods suffer from unstable detections across frames as they use a single image as input. To deal with the issue, video detection [41]–[43] has been investigated which uses multiple frames as inputs. Although producing more temporally-consistent detections than image-based methods, existing video-based methods cannot model object relations. Similar to video-based detection methods, our joint MOT method also takes multiple frames as inputs to obtain detections. However, our difference is that, in addition to leveraging multiple frames of input, we also model object relations via GNNs, which can further improve detection performance when the objects across consecutive frames are highly related to one another. (e.g., object co-occurring).

Multi-Object Tracking. Recent MOT work primarily focuses on data association component in the tracking-by-detection pipeline, which can be split into online and batch methods. Online methods [10], [14], [44], [45] only require information up to the current frame for data association and can be useful to online applications. On the other hand, batch methods [46]–[51] need to have access to global information (i.e., information up to the current frame plus information from future frames), which can theoretically achieve higher accuracy than online methods but are not applicable to online scenarios. We restrict the scope of this paper to online methods. Different from prior online methods that use off-the-shelf detections and only focus on the data association component, our method jointly optimizes detection and data

association with an additional GNN module for object-object relation modeling which improves performance in both tasks.

Joint Detection and Data Association. To enable gradient back-propagation in the joint MOT framework and improve overall performance, [22]–[31] attempted to jointly optimize detection and data association for MOT. As introduced before, prior joint MOT methods can be mostly split into three categories: 1) unify a single-object tracker with an object detector; 2) add a Re-ID branch to object detection network; 3) use anchor tube representation for one-shot joint MOT. For example, [24] builds on top of Faster-RCNN [2] and makes its bounding box regression head multi-purpose, i.e., not only to refine the detected bounding boxes but also to serve as a single object tracker that regresses the object location from the previous frame to the current frame. Our method shares a similar spirit with prior joint MOT methods but goes beyond them by modeling spatial-temporal object relations. We show that our method can improve performance for both object detection and data association tasks in MOT.

Graph Neural Networks for Relation Modeling. GNNs were first introduced by [52] to process data with a graph structure using neural networks. The key idea is to construct a graph with nodes and edges relating each other and update node/edge features based on relations, i.e., a process called node feature aggregation. In recent years, different GNNs (e.g., GraphConv [53], GCN [54], GAT [55], etc) have been proposed each with a unique feature aggregation rule which are shown to be effective on various tasks. Specifically for MOT, recent work [33]–[35], [56] formulates data association as an edge classification problem using GNNs, where each node represents an object and each edge relating two nodes denotes the similarity between detection and tracklets. These approaches use GNNs to update node features via object relation modeling and have shown improved MOT performance. As opposed to these methods that focus only on using GNNs to improve data association, our work uses GNNs for joint detection and association, where the detection branch also benefits from the GNN relation modeling.

III. APPROACH

We follow online MOT methodology. Given images F_{t-1} , F_t at frame $t-1$, t and tracklets at frame $t-1$, denoted as $T_{t-1} = \{T_{t-1}^1, T_{t-1}^2, \dots, T_{t-1}^{P_{t-1}}\}$ as inputs, we aim to detect objects $D_t = \{D_t^1, D_t^2, \dots, D_t^{K_t}\}$ in F_t , and associate them to tracklets T_{t-1} , where K_t and P_{t-1} are number of detections and tracklets which can vary across frames. By associating detections to past tracklets, we can determine whether to extend or terminate an existing tracklet or to initiate a new tracklet at frame t . We perform this process iteratively at every frame to obtain the tracklets of the entire video.

To simultaneously detect and associate objects for MOT, we integrate a detector and a Re-ID module in our model. However, doing so alone does not leverage spatial-temporal object relations. Therefore, we use GNNs to extract relations between objects and learn better features to improve both detection and data association. For illustration, we show the

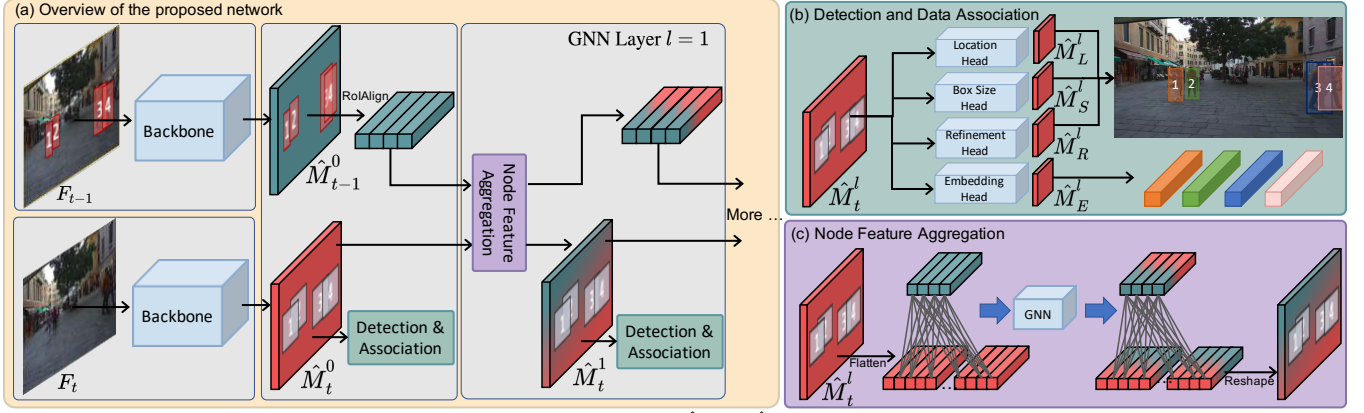


Fig. 2. (a) **Overview of the Proposed Network.** We first extract features $\hat{M}_{t-1}^0, \hat{M}_t^0$ from images F_{t-1} and F_t using a **shared backbone**. To obtain feature of each tracklet in T_{t-1} , we use **RoIAlign** to crop feature from the image feature \hat{M}_{t-1}^0 given the tracklets' boxes (red boxes in \hat{M}_{t-1}^0). To obtain features of potential detections, we use **feature of every pixel in \hat{M}_t^0** . To construct a graph with manageable number of edges, we only define edges between features of potential detections to tracklets if their **spatial distances are within a window** (grey boxes in \hat{M}_t^0). With the constructed graph, we use **3-layer GNNs** to update features of tracklets and potential detections via **node feature aggregation**. A detection and data association head is applied to every layer of GNNs to obtain final detections and matching. (b) **Detection and Data Association:** The location, box size, and refinement heads generate \hat{M}_L^l, \hat{M}_S^l , and \hat{M}_R^l which are used to obtain detections. The embedding head generates \hat{M}_E^l to compute identity embedding for data association. (c) **Node Feature Aggregation.** The mixed color illustrates that features from tracklets and potential detections affect each other via relation modeling.

proposed network architecture in Fig. 2, which contains four modules: feature extraction (Sec. III-A), node feature aggregation (Sec. III-C), detection (Sec. III-A) and association (Sec. III-B). As we use **GNNs** for **Simultaneous Detection and Tracking**, we abbreviate our method as **GSdT**.

A. Feature Extraction and Object Detection

Given two input images F_{t-1} and F_t , we first use a shared **DLA-34 backbone** [57] to extract feature maps at two frames $\hat{M}_{t-1}^0, \hat{M}_t^0 \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r} \times C}$ as shown in Fig. 2 (a) left, where r is the downsample ratio, W/H are width and heights of the images, and C is the number of channels. Both image features $\hat{M}_{t-1}^0, \hat{M}_t^0$ will be used in the following relation modeling, object detection and data association modules. Also, for the image feature at frame t , we will update it at the l th layer of GNNs to obtain \hat{M}_t^l (Sec. III-C).

For object detection, we follow **CenterNet** [58] and detect each object by finding its center coordinate (x, y) and width/height (w, h) . As we only perform detection in the frame t , we feed \hat{M}_t^l to three heads (location, box size and refinement heads) as shown in Fig. 2 (b), which provides three maps, *i.e.*, a location map $\hat{M}_L^l \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r}}$, a size map $\hat{M}_S^l \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r} \times 2}$, and a refinement map $\hat{M}_R^l \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r} \times 2}$. As the names suggest, \hat{M}_L^l provides rough estimates of object center coordinates, \hat{M}_S^l gives estimated widths and heights, and \hat{M}_R^l refines rough coordinates to be precise. Note that these three detection heads can operate on any GNN layer l .

Training. To train three detection heads and back-propagate the gradients through the entire network, we need to construct ground truth (GT) for each map and apply loss functions. For GT location map $M_L^l \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r}}$, we use a **Gaussian heatmap where the peaks occupy the locations of GT objects**. Specifically, the value at the location (i, j) is:

$$M_L^l(i, j) = \sum_{k=1}^N \exp\left(-\frac{(i - \lfloor \frac{x_k}{r} \rfloor)^2 + (j - \lfloor \frac{y_k}{r} \rfloor)^2}{2\sigma_k^2}\right), \quad (1)$$

where N is number of GT objects and (x_k, y_k) is the center coordinate of GT object k , and $\lfloor \cdot \rfloor$ is a floor function. The standard deviation σ_k scales with the GT box size (w_k, h_k) [40]. For GT box size and refinement maps M_S^l, M_R^l , if there exists a GT object center at location (i, j) , then:

$$M_S^l(i, j, :) = (w, h), \quad (2)$$

$$M_R^l(i, j, :) = \left(\frac{x}{r} - \lfloor \frac{x}{r} \rfloor, \frac{y}{r} - \lfloor \frac{y}{r} \rfloor\right). \quad (3)$$

After we construct the GT for three maps, we use **focal loss** for the location map L_{loc}^l , and L1 loss for the size map L_{size}^l and refinement map L_{ref}^l . Note that for size/refinement maps, their losses are only applied to locations with GT objects. The overall loss for detection is then:

$$L_{det}^l = \lambda_1 L_{loc}^l + \lambda_2 L_{size}^l + \lambda_3 L_{ref}^l, \quad (4)$$

where we empirically found $\lambda_1 = \lambda_3 = 1, \lambda_2 = 0.1$ could achieve a well balance between each individual loss.

B. Data Association

To match detections with tracklets, we **add an additional embedding head** as shown in Fig. 2 (b) to learn an identity embedding for each potential detection, *i.e.*, every pixel in \hat{M}_t^l . These identity embeddings are used to compute similarity between every pair of tracklet and potential detection during testing, where the identity embeddings of tracklets are cached when they are detected in the previous frame. Specifically, given \hat{M}_t^l as inputs, our embedding head outputs an embedding map $\hat{M}_E^l \in \mathbb{R}^{\frac{W}{r} \times \frac{H}{r} \times D}$, where D denotes the embedding dimension and each pixel in \hat{M}_E^l has an embedding for a potential detection centered at this pixel.

Training. To optimize our embedding head, we further use embedding at each pixel of the embedding map \hat{M}_E^l as input, and predict its identity vector $\hat{\mathbf{p}}$, the i th entry of which represents the probability of this pixel has an object identity of i . Also, we will need the GT identity vector \mathbf{p} for training, which is a one-hot vector of length M (M is the number

of object identities in the target dataset). Note that in the GT identity vector \mathbf{p} , only the GT identity index is filled with 1 while all other indexes are 0s. Similar as before, we only supervise our embedding map \hat{M}_E^l at pixels where there exists a GT object using a cross-entropy loss as follows:

$$L_{emb}^l = \frac{1}{N} \sum_{k=1}^N \sum_{m=1}^M \mathbf{p}^k(m) \log(\hat{\mathbf{p}}^k(m)). \quad (5)$$

C. Graph Neural Networks for Relation Modeling

As detection and data association modules alone overlook spatial-temporal relations between tracklets and detections, we use GNNs to further leverage such relation information and obtain better features for joint MOT.

Graph Construction. To perform graph feature learning, we need to construct a graph $G(V, E)$ with nodes V being features of detections and tracklets. For features of tracklets, we can easily obtain them as tracklets' boxes are given as inputs. Specifically, we crop tracklet features from the image feature \hat{M}_{t-1}^0 using the RoIAlign [38] operator. However, it is relatively non-trivial to obtain features of detections. This is because objects have not been detected in \hat{M}_t^l so we do not have their locations to crop features from \hat{M}_t^l . To overcome this issue, we use feature at every pixel of \hat{M}_t^l to represent a potential detection, resulting in $\frac{W}{r} \times \frac{H}{r}$ detection nodes. This design choice is made because we believe that \hat{M}_t^l contains enough information about detections, as all three detection heads are branched from \hat{M}_t^l .

In addition to node construction, we also need the set of edges E . A simple solution is to define an edge between every pair of nodes, resulting in a fully-connected graph. However, such a graph with massive number of edges can be computationally expensive and sometimes impractical when the number of tracklets and potential detections (*i.e.*, number of pixels in \hat{M}_t^l) are too large. Instead, we take advantage of domain knowledge in MOT: (a) data association only happens across frames (not in the same frame); (b) displacement of the same object is often small across frames. According to (a), we only define edges between tracklet nodes and detection nodes. Based on (b), for every tracklet node, we only connect it to detection nodes that are in nearby locations, *i.e.*, within a $s \times s$ spatial window centered at the tracklet location in F_{t-1} . We use $s = 15$ in our model which balances computation and performance. We illustrate the idea of edge construction in Fig. 2, where grey boxes on \hat{M}_t^l enclose the pixels (detection nodes) that have edges with tracklets.

Node Feature Aggregation. The key idea of our method is using GNNs to model object-object relations and improve feature learning for both detection and data association. To that end, we iteratively update node features by aggregating features from its neighborhood nodes connected with edges. In this way, information can propagate through the graph and nodes are allowed to interact. So the question is how exactly we should perform the node feature aggregation?

Recently, there are many GNNs proposed, *e.g.*, GraphConv [53], AGNNConv [59], EdgeConv [60], each with a different node aggregation rule. To explore how different GNNs affect

our joint MOT method, we try five popular GNNs in our model in the ablation study and compare their performance. Our final model uses GraphConv as it provides the best empirical performance. Specifically, GraphConv updates node features using the following rule:

$$h_l^i = \rho_1(h_{l-1}^i) + \sum_{j \in \mathcal{N}(i)} \rho_2(h_{l-1}^j), \quad (6)$$

where h_l^i represents the feature of node i at GNN layer l , $\mathcal{N}(i)$ defines neighborhood of node i , and ρ_1, ρ_2 are linear layers. Note that, for detection nodes that have no edge to any tracklets, the second term in Eq. 6 disappears, and these potential detection nodes are only used for discovering new objects in frame t (not having corresponding tracklet in frame $t-1$). Also, one might argue that, as our edges are only defined across frames, object-object relations are only modeled in the temporal domain for data association but not in the spatial domain for object detection, *i.e.*, nodes within the same frame cannot interact. This is true if we only have a single layer of GNNs. In the case we use more than one layer, the information can be propagated back and forth, enabling spatial-temporal object relation modeling. For example, in the first layer, the feature of a detection node i_1 is aggregated to a tracklet node j . Then in the second layer, the feature of the tracklet node j is propagated to a different detection node i_2 , *i.e.*, part of the node feature i_1 is propagated to the detection node i_2 which enables spatial relation modeling in the same frame. In ablation study, we will explore how the number of GNN layers will affect performance of our model. In our best model, we use three layers of GNNs.

D. Joint Detection and Association with GNNs

Although the detection and association heads introduced in Sec. III-A and III-B can solve joint detection and data association problem by training two modules together, they do not leverage object relations if they only use features from $\hat{M}_{t-1}^0, \hat{M}_t^0$. As shown in Fig. 2, to leverage GNNs, we apply the detection and association head also to features obtained in GNN layer 1, 2, *etc.*, which have better features after node feature aggregation by encoding relations. To summarize, the overall loss of our network is a summation of detection and data association losses over all layers of GNNs:

$$L_{total} = \sum_l \eta_1 L_{det}^l + \eta_2 L_{emb}^l, \quad (7)$$

where l is the index of GNN layers and η_1, η_2 denote weights of each loss. We adopt an automatic loss weighting scheme as in [61] to balance η_1 and η_2 . We refer our readers to [61] for details about this automatic loss weighting.

E. Inference and Tracking Management

At test time, we iteratively detect objects and associate them to existing tracklets. At every frame, we first obtain estimated detections D_t in F_t by post processing \hat{M}_L^l, \hat{M}_S^l , and \hat{M}_R^l followed by non-maximal suppression. Also, we obtain identity embeddings from \hat{M}_E^l at pixels where objects are detected, which provides us identity embeddings of detections D_t . The identity embeddings for T_{t-1} are cached when they are detected in the previous frame so they are

also available. We use these embeddings to compute affinity matrix \hat{S} where each entry represents the similarity between every pair of tracklet and detection. We then feed \hat{S} to the **Hungarian Algorithm** [62] to find the best matches between D_t and T_{t-1} . For each detection that is not matched to any tracklets, we use it to initialize a new tracklet if its detection confidence score is higher than 0.4. For unmatched tracklets, they might either exit the scene or be still in the scene but not detected. To avoid terminating tracklets that are still in the scene, we use the **Kalman filter** [63] to extend the unmatched tracklets for τ_{term} frames, while keep trying to match these tracklets with detections in following frames. If **after τ_{term} frames an unmatched tracklet still cannot be matched to** any new detection, we terminate this tracklet. Empirically, we found $\tau_{term} = 30$ works well in our model.

IV. EXPERIMENTS

Datasets. We evaluate our method on MOTChallenges [64]–[67], including the 2DMOT15/MOT16/MOT17/MOT20 benchmarks. These datasets have two tracks: public and private, where the public track asks methods to use off-the-shelf detections provided by the datasets while methods in the private track can use their own detections. As we jointly detect and track objects (not using off-the-shelf detections for disjoint MOT), it is straightforward to evaluate our method in private track. For a fair comparison with prior work, we evaluate on the test set by submitting our results to the MOT test server.

Evaluation Metrics. For MOT, we use standard CLEAR MOT metrics [68] and IDF1 metric [69]. For object detection, we report the Average Precision (AP) using official MOT17Det and MOT20Det evaluation protocol.

Implementation Details. We use an Adam optimizer with an initial learning rate of $1e^{-4}$ decaying by a factor of 0.1 at epoch 20 and 27. We train the network for a total of 30 epochs with a batch size of 16. Following TR-MOT [27], we use a mix of **six pedestrian detection datasets for pre-training, including CalTech [70], MOT17 [65], CUHK-SYSU [32], ETH [71], PRW [72], and CityPersons [73].** Same as [27], we removed a few sequences in the above six datasets which overlap with MOT test set to avoid training with test data, for the sake of fair comparison.

Evaluating Multi-Object Tracking. We summarize results on the MOTChallenges for ours and published methods in Table I, marking the **best**, **second best**, and **third best**. Across most metrics (*e.g.*, primary metrics MOTA and IDF1), we achieve the best performance on all 2DMOT2015/MOT16/MOT17/MOT20 datasets among published methods. As the MOTA metric emphasizes more on detection side, we believe our strong MOTA performance shows that our detector is stronger than prior methods. Also, our strong IDF1 performance suggests that our method is able to produce temporally-consistent tracklets due to the spatial-temporal relation modeling via GNNs. Moreover, when comparing with prior joint MOT methods without modeling object relations such as [22],

TABLE I
MOT EVALUATION ON 2DMOT2015/MOT16/MOT17/MOT20 TEST SETS AS OF 2021/03/22 (PUBLISHED METHODS ONLY).

	Method	MOTA(%) \uparrow	IDF1(%) \uparrow	MT(%) \uparrow	ML(%) \downarrow	IDS \downarrow
2DMOT2015	DMT [22]	44.5	49.2	34.7	22.1	684
	Tracktor++V2 [24]	46.6	47.6	18.2	27.9	1,290
	MDP_SubCNN [9]	47.5	55.7	30.0	18.6	628
	CDA_DDAL [74]	51.3	54.1	36.3	22.2	544
	MPNTrack [34]	51.5	58.6	31.2	25.9	375
	Lif.T [75]	52.5	60.0	33.8	25.8	1,047
	EAMTT [76]	53.0	54.0	35.9	19.6	776
	AP_HWDPL [77]	53.0	52.0	29.1	20.2	708
	NOMTwSDP [78]	55.5	<u>59.1</u>	39.0	25.8	<u>427</u>
	RAR15 [79]	<u>56.5</u>	<u>61.3</u>	<u>45.1</u>	<u>14.6</u>	<u>428</u>
MOT16	Tube_TK [30]	<u>58.4</u>	53.1	<u>39.3</u>	<u>18.0</u>	<u>854</u>
	GSDT (Ours)	60.7	64.6	47.0	10.5	480
	MPNTrack [34]	58.6	61.7	27.3	34.0	354
	Lif.T [75]	61.3	64.7	27.0	34.0	<u>389</u>
	DeepSORT_2 [11]	61.4	62.2	32.8	<u>18.2</u>	1,423
	NOMTwSDP16 [78]	62.2	62.6	32.5	<u>31.1</u>	<u>406</u>
	VMaxx [80]	62.6	49.2	32.7	21.1	1,389
	RAR16wVGG [79]	63.0	63.8	<u>39.9</u>	22.1	482
	TAP [81]	64.8	73.5	38.5	21.6	571
	CNNMTT [82]	65.2	62.2	32.5	21.3	946
MOT17	POI [83]	66.1	<u>65.1</u>	34.0	20.8	3,093
	Tube_TK_POI [30]	<u>66.9</u>	62.2	<u>39.0</u>	16.1	1,236
	CTracker_V1 [29]	<u>67.6</u>	57.2	<u>32.9</u>	23.1	1,897
	GSDT (Ours)	74.5	<u>68.1</u>	41.2	<u>17.3</u>	1,229
	MPNTrack [34]	58.8	61.7	28.8	33.5	1,185
	Lif.T [75]	60.5	<u>65.6</u>	27.0	33.6	1,189
	Tube_TK [30]	63.0	58.6	31.2	<u>19.9</u>	4,137
	CTrackerV1 [29]	<u>66.6</u>	57.4	<u>32.2</u>	<u>24.2</u>	5,529
	CTTrack17 [25]	<u>67.8</u>	<u>64.7</u>	<u>34.6</u>	<u>24.6</u>	<u>3,039</u>
	GSDT (Ours)	73.2	66.5	41.7	17.5	3,891
MOT20	SORT20 [10]	42.7	45.1	16.7	<u>26.2</u>	4,334
	Tracktor++V2 [24]	<u>52.6</u>	<u>52.7</u>	<u>29.4</u>	<u>26.7</u>	1,648
	MPNTrack [34]	<u>57.6</u>	<u>59.1</u>	<u>38.2</u>	<u>22.5</u>	1,210
	GSDT (Ours)	67.1	67.5	53.1	13.2	<u>3,133</u>

TABLE II
DETECTION EVALUATION ON MOT17DET/MOT20DET TEST SET.

	Method	AP(%) \uparrow	MODA(%) \uparrow	Recall(%) \uparrow	Precision(%) \uparrow
MOT17	DPM [84]	0.61	31.2	68.1	64.8
	FRCNN [2]	0.72	68.5	77.3	89.8
	ZIZOM [85]	0.81	72.0	83.3	88.0
	SDP [86]	0.81	76.9	83.5	92.6
	F_ViPeD_B [87]	0.89	-14.4	93.2	46.4
	GSDT (Ours)	0.89	78.1	90.7	87.8
MOT20	ViPeD20 [87]	0.80	46.0	86.5	68.1
	GSDT (Ours)	0.81	79.3	88.6	90.6

[30], our method achieves better performance as shown in 2DMOT15/MOT16/MOT17 benchmarks, suggesting that adding object relation modeling is useful in joint MOT. When comparing with prior disjoint MOT methods with relation modeling for data association such as [34], our method also performs better across all benchmarks, suggesting that only modeling object relations in data association is not enough, and that it is useful to model object relations in object detection as well. All these results confirm our hypothesis that modeling object relations in the joint MOT framework can lead to better performance. Note that all numbers in Table I are updated from the **official leaderboard** by 2021/03/22. To visually verify our results, we also provide qualitative results

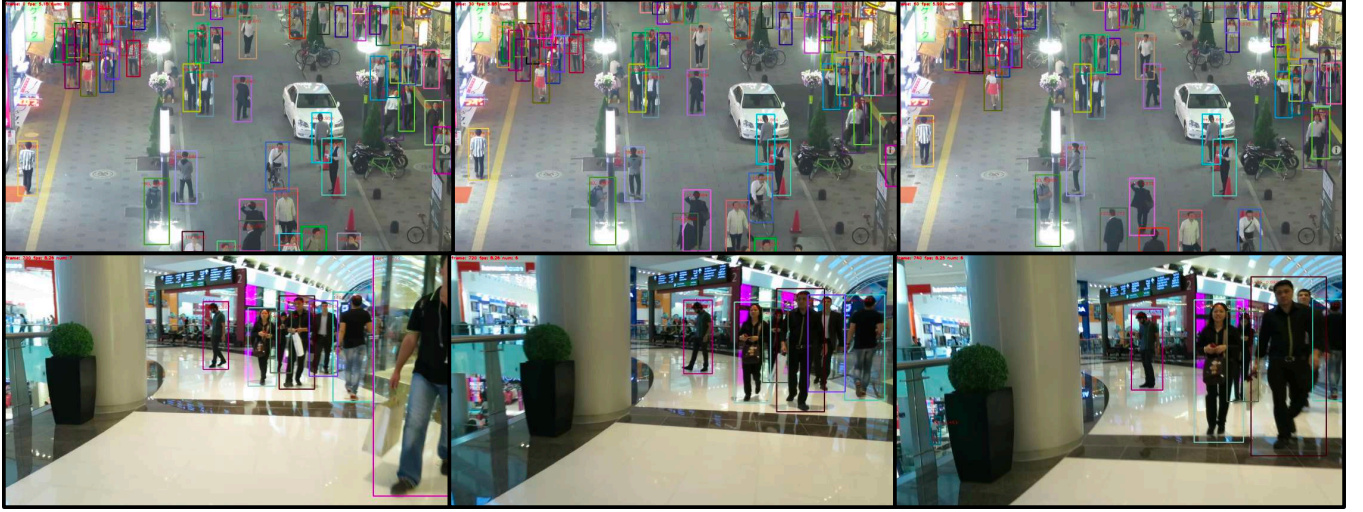


Fig. 3. Visualization of our detection and tracking results on two sequences of the MOT17 test set.

TABLE III
ABLATION STUDY ON EFFECTIVENESS OF GNNs.

Method	MOTA(%) \uparrow	IDF1(%) \uparrow	MT(%) \uparrow	ML(%) \downarrow	FP \downarrow	FN \downarrow	IDS \downarrow
No GNNs	79.5	73.8	71.8	8.8	1,948	1,487	139
1-layer GNNs	81.6	77.4	66.3	13.5	1,394	1,733	68
2-layer GNNs	81.9	75.0	65.7	13.5	1,234	1,866	54
3-layer GNNs	82.1	76.4	65.4	12.6	1,281	1,793	71

of our method in Fig. 3

Evaluating Object Detection. We compare our method with published methods in Table II. We observed that our detector outperforms prior methods in most of the metrics. We highlight that we achieve a higher Average Precision than most of the prior methods, which could be attributed to our use of GNNs that makes it easier to find existing objects and harder to generate false positives by exploiting object relation information.

V. ABLATION STUDY

We conduct ablation study on the 2DMOT2015 validation set, removing sequences overlapping with our training set.

Effect of GNNs. To answer the question on whether adding a GNN module is useful, we train our model with {1,2,3}-layer/no GNNs. The results are shown in Table III. When comparing 1-layer/no GNNs model, we see clear improvements in primary metrics MOTA and IDF1, which shows the usefulness of GNNs in the task of MOT. Also, with GNNs, we see a clear decrease in IDS comparing to the no-GNN model, which suggests that GNNs help learn more discriminative features and reduce confusion in data association. Moreover, when comparing {1,2,3}-layer GNNs model, we see further improvements in MOTA but lower numbers in IDF1, especially 2-layer model. We hypothesize that adding more layers might have made our model emphasize more on detection but less on temporally-consistent tracking. How to balance the task of detection and data association during the joint MOT learning can be a very interesting research topic in the future. Note that we use AGNNConv [59] in

this ablation study and we do not explore more than 3 GNN layers as GPU memory is not enough to fit our model.

VI. CONCLUSIONS

We proposed a new instance of joint MOT approach that simultaneously optimizes object detection and data association modules. Moreover, to model spatial-temporal object relations, we used GNNs to learn more discriminative features that benefit both detection and data association. Through extensive experiments, we showed effectiveness of GNNs under the joint MOT framework and achieved state-of-the-art performance on the MOT challenges. Our code will be released so that future follow-up work can easily build on top of our method to advance the state-of-the-art.

REFERENCES

- [1] R. Girshick, "Fast R-CNN," *ICCV*, 2015.
- [2] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *NIPS*, 2015.
- [3] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár, "A MultiPath Network for Object Detection," *BMVC*, 2016.
- [4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *CVPR*, 2016.
- [5] T.-y. Lin, P. Goyal, R. Girshick, K. He, and P. Doll, "Focal Loss for Dense Object Detection," *ICCV*, 2017.
- [6] B. Singh, H. Li, A. Sharma, and L. S. Davis, "R-FCN-3000 at 30fps: Decoupling Detection and Classification," *CVPR*, 2018.
- [7] X. Weng and K. Kitani, "Monocular 3D Object Detection with Pseudo-LiDAR Point Cloud," *ICCVW*, 2019.
- [8] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding Box Regression with Uncertainty for Accurate Object Detection," *CVPR*, 2019.
- [9] Y. Xiang, A. Alahi, and S. Savarese, "Learning to Track: Online Multi-Object Tracking by Decision Making," *ICCV*, 2015.
- [10] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," *ICIP*, 2016.
- [11] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," *ICIP*, 2017.
- [12] S. Schuster, P. Vernaza, W. Choi, and M. Chandraker, "Deep Network Flow for Multi-Object Tracking," *CVPR*, 2017.
- [13] A. Maksai, X. Wang, F. Fleuret, and P. Fua, "Non-Markovian Globally Consistent Multi-Object Tracking," *ICCV*, 2017.
- [14] X. Weng, J. Wang, D. Held, and K. Kitani, "3D Multi-Object Tracking: A Baseline and New Evaluation Metrics," *IROS*, 2020.

- [15] H. Karunasekera, H. Wang, and H. Zhang, "Multiple Object Tracking with Attention to Appearance, Structure, Motion and Size," *IEEE Access*, 2019.
- [16] C. Wang, Y. Wang, Y. Wang, C.-t. Wu, and G. Yu, "muSSP: Efficient Min-cost Flow Algorithm for Multi-object Tracking," *NeurIPS*, 2019.
- [17] W. Luo, B. Yang, and R. Urtaşun, "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net," *CVPR*, 2018.
- [18] S. Wang, D. Jia, and X. Weng, "Deep Reinforcement Learning for Autonomous Driving," *arXiv:1811.11329*, 2018.
- [19] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A Survey of Autonomous Driving: Common Practices and Emerging Technologies," *arXiv:1906.05113*, 2019.
- [20] C. Badue, R. Guidolini, R. V. Carneiro, P. Azevedo, V. B. Cardoso, A. Forechi, L. Jesus, R. Berriel, T. Paixão, F. Mutz, L. Veronese, T. Oliveira-Santos, and A. F. De Souza, "Self-Driving Cars: A Survey," *arXiv:1901.04407*, 2019.
- [21] X. Weng, J. Wang, S. Levine, K. Kitani, and R. Nick, "Inverting the Forecasting Pipeline with SPF2: Sequential Pointcloud Forecasting for Sequential Pose Forecasting," *CoRL*, 2020.
- [22] H.-u. K. B. and C.-s. Kim, "CDT: Cooperative Detection and Tracking for Tracing Multiple Objects in Video Sequences," *ECCV*, 2016.
- [23] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Detect to Track and Track to Detect," *ICCV*, 2017.
- [24] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without Bells and Whistles," *ICCV*, 2019.
- [25] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking Objects as Points," *ECCV*, 2020.
- [26] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe, "MOTS: Multi-Object Tracking and Segmentation," *CVPR*, 2019.
- [27] Z. Wang, L. Zheng, Y. Liu, Y. Li, and S. Wang, "Towards Real-Time Multi-Object Tracking," *ECCV*, 2020.
- [28] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "FairMOT: On the Fairness of Detection and Re-Identification in Multiple Object Tracking," *arXiv:2004.01888*, 2020.
- [29] J. Peng, C. Wang, F. Wan, Y. Wu, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, and Y. Fu, "Chained-Tracker: Chaining Paired Attentive Regression Results for End-to-End Joint Multiple-Object Detection and Tracking," *ECCV*, 2020.
- [30] B. Pang, Y. Li, Y. Zhang, M. Li, and C. Lu, "TubeTK: Adopting Tubes to Track Multi-Object in a One-Step Training Model," *CVPR*, 2020.
- [31] S. Sun, N. Akhtar, X. Song, H. Song, A. Mian, and M. Shah, "Simultaneous Detection and Tracking with Motion Modelling for Multiple Object Tracking," *ECCV*, 2020.
- [32] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang, "Joint Detection and Identification Feature Learning for Person Search," *CVPR*, 2017.
- [33] X. Weng, Y. Wang, Y. Man, and K. Kitani, "GNN3DMOT: Graph Neural Network for 3D Multi-Object Tracking with 2D-3D Multi-Feature Learning," *CVPR*, 2020.
- [34] G. Brasó and L. Leal-Taixé, "Learning a Neural Solver for Multiple Object Tracking," *CVPR*, 2020.
- [35] J. Li, X. Gao, and T. Jiang, "Graph Networks for Multiple Object Tracking," *WACV*, 2020.
- [36] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," *ECCV*, 2014.
- [37] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," *ECCV*, 2016.
- [38] K. He, G. Gkioxari, P. Doll, and R. Girshick, "Mask R-CNN," *ICCV*, 2017.
- [39] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as Points," *arXiv:1904.07850*, 2019.
- [40] H. Law and J. Deng, "CornerNet: Detecting Objects as Paired Key-points," *ECCV*, 2018.
- [41] X. Zhu, J. Dai, L. Yuan, and Y. Wei, "Towards High Performance Video Object Detection," *CVPR*, 2018.
- [42] M. Ramzy, H. Rashed, A. E. Sallab, and S. Yogamani, "RST-MODNet: Real-time Spatio-temporal Moving Object Detection for Autonomous Driving," *NeurIPS*, 2019.
- [43] F. Xiao and Y. J. Lee, "Video Object Detection with an Aligned Spatial-Temporal Memory," *ECCV*, 2018.
- [44] J. Zhang, S. Zhou, J. Wang, and D. Huang, "Frame-Wise Motion and Appearance for Real-time Multiple Object Tracking," *BMVC*, 2019.
- [45] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M. H. Yang, "Online Multi-Object Tracking with Dual Matching Attention Networks," *ECCV*, 2018.
- [46] A. Milan, S. Roth, and K. Schindler, "Continuous Energy Minimization for Multitarget Tracking," *TPAMI*, 2014.
- [47] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-Optimal Greedy Algorithms for Tracking a Variable Number of Objects," *CVPR*, 2011.
- [48] S. T. B. B. Andres, M. Andriluka, and B. Schiele, "Multi-Person Tracking by Multicut and Deep Matching," *ECCVW*, 2016.
- [49] W. Brendel, M. Amer, and S. Todorovic, "Multiobject Tracking as Maximum Weight Independent Set," *CVPR*, 2011.
- [50] A. Dehghan, S. M. Assari, and M. Shah, "GMMCP Tracker: Globally Optimal Generalized Maximum Multi Clique Problem for Multiple Object Tracking," *CVPR*, 2015.
- [51] A. Roshan Zamir, A. Dehghan, and M. Shah, "GMCP-Tracker: Global Multi-Object Tracking Using Generalized Minimum Clique Graphs," *ECCV*, 2012.
- [52] M. Gori, G. Monfardini, and F. Scarselli, "A New Model for Learning in Graph Domains," *IJCNN*, 2005.
- [53] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, "Weisfeiler and Leman Go Neural: Higher-Order Graph Neural Networks," *AAAI*, 2019.
- [54] T. N. Kipf and M. Welling, "Semi-Supervised Classification with Graph Convolutional Networks," *ICLR*, 2017.
- [55] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," *ICLR*, 2018.
- [56] X. Weng, Y. Yuan, and K. Kitani, "PTP: Parallelized Tracking and Prediction with Graph Neural Networks and Diversity Sampling," *Robotics and Automation Letters*, 2021.
- [57] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep Layer Aggregation," *CVPR*, 2018.
- [58] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint Triplets for Object Detection," *ICCV*, 2019.
- [59] K. K. Thekumprampal, C. Wang, S. Oh, and L.-J. Li, "Attention-based Graph Neural Network for Semi-supervised Learning," *1803.03735*, 2018.
- [60] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic Graph CNN for Learning on Point Clouds," *ACM Transactions on Graphics*, 2019.
- [61] R. Cipolla, Y. Gal, and A. Kendall, "Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics," *CVPR*, 2018.
- [62] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," *Naval Research Logistics Quarterly*, 1955.
- [63] R. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, 1960.
- [64] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking," *TPAMI*, 2015.
- [65] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: A Benchmark for Multi-Object Tracking," *TPAMI*, 2016.
- [66] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, "CVPR19 Tracking and Detection Challenge: How crowded can it get?" *TPAMI*, 2019.
- [67] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé, "MOT20: A benchmark for multi object tracking in crowded scenes," *arXiv:2003.09003*, 2020.
- [68] K. Bernardin and R. Stiefelhagen, "Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics," *Journal on Image and Video Processing*, 2008.
- [69] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara, and C. Tomasi, "Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking," *ECCVW*, 2016.
- [70] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: A benchmark," *CVPR*, 2009.
- [71] A. Ess, B. Leibe, K. Schindler, and L. van Gool, "A Mobile Vision System for Robust Multi-Person Tracking," *CVPR*, 2008.
- [72] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian, "Person Re-identification in the Wild," *CVPR*, 2017.
- [73] S. Zhang, R. Benenson, and B. Schiele, "CityPersons: A Diverse Dataset for Pedestrian Detection," *CVPR*, 2017.
- [74] S. Bae and K. Yoon, "Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking," *TPAMI*, 2018.

- [75] A. Hornakova, R. Henschel, B. Rosenhahn, and P. Swoboda, "Lifted Disjoint Paths with Application in Multiple Object Tracking," *ICML*, 2020.
- [76] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, "Online Multi-Target Tracking with Strong and Weak Detections," *ECCVW*, 2016.
- [77] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai, "Online multi-object tracking with convolutional neural networks," *ICIP*, 2017.
- [78] W. Choi, "Near-Online Multi-Target Tracking with Aggregated Local Flow Descriptor," *ICCV*, 2015.
- [79] K. Fang, Y. Xiang, X. Li, and S. Savarese, "Recurrent Autoregressive Networks for Online Multi-Object Tracking," *WACV*, 2018.
- [80] X. Wan, J. Wang, Z. Kong, Q. Zhao, and S. Deng, "Multi-Object Tracking Using Online Metric Learning with Long Short-Term Memory," *ICIP*, 2018.
- [81] Z. Zhou, J. Xing, M. Zhang, and W. Hu, "Online Multi-Target Tracking with Tensor-Based High-Order Graph Matching," *ICPR*, 2018.
- [82] N. Mahmoudi, S. M. Ahadi, and Mohammad, "Multi-Target Tracking Using CNN-Based Features: CNNMTT," *Multimedia Tools and Applications*, 2019.
- [83] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "POI: Multiple Object Tracking with High Performance Detection and Appearance Feature," *ECCVW*, 2016.
- [84] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part Based Models," *TPAMI*, 2009.
- [85] C. Lin, J. Lu, G. Wang, and J. Zhou, "Graininess-Aware Deep Feature Learning for Pedestrian Detection," *ECCV*, 2018.
- [86] F. Yang, W. Choi, and Y. Lin, "Exploit All the Layers: Fast and Accurate CNN Object Detector with Scale Dependent Pooling and Cascaded Rejection Classifiers," *CVPR*, 2016.
- [87] L. Ciampi, N. Messina, F. Falchi, C. Gennaro, and G. Amato, "Virtual to Real Adaptation of Pedestrian Detectors," *Sensors*, 2020.