Detection Recovery in Online Multi-Object Tracking with Sparse Graph Tracker

Jeongseok Hyun[1]* Myunggu Kang[2] Dongyoon Wee[2] Dit-Yan Yeung[1]
[1]The Hong Kong University of Science and Technology [2]Clova AI, NAVER Corp.

Abstract

In existing joint detection and tracking methods, pairwise relational features are used to match previous tracklets to current detections. However, the features may not be discriminative enough for a tracker to identify a target from a large number of detections. Selecting only high-scored detections for tracking may lead to missed detections whose confidence score is low. Consequently, in the online setting, this results in disconnections of tracklets which cannot be recovered. In this regard, we present Sparse Graph Tracker (SGT), a novel online graph tracker using higher-order relational features which are more discriminative by aggregating the features of neighboring detections and their relations. SGT converts video data into a graph where detections, their connections, and the relational features of two connected nodes are represented by nodes, edges, and edge features, respectively. The strong edge features allow SGT to track targets with tracking candidates selected by top-K scored detections with large K. As a result, even low-scored detections can be tracked, and the missed detections are also recovered. The robustness of K value is shown through the extensive experiments. In the MOT16/17/20 and HiEve Challenge, SGT outperforms the state-of-the-art trackers with real-time inference speed. Especially, a large improvement in MOTA is shown in the MOT20 and HiEve Challenge. Code is available at https://github.com/HYUNJS/SGT.

1. Introduction

In the online setting, missed detection problem is far more critical than in the offline setting; tracklets are disconnected once the corresponding detections are missed, while tracklet interpolation is infeasible to fill the past missed detections. As illustrated in Figure 1, occlusion leads to low-confident detections, and if they are included in the association step, the complexity of tracking increases with too many spurious detections. Pairwise relational features (*e.g.*, position or visual similarity) may not be discriminative enough to distinguish targets in such case and result in wrong matching. Thus, existing works [45, 50, 19, 35], which use the pairwise relations for tracking, exploit only high-scored detections as tracking candidates.

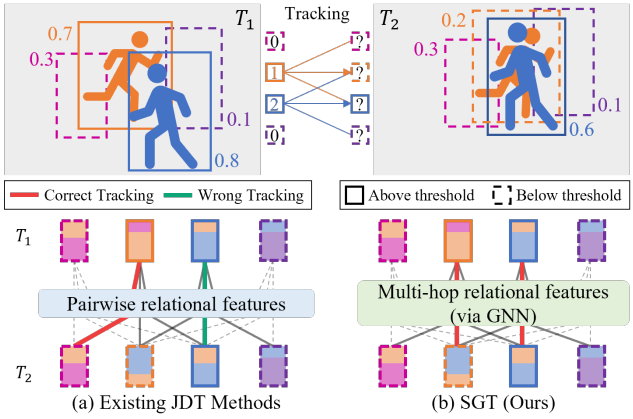

Figure 1. **Motivation of detection recovery by tracking.** Tracking of 1 cannot be performed if the high-scored detection (?) is selected as the tracking candidate. Meanwhile, adding low-scored detections (?, ?, ?) to the candidate pool results in wrong matching of 1 with ? in **(a)** since pairwise relations are ambiguous to be used for discriminating 1 among them. In contrast, our SGT **(b)** exploits multi-hop relations updated via a GNN to contain visual features of neighboring detections and their relations. Despite many tracking candidates selected by top-K, SGT successfully tracks 1, and its missed detection is consequently recovered.

A graph is an effective way to represent relations between objects in a video, and a graph neural network (GNN) is effective in modeling the relationship. Bearing this in mind, we model the spatio-temporal relationship in video data using a GNN to extract higher-order relational features (*i.e.*, multi-hop relational features) which consider the relations between neighboring objects or background patches. These features are powerful and can perform association correctly even if a large number of detections (*e.g.*, 300) is selected as tracking candidates. Thus, we propose Sparse Graph Tracker (SGT), a novel online graph tracker that adopts joint detection and tracking (JDT) framework [35] where object detector and tracker share a backbone network to achieve fast inference speed.

In the association step, existing online JDT methods uti-

*The work was partially done during an intern at Clova AI.

lize pairwise relational features between two detections, such as similarity of appearance features [23, 35, 45, 34], center point distance [48], and Intersection over Union (IoU) score [32]. Although [37, 35, 45, 34] fuse appearance information and motion information by weighted sum, these features reflect only the relations between two objects and are not discriminative for accurate matching in a crowded scene. Motion predictor (*e.g.*, Kalman filter [3]) is commonly employed to improve tracking performance. On the contrary, we utilize higher-order relational features by aggregating the features of neighboring nodes and edges through iterations of GNNs. Even without a motion predictor, higher-order relational features are still powerful to correctly match the previous tracklets with current frame's (I_{t2}) top-K scored detections which contain a large number of spurious detections due to a large K value.

The main contributions of this work are as follows:

1. We propose a novel online graph-based tracker that is jointly trained with object detector and performs long-term association without any motion model. Our SGT shows superior performance on the MOT16/17/20 and HiEve benchmarks with real-time inference speed.
2. We propose training and inference techniques for SGT effectively achieving detection recovery by tracking. Their effectiveness is demonstrated through extensive ablation experiments and a large improvement on MOT20 where severe crowdedness results in low confident detections for the occluded objects.

2. Related Works

2.1. JDT Methods

Recently, many JDT methods are proposed due to fast inference speed and single-stage training based on the shared backbone. They extend object detectors to MOT models with the extra tracking branch which is jointly trained with the detector. They fall into two categories: (1) re-identification (reID) branch outputs discriminative appearance features for tracking, and (2) motion prediction branch outputs the updated location of tracklets for tracking.

JDT by reID. RetinaTrack [23], JDE [35], FairMOT [45] append reID branch to RetinaNet [20], YOLOv3 [28], and CenterNet [49], respectively. Liang *et al*. [19] points out that the objectives of detection and ReID are conflicting, and proposes a cross-correlation network that learns task-specific features. On the other hand, GSDT [34] and CorrTracker [33] enhance the current features by spatio-temporal relational modeling that exploits previous frames. CorrTracker [33], the current state-of-ther-art model, fuses the correlation in both spatial and temporal dimensions to the image features at multiple pyramid levels. All these methods associate the tracklets and detections using the similarity of reID features. Also, Kalman filter [3] is commonly employed and motion information is fused to the similarity.

JDT by motion prediction. D&T [11] and CenterTrack [48] append the learnable motion predictor into R-FCN [6] and CenterNet [49], respectively. CenterTrack associates the tracklets and detections using the center point distance of the detections and the tracklets updated by predicted motion. TraDeS [38] predicts the center offset of objects between two consecutive frames based on a cost volume which is computed by a similarity of reID features of the frames. TransTrack [32] is a transformer-based tracker that propagates the previous frame's tracklets to the current frame and matches with the current detections by IoU score.

Comparison. Our SGT extends CenterNet [49] with a graph tracker. Compared with others using pairwise relational features (*e.g.*, IoU, cosine similarity or fusion of them), SGT exploits edge features updated through GNN which are higher-order relational features, and solves association as edge classification as shown in Figure 1.

2.2. Graph-based Multi-object Tracking

A graph is an effective way to represent relational information, and GNN can learn higher-order relational information through a message passing process that propagates node or edge features to the connected nodes or edges and aggregates neighboring features. STRN [40] is an online MOT method with a spatio-temporal relation network that consists of a spatial relation module and a temporal relation module. The features from these two modules are fused to predict the affinity score for association. MPNTrack [5] adopts a message passing network [12] with time-aware node update module that aggregates past and future features separately and solves MOT problem as edge classification. LPCMOT [7] generates and scores tracklet proposals based on a set of frames and detections with Graph Convolution Network (GCN) [14]. GSDT [34] is the first work that applies a GNN in an online JDT method, but its use of GNN is limited to enhancing the current feature map and tracking is still performed using pairwise relational features. In contrast, SGT is the first JDT method using higher-order relational features for tracking.

2.3. Online Detection Recovery

In the TBD framework, detections to be tracked are decided based on certain threshold. Two detection threshold values are commonly used in online MOT methods [35, 45, 19]: τ_{init} and τ_D for initializing unmatched detections as new tracklets and choosing tracking candidates, respectively. Due to a high value of τ_D (*e.g.*, 0.4), some low-scored true detections are not included in the tracking candidates. In ByteTrack [44], an extra association stage is deployed that the unmatched tracklets are matched with low-

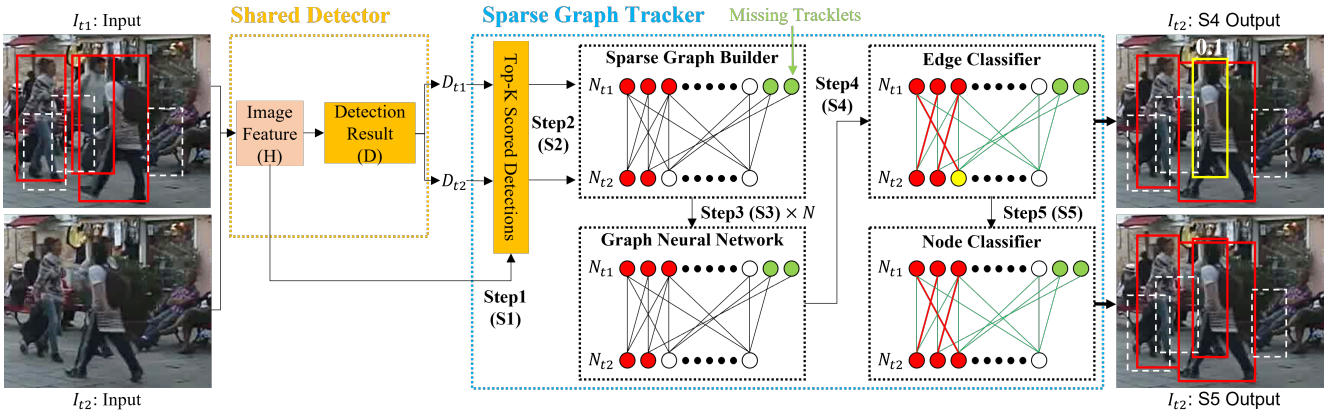

Figure 2. Overview of detection recovery in the inference pipeline of SGT. *(S1)* Top-K scored detections and their features are extracted from I_{t1} and I_{t2}. The red boxes indicate positive detection that contains ID. *(S2)* A sparse graph is built, where a node, $n_T^i \in N_T$ ($i \in [1, K]$), is a detection of frame $T = \{t1, t2\}$ and an edge ($e_{i,j}$) is a connection between n_{t1}^i and n_{t2}^j. The green nodes are the tracklets that are missed until $t1$ and they are appended to N_{t1}. The red nodes indicate the positive detection that has its assigned ID. Two nodes at N_{t2} are red since their detection scores are above τ_{init}, so new ID can be assigned to them. *(S3)* GNN updates the features of nodes and edges to become higher-order by aggregating neighboring features. *(S4)* The edge score of the red line (a positive edge) is above the edge threshold (τ_E) while the green line represents a negative edge. The yellow node (n_{t2}^3) is an example of detection recovery. It was previously negative detection due to its low score, but it becomes a positive detection with the help of a positive edge. *(S5)* The recovered detection (n_{t2}^3) in S4 is verified by the node score. If the node score is below the node threshold (τ_N), it is regarded as a false positive and is filtered out. Otherwise, the node is recovered and hence can be successfully detected which is shown by the yellow node becoming red.

scored detections using IoU score. However, a new detection threshold, $\tau_{D_{low}}$ (*e.g.*, 0.2), is introduced for selecting the low-scored detections as the candidates, and this is a critical value deciding the trade-off between FP and FN. OMC [18] introduces an extra stage before the association step to complement missed detections which may not be detected due to the low confidence score.

3. Sparse Graph Tracker

3.1. Overall Architecture

Figure 2 shows the architecture of SGT. While various image backbones and object detectors can be flexibly adopted to SGT, our main experiment is based on CenterNet [49] with a variant of DLA-34 backbone [42] as same as our baseline, FairMOT [45]. Following [45], we modify CenterNet that the box size predictor outputs left, right, top, and bottom sizes (s_l, s_r, s_t, s_b) from a center point of an object instead of the width and height. CenterNet is a point-based detector that predicts object at every pixel of a feature map. The score head's output is denoted as $B_{score} \in \mathbb{R}^{H_h \times H_w \times 1}$, where H_h and H_w are height and width of the feature map. The output from the size head is denoted as $B_{size} \in \mathbb{R}^{H_h \times H_w \times 4}$. The offset head adjusts the center coordinates of objects using $B_{off} \in \mathbb{R}^{H_h \times H_w \times 2}$. At frame T, CenterNet outputs detections $D_T = (S_T, B_T)$, where S_T is the detection score (B_{score}) and $B_T \in \mathbb{R}^{H_h \times H_w \times 4}$ is top-left and bottom-right coordinates.

Sparse graph builder takes top-K scored detections from each frame (I_{t1} and I_{t2}) and sets them as the nodes of a graph (N_{t1} and N_{t2}). In the inference phase, the previous timestep's N_{t2} will be the current timestep's N_{t1}. We sparsely connect N_{t1} and N_{t2} only if they are close in either Euclidean or feature space. Specifically, $n_{t1}^i \in N_{t1}$ is connected to N_{t2} with three criteria: 1) small distance between their center coordinates; 2) high cosine similarity between their features; 3) high IoU score. For each criterion, the given number of N_{t2} (*e.g.*, 10) are selected to be connected to n_{t1}^i without duplicates. The connection is bidirectional so that both N_{t1} and N_{t2} update their features. The visual features of the detections and relational features are used as the features of nodes (V) and edges (E), respectively.

To include low-scored detections for tracking, a low threshold value can be an alternative to top-K. Although it can also achieve good performance if it is small enough as shown in the supplementary material, such detection threshold value is sensitive to the detector's score distribution. As a result, careful calibration is required for different detectors and datasets. In contrast, top-K method is robust to such issues as it is not affected by the score distribution. Since K is the maximum number of objects that the model can track, we set K to be sufficiently larger than the maximum number of people in the dataset (*e.g.*, 100 in MOT16/17; 300 in MOT20). In Table 7, we experimentally show the robustness of K values.

Some tracklets are failed to track for a while when they are invisible due to full occlusion. These missing tracklets are stored for a period of age_{max} and appended to N_{t1}. Al-

though existing MOT works [4, 35, 45, 33] apply a motion predictor (*e.g.*, Kalman filter [3]) for predicting the possible location of missing tracklets, SGT can perform long-term association without a motion predictor. Here, we store the tracklets whose length is longer than age_{min} to prevent false positive cases.

Graph neural network updates features of nodes (V) and edges (E) in a graph through the message passing process, as described in Figure 3, that propagates the features to the neighboring nodes and edges and then aggregates them. By iterating this process, V now contains the features of both the neighboring nodes and edges, and E indirectly aggregates the features of other edges which are connected to the same node. While the initial edge features represent the pairwise relation of two detections, the iteration of the process allows the updated edge features to represent the higher-order (multi-hop) relation that also considers neighboring detections. Section 3.2 presents more details.

Edge classifier is a FC layer that predicts the edge score (ES) from the updated edge features. The edge score is the probability that the connected detections at $t1$ and $t2$ refer to the same object. Since n^i_{t1} is connected to many nodes at $t2$, we use the Hungarian algorithm [15] for optimal matching based on the edge score matrix. As a result, n^i_{t1} has only one edge score which is optimally assigned. Then, the edge threshold (τ_E) is used for deciding a positive or negative edge. The yellow box shown in Figure 2 is the recovered detection that n^3_{t2} is negative due to its low detection score, but its connected node, n^1_{t1}, and edge ($e_{1,3}$) are positive.

Node classifier is a FC layer that prevents incorrect detection recovery by predicting the node score (NS) from the updated node features. If the recovered detection's node score is below the node threshold (τ_N), we decide not to recover it, thus the node stays negative. Otherwise, we confirm recovery of the missed detection and the node becomes positive as shown by n^3_{t2} in Figure 2.

3.2. Graph Construction and Update

This section explains design of the node and edge features in SGT. Note that a FC block refers to a stack of FC layer, layer normalization [1], and ReLU function

Initial node features. Contrary to the graph-based MOT works using reID features of detected objects [5, 40, 36], SGT exploits the image backbone's visual features (H) which are shared for detection and jointly trained.

Initial edge features. Edge feature are denoted as $e^l_{i,j}$, where i and j are the starting and ending node indices respectively, and l indicates iteration. Inspired by MPNTrack [5], SGT initializes high-dimensional edge features as Eq. 1.

$$e^0_{i,j} = f_{enc}\left(\left[x_i - x_j,\, y_i - y_j,\, \log(\tfrac{w_i}{w_j}),\, \log(\tfrac{h_i}{h_j}),\, IoU_{i,j},\, Sim_{i,j}\right]\right), \quad (1)$$

where $[\cdot]$ is concatenation operator, x and y are the center coordinates, h and w are the height and width of a bounding box, Sim is cosine similarity, and f_{enc} refers to two FC blocks. As the initialized edge features are direction-aware, two edges connecting the same nodes but reversely will have different features considering different relations (*e.g.*, $t1 \rightarrow t2$ and $t2 \rightarrow t1$). V_{t1} and V_{t2} are updated on two different MLPs with these different edge features. After updating in GNN, these bidirectional edge features are averaged to predict a single edge score.

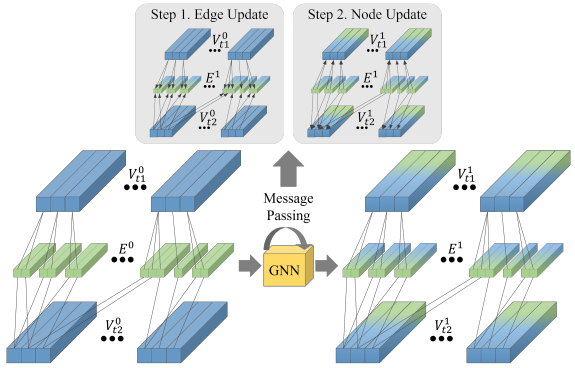

Figure 3. Illustration of message passing in a GNN. Initial edge features, E^0, are updated to E^1 containing the features of two connected nodes, V^0_{t1} and V^0_{t2}. Initial node features, V^0, are then updated to V^1 containing the features of connected nodes, V^0, and the updated edges features, E^1. For simplicity, we omit bidirectional connections and show only few edges.

Initial graph, shown in the left of Figure 3, is denoted by $G^0 = \{V^0, E^0\}$, where $E^0 = \{e^0_{i,j} | 1 \leq i,j \leq 2K + |V_{miss}|\}$ is a set of initial edge features and $V^0 = V^0_{t1} \bigcup V^0_{t2} \bigcup V^0_{miss}$ is a set of initial node features at $t1$, $t2$, and missing tracklets.

Update in node and edge features. Figure 3 describes two steps to update the features of nodes and edges during the message passing process in GNN. The initial edge features $e^0_{i,j}$, shown in the left side of the graph, are pairwise relational features considering only the two connected nodes at $t1$ and $t2$ (direction $i \rightarrow j$).

In Step 1 of Figure 3, the edge features are updated as Eq. 2.

$$e^l_{i,j} = f_e\left(\left[v^{l-1}_i,\, v^{l-1}_j,\, e^0_{i,j},\, e^{l-1}_{i,j}\right]\right), \quad (2)$$

where f_e refers to two FC blocks, l is the number of iterations ($l \in [1, N_{iter}]$), v_i is the features of node i, and v^{l-1}_i indicates the node features of the previous iteration. Therefore, the current state of the two connected nodes, initial and current edge features are concatenated and passed to f_e to update edge features as $e^l_{i,j}$. Initial edge features ($e^0_{i,j}$) are concatenated every iteration to prevent the over-smoothing issue in GNN [25]. Although we use the shared MLPs (f_e) for the edges of two different directions, the edge features

of the opposite direction may not be the same since their edge features are encoded in a direction-aware manner.

In Step 2 of Figure 3, node j aggregates the features of the connected nodes and edges as Eq. 3.

$$v_j^l = f_{v_{out}}\left(\frac{1}{|E_{:,j}^l|}\sum_i f_{v_{enc}}\left(\left[v_i^{l-1}, e_{i,j}^l\right]\right)\right), \quad (3)$$

where $f_{v_{out}}$ is an FC block, $|E_{:,j}^l|$ is the number of edges connected to the node j, $f_{v_{enc}}$ refers to two FC blocks, $e_{i\,j}^l$ is the updated edge features in Step 1 (Eq. 2) and v_i^{l-1} is the features of starting node. We suppose the index of N_{t2} is from 1 to K and N_{t1} is from $K+1$ to $2K+|V_{miss}|$. When $i > j$, $e_{i\,j}$ is the edge features with direction of $t1 \rightarrow t2$. Thus, message passing is from $t1$ to $t2$ and V_{t2} are updated. As our edge features are direction-aware, we use different $f_{v_{enc}}$ for message passing $t1 \rightarrow t2$ and $t2 \rightarrow t1$.

3.3. Training and Inference Techniques

SGT is trained by the sum of the detection loss ($\mathcal{L}_D$) and the association loss ($\mathcal{L}_A$).

Detection loss. Since we adopt CenterNet [49] as a detector, we follow [49] to compute the detection loss which is the weighted sum of losses from three heads as Eq. 4.

$$\mathcal{L}_D = \mathcal{L}_{score} + w_{size}\mathcal{L}_{size} + w_{off}\mathcal{L}_{off} \quad (4)$$

The size head outputs B_{size} composed of (s_l, s_r, s_t, s_b). The offset head outputs B_{off} which is the quantization error of the center coordinates caused by the stride of feature map (*e.g.*, 4). For each ground-truth (GT) object $\hat{b}^i = (\hat{x}_l^i, \hat{y}_t^i, \hat{x}_r^i, \hat{y}_b^i)$, GT size $\hat{b}_{size}^i = (\hat{s}_l^i, \hat{s}_r^i, \hat{s}_t^i, \hat{s}_b^i)$ is computed by the difference between center coordinates $(\hat{c}_x^i, \hat{c}_y^i) = (\frac{\hat{x}_l^i+\hat{x}_r^i}{2}, \frac{\hat{y}_t^i+\hat{y}_b^i}{2})$ and $\hat{b}^i$. Each GT size $\hat{b}_{size}^i$ is assigned to the prediction $b_{size}^{xy} \in B_{size}$, where $(x, y) = (\lfloor\frac{\hat{c}_x^i}{4}\rfloor, \lfloor\frac{\hat{c}_y^i}{4}\rfloor)$. Each GT offset $(\hat{o}_x, \hat{o}_y) = (\frac{\hat{c}_x^i}{4} - \lfloor\frac{\hat{c}_x^i}{4}\rfloor, \frac{\hat{c}_y^i}{4} - \lfloor\frac{\hat{c}_y^i}{4}\rfloor)$ is assigned to the prediction b_{off}^{xy}. Then, l_1 loss is used to compute $\mathcal{L}_{size}$ and $\mathcal{L}_{off}$. For training the score head, GT heatmap $M^{xy} \in \mathbb{R}^{H_h \times H_w \times 1}$ is generated by the Gaussian kernel as Eq. 5.

$$M^{xy} = \sum_{i=1}^{N_D} \exp\left(-\frac{(x-\lfloor\frac{\hat{c}_x^i}{4}\rfloor)^2)+(y-\lfloor\frac{\hat{c}_y^i}{4}\rfloor)^2)}{2\sigma_d^2}\right), \quad (5)$$

where N_D is the number of GT object and σ_d is computed by width and height of each object [16]. $\mathcal{L}_{score}$ is computed as the pixel-wise logistic regression with the penalty-reduced focal loss [20].

Association loss. Our association loss is the weighted sum of the edge and node classification losses as Eq. 6.

$$\mathcal{L}_A = w_{edge}\mathcal{L}_{edge} + w_{node}\mathcal{L}_{node} \quad (6)$$

In SGT, the edge and node classifiers output the edge and node scores (ES and NS), respectively. $\mathcal{L}_{edge}$ and $\mathcal{L}_{node}$ are computed on these scores with the focal loss [20]. Since it is difficult to assign GT labels to the edges connecting the background patches, we exclude them in $\mathcal{L}_{edge}$ as Eq. 7.

$$\mathcal{L}_{edge} = \frac{1}{N_{E^+}}\sum_{e_{i,j}\in E}\begin{cases}\text{FL}(ES_{i,j}, ey_{i,j}), & \text{if } ny_i = 1 \text{ or } ny_j = 1;\\ 0 & \text{otherwise,}\end{cases} \quad (7)$$

where N_{E^+} is the number of GT edges which at least one of the endpoints is positive, E is a set of edges in G, FL is the focal loss, edge in direction of $t1 \rightarrow t2$, $ey_{i,j}$ is the GT label of edge connecting the nodes n_i and n_j, and ny_i is the GT label of n_i. We compute $\mathcal{L}_{node}$ only on the node scores at $t2$ as Eq. 8.

$$\mathcal{L}_{node} = \frac{1}{N_{N_{t2}^+}}\sum_{n_j\in N_{t2}} \text{FL}(NS_j, ny_j), \quad (8)$$

where $N_{N_{t2}^+}$ is the number of GT positive nodes at $t2$. We output zero when $N_{E^+} = 0$ or $N_{N_{t2}^+} = 0$.

Node and edge label assignment is an essential step for computing the association loss. While existing GNN-based tracker [5] trains its matching network using GT objects, we introduce a novel training technique using pseudo labels to effectively train the edge and node classifiers in a single step with a detector and a shared backbone network. The top-K detections are optimally matched with the GT objects based on their IoU score matrix and Hungarian algorithm [15], and IDs of the GTs are assigned to the matched detections. To prevent the misallocation of GT ID, the assigned IDs are filtered out if IoU of their matching is lower than the threshold (*e.g.*, 0.5). This step is repeated for N_{t1} and N_{t2} to assign (ny_i and ny_j). Finally, the GT edge label ($ey_{i,j}$) is assigned to the edges by matching the IDs of nodes. An edge is labeled as 1 if the two connected nodes have the same GT ID, and 0 otherwise.

Adaptive feature smoothing (AdapFS) is a novel inference technique for the proposed detection recovery framework. Following JDE [35], recent online TBD models update appearance features of tracklets in an exponential moving average manner as $emb_{t2}^{trk} = \alpha \times emb_{t1}^{trk} + (1-\alpha) \times emb_{t2}^{det}$. The features of tracklets are updated by adding the features of new detections with the fixed weight, α. However, low-scored recovered objects have unreliable appearance features since they may suffer from occlusion or blur. Thus, we incorporate adaptive weight computed by the object scores (S_T) of matched tracklets and detections as Eq. 9.

$$emb_{t2}^{trk} = emb_{t1}^{trk} \times \frac{S_{t1}}{S_{t1}+S_{t2}} + emb_{t2}^{det} \times \frac{S_{t2}}{S_{t1}+S_{t2}} \quad (9)$$

4. Experiments

4.1. Datasets and Implementation Details

Datasets. We train and evaluate the proposed method using MOT16/17/20 and HiEve Challenge datasets [24, 8, 22]

which target pedestrian tracking. MOT20 and HiEve are complex datasets composed of crowded scenes. On each frame, MOT20 has 170 people on average compared to MOT17 containing 30 people. Due to the small size of the MOT datasets, JDE [35] introduces the pedestrian detection and reID datasets [10, 43, 9, 39, 47] for training. FairMOT [45] further exploits extra pedestrian detection dataset, CrowdHuman [30]. We only use CrowdHuman as an extra training dataset to achieve competitive performance. Since CrowdHuman does not have ID labels and is not a video dataset, we assign a unique ID to every object and randomly warp an image to generate a pair of consecutive frames (I_{t1}–I_{t2}).

Implementation details. We use CenterNet [49] pretrained on COCO object detection dataset [21] to initialize SGT's detector. For fair comparison with [45, 33, 41, 34], we use the image size of 1088×608 and the feature map size ($H_w \times H_h$) of 272×152. Two consecutive frames are randomly sampled in the interval of $[1, 30]$. Following [45], random flip, warping and color jittering are selected as data augmentation. The same augmentation is applied to a pair of images. We use Adam optimizer [13] with a batch size of 12 and initial learning rate (lr) of $2e^{-4}$ which drops to $2e^{-5}$. There are 60 training epochs and lr is dropped at 50. For training, we use 1 for w_{off}, 0.1 for w_{size}, w_{edge}, and 10 for w_{node}. For inference, we use 0.5, 0.4 and 0.4 as τ_{init}, τ_E and τ_N, respectively. These values are chosen empirically.

4.2. MOT Challenge Evaluation Results

We submit our result to the MOT16/17/20 Challenge test server and compare it with the recent online MOT models as shown in Table 1. Note that the methods using tracklet interpolation as post-processing (*e.g.*, ByteTrack [44]) are excluded to satisfy *online setting*. Visualization results are provided in the supplementary material.

Evaluation metrics. We use the standard evaluation metrics for 2D MOT [2]: Multi-object Tracking Accuracy (MOTA), ID F1 Score (IDF1), False Negative (FN), False Positive (FP), and Identity Switch (IDS) [17]. While MOTA is computed by FP, FN, and IDS, and thus focuses on the detection performance, IDF1 [29] is a metric focused on tracking performance. Also, mostly tracked targets (MT) and mostly lost targets (ML) represent the ratio of GT trajectories covered by a track hypothesis for at least 80% and at most 20% of their respective life span, respectively.

Evaluation results of MOT16/17. Without extra training datasets, MOTA of SGT is higher than CSTrack [19] and FairMOT [45] by about 3%, and it is comparable with FairMOT trained with extra training datasets. With CrowdHuman as an extra training dataset, SGT achieves the highest MOTA on MOT16/17 based on the best trade-off between FP and FN. The highest MT indicates that SGT generates stable and long-lasting tracklets which is attributed to the proposed detection recovery mechanism. Compared with the models based on the same detector [48, 26, 38, 34, 45, 41, 33, 46, 33], SGT outperforms all of them, except CorrTracker [33] which shows marginally higher MOTA by 0.1% on MOT17. When we compare SGT with OMC–F [18] which is also specially designed to track low-scored detections on top of FairMOT [45], SGT shows lower FN with lower FP, demonstrating the superiority of our detection recovery mechanism over OMC [18]. SGT achieves excellent tracking performance on MOT20, while it shows high IDS on MOT17. This is caused by non-human occluders (*e.g.*, vehicles) frequently appearing in MOT17. Since pedestrian is the only target class for training the detector, other non-human objects are not included in top-K detections and for relational modeling as well. This leads to inferior tracking performance in MOT17.

Evaluation results of MOT20. In MOT20, scenes are

Table 1. Evaluation results of ours and recent online JDT models on the MOT16/17/20 benchmarks (private detection). OMC–F [18] applies its method on FairMOT [45]. For each metric, the best is bolded and the second best is underlined. The values not provided are filled by "-". † indicates no extra training dataset.

Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓
MOT16 [24]							
QDTrack [26] †	69.8	67.1	41.6	19.8	9861	44050	1097
TraDes [38]	70.1	64.7	37.3	20.0	**8091**	45210	1144
CSTrack [19] †	71.3	68.6	-	-	-	-	1356
SGT (Ours) †	74.1	71.0	43.6	15.8	9784	35946	1528
GSDT [34]	74.5	68.1	41.2	17.3	8913	36428	1229
FairMOT [45]	74.9	72.8	44.7	15.9	-	-	1074
CSTrack [19]	75.6	73.3	42.8	16.5	9646	33777	1121
RelationTrack [41]	75.6	**75.8**	43.1	21.5	9786	34214	**448**
OMC [18]	76.4	74.1	46.1	13.3	10821	31044	-
CorrTracker [33]	76.6	74.3	47.8	13.3	10860	30756	979
SGT (Ours)	**76.8**	73.5	**49.3**	**10.5**	10695	**30394**	1276
MOT17 [24]							
CTracker [27] †	66.6	57.4	32.2	24.2	22284	160491	5529
CenterTrack [48]	67.8	64.7	34.6	24.6	**18489**	160332	3039
QDTrack [26] †	68.7	66.3	40.6	21.9	26598	146643	3378
TraDes [38]	69.1	63.9	36.4	21.5	20892	150060	3555
FairMOT [45] †	69.8	69.9	-	-	-	-	3996
SOTMOT [46]	71.0	71.9	42.7	15.3	39537	118983	5184
GSDT [34]	73.2	66.5	41.7	17.5	26397	120666	3891
SGT (Ours) †	73.2	70.2	42.0	17.7	25332	121155	4809
FairMOT [45]	73.7	72.3	43.2	17.3	27507	117477	3303
RelationTrack [41]	73.8	**74.7**	41.7	23.2	27999	118623	**1374**
TransTrack [32]	74.5	63.9	46.8	**11.3**	28323	112137	3663
OMC–F [18]	74.7	73.8	44.3	15.4	30162	108556	-
CSTrack [19]	74.9	72.3	41.5	17.5	23847	114303	3567
OMC [18]	76.3	73.8	44.7	13.6	28894	101022	-
SGT (Ours)	76.4	72.8	**48.0**	11.7	25974	102885	4101
CorrTracker [33]	**76.5**	73.6	47.6	12.7	29808	**99510**	3369
MOT20 [8]							
FairMOT [45]	61.8	67.3	**68.8**	**7.6**	103440	**88901**	5243
TransTrack [32]	64.5	59.2	49.1	13.6	28566	151377	3565
SGT (Ours) †	64.5	62.7	62.7	10.2	67352	111201	4909
CorrTracker [33]	65.2	69.1	66.4	8.9	79429	95855	5183
CSTrack [19]	66.6	68.6	50.4	15.5	25404	144358	3196
GSDT [34]	67.1	67.5	53.1	13.2	31913	135409	3131
RelationTrack [41]	67.2	70.5	62.2	8.9	61134	104597	4243
SOTMOT [46]	68.6	**71.4**	64.9	9,7	57064	101154	4209
OMC [18]	70.7	67.8	56.6	13.3	**22689**	125039	-
SGT (Ours)	**72.8**	70.6	64.3	12.7	25161	112963	**2474**

severely crowded and partial occlusion is dominant compared with MOT16/17. When the same detection threshold (τ_D) is employed, existing methods suffer from missed detections caused by less confident detection output. [45, 33, 41] use lower detection threshold for MOT20, yet this results in high FP, IDS and low IDF1 since their pairwise relational features are not strong enough for correctly matching with a large number of tracking candidates. On the other hand, our higher-order relational features allows SGT to effectively address this problem and achieve state-of-the-art in MOTA as shown in Table 1. SGT surpasses CorrTracker [33] in MOTA by 7.6% while it shows higher MOTA than SGT in MOT17. SGT achieves better trade-off between FN and FP, and higher MOTA and IDF1 than OMC [18] whose detection recovery method is applied to CSTrack [19]. Although OMC exploits past frame as temporal cues to carefully select the low-scored detections, its matching is still limited by the pairwise relational features. In contrast, SGT performs matching using the higher-order relational features updated by GNNs, and thus, SGT outperforms OMC in spite of the low-scored detections simply selected by top-K sampling. Table 8 further supports the importance of higher-order relational features.

Inference speed. We measure the inference speed in terms of frames-per-second (FPS) using a single V100 GPU. SGT runs at 23.0/23.0/19.9 FPS on MOT16/17/20, respectively. For fair comparison, we select the methods reporting FPS measured on the same GPU. CorrTracker [33] and TransTrack [32] run at 14.8 and 10.0 FPS, respectively, on MOT17, while CorrTracker runs at 8.5 FPS on MOT20. SGT runs much faster than them on both MOT17/20 since SGT performs relational modeling sparsely at object-level by top-K sampling of detections, compared with them densely modeling the relationship of features at pixel-level.

4.3. HiEve Challenge Evaluation

Table 2 compares ours and online JDT models on the HiEve Challenge. Without extra training datasets, SGT achieves 47.2 MOTA and 53.7 IDF1. Under the condition without extra datasets, SGT shows a large improvement in both MOTA and IDF1 compared to FairMOT [45] and CenterTrack [48]. SGT can even achieve comparable MOTA and higher IDF1 than CSTrack [19] which uses extra training datasets following the MOT benchmarks.

4.4. Ablation Experiments

Ablation experiments are conducted by training model on the first half of the MOT17 train dataset and evaluating it on the rest. More analysis on detection recovery and ablation experiments can be found in the supplementary material.

Detection recovery. We conduct analysis of our architecture and comparison with another detection recovery method, BYTE [44], and the results are shown in Table 3. Spatio-temporal relational modeling with top-K detections of the previous frame benefits SGT to extract discriminative edge features. Otherwise, tracking performance is degraded with high IDS. In SGT, low-scored detections are also used for tracking and can be recovered, but FP recovery could be occurred. The node classifier verifies recovery cases with the updated node features and reduces FP. When we compare DR-GNN with BYTE [44] applied on FairMOT [45], DR-GNN achieves better trade-off between FP and FN, and consequently, higher MOTA. This demonstrates the effectiveness of edge features updated in the GNN.

Training strategy. According to Table 4, pseudo labeling based on top-K detections is an important training technique for SGT. Both object–object and object–background pairs are included in edge labels by employing top-K de-

Table 2. Evaluation results of ours and recent online JDT models on the HiEve benchmark (private detection). For each metric, the best is bolded and the second best is underlined. † indicates no extra training dataset.

Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓
DeepSORT [37] †	27.1	28.5	8.4	41.4	5894	42668	3122
JDE [35] †	33.1	36.0	15.1	**24.1**	6318	43577	3747
FairMOT [45] †	35.0	46.7	16.3	44.2	6523	37750	**995**
CenterTrack [48] †	40.9	45.1	10.8	32.2	<u>3208</u>	36414	1568
NewTracker [31]	46.4	43.2	**26.3**	30.8	4667	**30489**	2133
SGT (Ours) †	<u>47.2</u>	**53.7**	<u>24.0</u>	<u>28.8</u>	4699	<u>30727</u>	<u>1361</u>
CSTrack [19]	**48.6**	<u>51.4</u>	20.4	33.5	**2366**	31933	1475

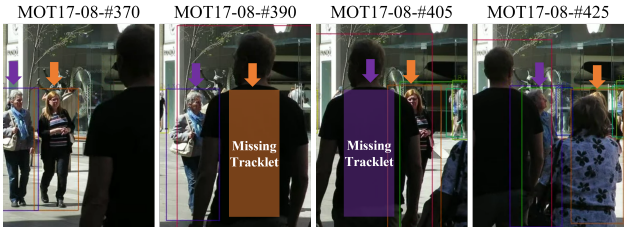

Figure 4. Illustration of long-term association in SGT without Kalman filter [3]. Each color indicates the unique ID.

Table 3. Ablation study of the detection recovery (DR) method. DR–GNN denotes DR by tracking using the edge features updated by GNNs. NC denotes the node classifier that filtering out recovered detections using the node classifier. BG denotes that top-K scored detections of I_{t1} are used as N_{t1}.

Model	DR–GNN	NC	BG	MOTA↑	IDF1↑	MT↑	FP↓	FN↓	IDS↓
SGT (Ours)	✓			70.7	73.3	**49.9**	3400	**11794**	619
	✓	✓		70.8	73.3	45.4	**1952**	13074	750
	✓	✓	✓	**71.3**	**73.8**	46.6	2190	12742	588
FairMOT [45]				69.6	72.5	44.0	2681	13341	414
BYTE [44]				69.7	73.3	47.5	3638	12347	**400**

Table 4. Ablation study of training techniques. J and P denote joint training and pseudo labeling, respectively.

P	J	MOTA↑	IDF1↑	MT↑	FP↓	FN↓	IDS↓
		38.5	55.8	**54.3**	21678	**10136**	1418
✓		69.1	69.7	47.2	3478	12360	847
✓	✓	**71.3**	**73.8**	46.6	**2190**	12742	**588**

Table 5. Effectiveness of adaptive feature smoothing (AdapFS).

Feature Soothing	Adaptive Weight	MOTA↑	IDF1↑	MT↑	FP↓	FN↓	IDS↓
		71.3	73.1	46.9	2178	12724	590
✓		71.4	72.9	46.9	2170	12713	589
✓	✓	71.3	73.8	46.6	2190	12742	588

Table 6. Ablation study of the long-term association in SGT. The unit of age_{max} is second while that of age_{min} is frame.

age_{max}	age_{min}	MOTA↑	IDF1↑	MT↑	FP↓	FN↓	IDS↓
0	0	70.5	63.1	45.7	**2043**	13048	833
1	0	71.2	72.9	**47.2**	2374	**12599**	627
1	1	71.2	**73.8**	**47.2**	2341	12620	**587**
1	10	**71.3**	**73.8**	46.6	2190	12742	588

Table 7. Robustness of K. Memory and time taken for training, MOT performance and speed are measured on different K values.

K (Train)	K (Test)	MOTA↑	IDF1↑	FPS↑	Memory (GB)	Time (hr)
100	50	71.3	73.8	23.6	13.5	3
100	100	71.3	73.8	23.5	13.5	3
100	300	71.5	72.8	22.2	13.5	3
100	500	71.4	72.9	21.8	13.5	3
300	300	71.2	75.2	22.0	14.4	3.4
500	500	71.8	73.4	21.9	15.6	3.8

Table 8. Effect of the number of GNN iterations.

N_{iter}	MOTA↑	IDF1↑	MT↑	FP↓	FN↓	IDS↓
0	67.3	71.2	42.2	2538	14547	578
1	70.9	73.2	46.0	2572	12571	604
2	71.0	73.4	**48.4**	2578	**12516**	**583**
3	**71.3**	**73.8**	46.6	**2190**	12742	588

tections as pseudo labels. Training with object–background pairs as extra negative examples effectively supervises SGT not to false positively match. Jointly training detector and tracker leads to better performance, instead of using frozen backbone which is pretrained with detector.

Effectiveness of AdapFS. With the fixed weight, IDF1 is slightly decreased as shown in Table 5. On the other hand, IDF1 increases and tracking performance is improved based on our proposed adaptive feature smoothing in SGT.

Long-term association. As shown in Table 6, introducing long-term association into SGT significantly increases IDF1. We use age_{min} of 10 frames so that only the stable tracklets are stored and false positive recovery cases are avoided as much as possible. When the objects are fully occluded as shown in Figure 4, SGT fails to track and recover them. However, SGT can match them without a motion model when they reappear.

Robustness of K value. We validate the robustness of K in SGT by training with different K values (*e.g.*, 100, 300, 500) and inference with unseen K values (*e.g.*, 50, 300, 500) using the model trained with $K = 100$. As shown in Table 7, consistent tracking performance is observed across different K values for training and unseen K values. Memory and time consumption for training and inference speed (FPS) are only slightly affected by increasing K.

Number of GNN iterations. As shown in Table 8, FN is much higher when GNN is not yet used for updating edge and node features. Also, more GNN iteration improves MOTA and IDF1. This trend proves that the higher-order relational features are more effective in learning spatio-temporal consistency to perform detection recovery than the pairwise relational features ($N_{iter} = 0$).

Table 9. Effect of choosing different relational features for initializing edge feature.

Combination of features	MOTA↑	IDF1↑	FP↓	FN↓	IDS↓
x, y, w, h, IoU, Sim	**71.3**	**73.8**	2190	**12742**	**588**
x, y, w, h, IoU	69.9	71.5	1953	13508	821
x, y, w, h	70.4	70.6	2113	13205	704
x, y	69.6	70.3	**1691**	13899	837
IoU, Sim	69.6	72.1	2066	13272	640

Design of edge feature. As stated in Eq. 1 of the main paper, we use difference of center coordinates, ratio of width and height, IoU, and cosine similarity to initialize the edge features. Here, both position and appearance relational features are included in the edge features. As shown in Table 9, SGT achieves the best by utilizing all of them.

5. Conclusion and Future Works

Partial occlusion in a video leads to low-confident detection output. Existing online MOT models suffer from missed detections since they use only high-scored detections for tracking. This paper presents SGT, a novel online graph tracker that is jointly trained with a detector and recovers the missed detections by tracking top-K scored detections. We also show that pseudo labeling is critical to training SGT and adaptive feature smoothing is a simple but effective inference technique. SGT captures the object-level spatio-temporal consistency in video data by exploiting higher-order relational features of objects and background patches from the current and past frames. SGT outperforms recent online MOT models in MOTA on MOT16/17/20, but particularly shows a large improvement in MOTA on MOT20 which is vulnerable to missed detections due to occlusion caused by severe crowdedness. Our effective detection recovery method contributes to the outstanding performance of SGT as demonstrated in the extensive ablation experiments. Future works will exploit longer temporal cues and model the spatio-temporal relationship of non-human objects (*e.g.*, vehicles). We hope SGT inspires MOT community to further explore graph tracker and detection recovery by tracking framework for online setting.

Acknowledgement

This work was partially supported by a Research Impact Fund project (R6003-21) and a Theme-based Research Scheme project (T41-603/20R) funded by the Hong Kong Government.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008.

[3] Gary Bishop and Greg Welch. An introduction to the kalman filter. *SIGGRAPH, Course*, 8(27599-23175):41, 2001.

[4] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, pages 1–6. IEEE, 2017.

[5] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *CVPR*, pages 6247–6257, 2020.

[6] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. *NeurIPS*, 29, 2016.

[7] Peng Dai, Renliang Weng, Wongun Choi, Changshui Zhang, Zhangping He, and Wei Ding. Learning a proposal classifier for multiple object tracking. In *CVPR*, pages 2443–2452, 2021.

[8] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020.

[9] Piotr Dollár, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: A benchmark. In *CVPR*, pages 304–311. IEEE, 2009.

[10] Andreas Ess, Bastian Leibe, Konrad Schindler, and Luc Van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, pages 1–8, 2008.

[11] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Detect to track and track to detect. In *ICCV*, pages 3038–3046, 2017.

[12] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, pages 1263–1272. PMLR, 2017.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[15] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[16] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, pages 734–750, 2018.

[17] Yuan Li, Chang Huang, and Ram Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *CVPR*, pages 2953–2960. IEEE, 2009.

[18] Chao Liang, Zhipeng Zhang, Xue Zhou, Bing Li, and Weiming Hu. One more check: Making "fake background" be tracked again. In *AAAI*, volume 36, pages 1546–1554, 2022.

[19] Chao Liang, Zhipeng Zhang, Xue Zhou, Bing Li, Shuyuan Zhu, and Weiming Hu. Rethinking the competition between detection and reid in multiobject tracking. *TIP*, 31:3182–3196, 2022.

[20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017.

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014.

[22] Weiyao Lin, Huabin Liu, Shizhan Liu, Yuxi Li, Rui Qian, Tao Wang, Ning Xu, Hongkai Xiong, Guo-Jun Qi, and Nicu Sebe. Human in events: A large-scale benchmark for human-centric video analysis in complex events. *arXiv preprint arXiv:2005.04490*, 2020.

[23] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *CVPR*, pages 14668–14678, 2020.

[24] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[25] Kenta Oono and Taiji Suzuki. Graph neural networks exponentially lose expressive power for node classification. *arXiv preprint arXiv:1905.10947*, 2019.

[26] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, June 2021.

[27] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *ECCV*, pages 145–161. Springer, 2020.

[28] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[29] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV*, pages 17–35. Springer, 2016.

[30] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[31] Bing Shuai, Andrew Berneshawi, Manchen Wang, Chunhui Liu, Davide Modolo, Xinyu Li, and Joseph Tighe. Application of multi-object tracking with siamese track-rcnn to the human in events dataset. In *ACM Multimedia*, pages 4625–4629, 2020.

[32] Peize Sun, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple-object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020.

[33] Qiang Wang, Yun Zheng, Pan Pan, and Yinghui Xu. Multiple object tracking with correlation learning. In *CVPR*, pages 3876–3886, 2021.

[34] Yongxin Wang, Kris Kitani, and Xinshuo Weng. Joint object detection and multi-object tracking with graph neural networks. In *ICRA*, pages 13708–13715, 2021.

[35] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *ECCV*, pages 107–122. Springer, 2020.

[36] Xinshuo Weng, Yongxin Wang, Yunze Man, and Kris M Kitani. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *CVPR*, pages 6499–6508, 2020.

[37] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *ICIP*, pages 3645–3649. IEEE, 2017.

[38] Jialian Wu, Jiale Cao, Liangchen Song, Yu Wang, Ming Yang, and Junsong Yuan. Track to detect and segment: An online multi-object tracker. In *CVPR*, pages 12352–12361, 2021.

[39] Tong Xiao, Shuang Li, Bochao Wang, Liang Lin, and Xiaogang Wang. Joint detection and identification feature learning for person search. In *CVPR*, pages 3415–3424, 2017.

[40] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *ICCV*, pages 3988–3998, 2019.

[41] En Yu, Zhuoling Li, Shoudong Han, and Hongwei Wang. Relationtrack: Relation-aware multiple object tracking with decoupled representation. *arXiv preprint arXiv:2105.04322*, 2021.

[42] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *CVPR*, pages 2403–2412, 2018.

[43] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. Citypersons: A diverse dataset for pedestrian detection. In *CVPR*, pages 3213–3221, 2017.

[44] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*, 2021.

[45] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *IJCV*, 129(11):3069–3087, 2021.

[46] Linyu Zheng, Ming Tang, Yingying Chen, Guibo Zhu, Jinqiao Wang, and Hanqing Lu. Improving multiple object tracking with single object tracking. In *CVPR*, pages 2453–2462, 2021.

[47] Liang Zheng, Hengheng Zhang, Shaoyan Sun, Manmohan Chandraker, Yi Yang, and Qi Tian. Person re-identification in the wild. In *CVPR*, pages 1367–1376, 2017.

[48] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV*, pages 474–490. Springer, 2020.

[49] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[50] Xingyi Zhou, Tianwei Yin, Vladlen Koltun, and Philipp Krähenbühl. Global tracking transformers. In *CVPR*, pages 8771–8780, 2022.