TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking

Peng Chu[1] Jiang Wang[1] Quanzeng You[1] Haibin Ling[2]
Zicheng Liu[1]
[1]Microsoft [2]Stony Brook University
{pengchu, jiangwang, quanzeng.you, zliu}@microsoft.com, hling@cs.stonybrook.edu

Abstract

Tracking multiple objects in videos relies on modeling the spatial-temporal interactions of the objects. In this paper, we propose TransMOT*, which leverages powerful graph transformers to efficiently model the spatial and temporal interactions among the objects. TransMOT is capable of effectively modeling the interactions of a large number of objects by arranging the trajectories of the tracked targets and detection candidates as a set of sparse weighted graphs, and constructing a spatial graph transformer encoder layer, a temporal transformer encoder layer, and a spatial graph transformer decoder layer based on the graphs. Through end-to-end learning, TransMOT can exploit the spatial-temporal clues to directly estimate association from a large number of loosely filtered detection predictions for robust MOT in complex scenes. The proposed method is evaluated on multiple benchmark datasets, including MOT15, MOT16, MOT17, and MOT20, and it achieves state-of-the-art performance on all the datasets.*

1. Introduction

Robust tracking of multiple objects in video is critical for many real-world applications, ranging from vision-based surveillance to autonomous driving vehicles. Most of the recent state-of-the-art Multiple Object Tracking (MOT) methods use the tracking-by-detection strategy, where target candidates proposed by an object detector on each frame are associated and connected to form target trajectories [3, 17, 23, 36, 40, 54, 57]. In this framework, detection and tracking can be treated separately, usually as two independent modules. This design allows the MOT tracking module to focus on solving association problem while adopting the state-of-the-art single-frame object detectors. It usually leads to higher overall tracking performance as a result. In this paper, we focus on building models for robust target association in the tracking module, where successfully modeling the temporal history and appearance of the targets, as well as their spatial-temporal relationships play an important role.

Traditional models of spatial-temporal relationships usually rely on manually designed association rules, such as social interaction models or spatial exclusion models [31]. The recent advances in deep learning inspire us to model spatial-temporal relationships using deep learning. The success of Transformer suggests a new paradigm of modeling sequential dependencies through the powerful self-attention mechanism. Recent explorations in [33, 47] have shown the feasibility of directly applying transformers for MOT. However, modeling the spatial-temporal relationships of all targets with a general Transformer is ineffective, because of the increasing complexity for representing both temporal and spatial information in the dense format, *e.g.* in the feature tensors, for a large number of objects. It requires large computational resources and training data to successfully learn the long-term temporal dependencies.

On the other hand, separating detection from tracking makes the detector unaware of the frame-to-frame correlation, which becomes a major limitation in complex scenes. To tackle this problem, optimization based trackers [10, 49] propose to consider association for loosely filtered detection predictions to recover occlusion and motion blur. But only pair-wise affinities clues like position and appearance are explored in those works, no higher order temporal-spatial information such as motion or gait can be leveraged. Their offline method design and threshold-based graph or hypothesis construction also limit their application.

In this paper, we propose a novel spatial-temporal graph Transformer for MOT (TransMOT) to resolve all the issues. In TransMOT, objects are arranged as a temporal series of sparse weighted graphs that are constructed using their spatial relationships within each frame. This formulation effectively handles a large and varied number of tracked targets and detection candidates during tracking. Using the graph representation, TransMOT encodes the features and spatial-temporal relationships for all tracked targets through its Spatial-Temporal Graph Transformer Encoder. The decoder models spatial and appearance correlations of detec-

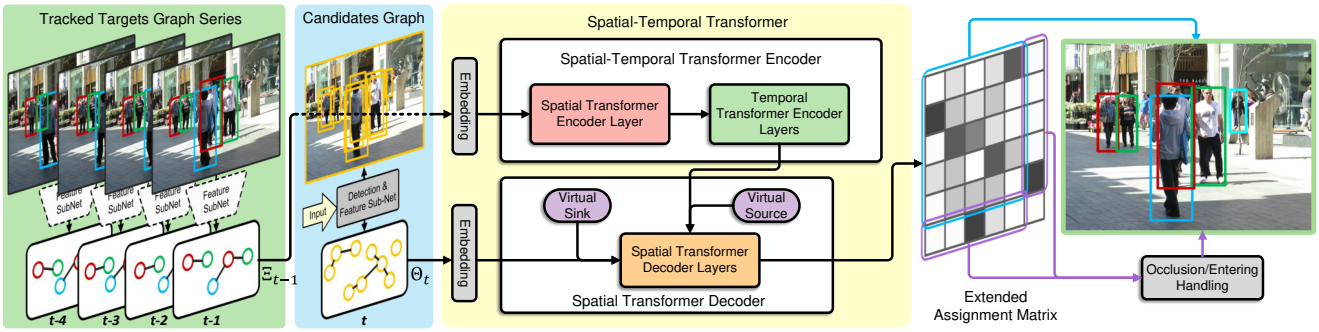

Figure 1. Overview of the proposed TransMOT for online MOT. The tracklets graph series $\boldsymbol{\Xi}_{t-1}$ till frame $t-1$ and detection candidates graph Θ_t at frame t serve as the source and target inputs, respectively, to the spatial-temporal graph transformer.

tion candidates by a specially designed Spatial Graph Transformer Decoder. Through end-to-end training, TransMOT can learn the spatial-temporal clues for association and directly generate the assignment matrix for MOT. Moreover, relying on the discriminative spatial-temporal clues and capability of modeling a large number of candidates, TransMOT can associate candidates from the a large number of loosely filtered detection predictions, most of which can be discarded by the post-processing of a detector.

In summary, we make the following contributions:

- We propose a spatial-temporal graph Transformer (TransMOT) to effectively model the spatial-temporal relationship of the objects for end-to-end learnable association in MOT.
- We design a TransMOT based tracker to model a large number of loosely filtered predictions from any single-frame detectors for robust MOT in complex scenes.

For evaluation, extensive experiments have been conducted on MOT15 [26], MOT16 [34], MOT17 and MOT20 [16] challenge datasets. The results demonstrate that the proposed TransMOT achieves the best overall performance and establishes a new state-of-the-art.

2. Related Works

Most of the recent Multiple Object Tracking (MOT) trackers are based on the tracking-by-detection framework. Tracking-by-detection framework generates tracklets by associating object detections in all the frames using matching algorithms such as Hungarian algorithm [6, 18, 22], network flow [15, 62, 63], and multiple hypotheses tracking [10, 24]. Many works solve the association problem by building graphs of object detections across all the frames, such as multi-cuts [23, 49] and lifting edges [50]. However, these methods need to perform computationally expensive global optimization on large graphs, which limits their application to online tracking.

Recently, deep learning-based association algorithm is gaining popularity in MOT [68]. In [37] and [35], recurrent neural networks (RNNs) are explored to solve the association problem using only the motion information. In [11], a power iteration layer is introduced in the rank-1 tensor approximation framework [46] to solve the multi-dimension assignment in MOT. In [58], a differentiable MOT loss is proposed to learn deep Hungarian Net for association. In [8], graph convolutional neural network is adopted as a neural solver for MOT, where a dense graph connecting every pair of nodes in different frames is constructed to infer the association. The proposed TransMOT also constructs a spatial graph for the objects within the same frame, but it exploits the Transformer networking architecture to jointly learn the spatial and temporal relationship of the tracklets and candidates for efficient association.

Some MOT solutions take into consideration of the target detection, either on the detector side to aggregate the temporal clues (*e.g.* [30, 67]) or on the tracker side to add additional object detection or single object tracking techniques (*e.g.* [4, 12]). However, those methods need specially designed detector structures or video-based training data, which greatly hurt the flexibility and efficiency of the original tracking-by-detection framework. By contrast, our method directly leverages the output from a generic image object detector and is capable of combining it with the learnable detector such as DETR to form a fully end-to-end tracker.

Transformer has achieved great success in various computer vision tasks, such as detection [9] and segmentation [28]. In [61], Transformer is adopted for trajectory prediction. The studies in [33,47] are the pioneer investigations in applying Transformer in MOT. Both methods use DETR for detection and feature extraction, and model the spatial-temporal relationship of the tracklets and detections using Transformer. The proposed TransMOT framework utilizes spatial graph transformer to model spatial relationship of the tracklets and detections, and it factorizes the spatial and temporal transformer encoder for efficient modeling.

3. Overview

We aim at joint detection and tracking multiple objects in videos in an online fashion. Our method, named TransMOT (Spatial-temporal graph Transformer for MOT), is built upon the tracking-by-detection framework, as shown in Fig. 1. At the t-th frame, TransMOT maintains a set of N_{t-1} tracklets, each of which represents a tracked object. Each tracklet l_{t-1}^i maintains a set of states, such as its past locations and appearance features on the previous T image frames. Given a new image frame I_t, the online tracking algorithm eliminates the tracklets whose tracked object exits the scene, determines whether any tracked objects are occluded, computes new locations for the existing tracklets , and generates new tracklets for new objects that enter the scene.

As shown in Fig. 1, our framework contains two major parts: the detection and feature extraction sub-networks, and spatial temporal graph transformer association sub-network. At each frame, the detection and feature extraction sub-networks generate M_t candidate object detection proposals $\mathbf{O}_t = \{o_t^j\}_{j=1}^{M_t}$, as well as visual features for each proposal. The spatial-temporal graph transformer finds the best candidate proposal for each tracklet and models the special events, such as entering, exiting, or occlusion.

For each tracklet l_{t-1}^i, its matching score to an object proposal o_t^j is measured by the affinity $\phi(l_{t-1}^i, o_t^j)$, where $\phi(\cdot)$ computes the affinity of the tracklet state and the candidate. Taking all tracklets into consideration, the problem can be formulated as a constrained optimization problem as

$$\max_{A_t=(a_{ij}^t)} \sum_{i=1}^{N_{t-1}} \sum_{j=1}^{M_t} a_{ij}^t \phi(l_{t-1}^i, o_t^j), \tag{1}$$

$$\text{s.t.} \begin{cases} \sum_i a_{ij}^t = 1, & \forall i = 1, \ldots, N_{t-1} \\ \sum_j a_{ij}^t = 1, & \forall j = 1, \ldots, M_t \\ a_{ij}^t \in \{0, 1\}, & \forall i = 1, \ldots, N_{t-1}; j = 1, \ldots, M_t \end{cases} \tag{2}$$

where $A_t = (a_{ij}^t)$ indicates the association between tracklets $\mathbf{L}_{t-1} = \{l_{t-1}^i\}_{i=1}^{N_{t-1}}$ and detected candidates $\mathbf{O}_t$. Eq. 2 is used to enforce the assignment constraints.

In order to more effectively model the spatial-temporal relationship between all the tracklets and candidates, the proposed framework rewrites Eq. 1 and Eq. 2 into a single function $A_t = \Phi(\mathbf{L}_{t-1}, \mathbf{O}_t)$, where $\mathbf{L}_{t-1}$ and $\mathbf{O}_t$ consist of all the tracklets and candidates, respectively.

To model the spatial-temporal object correlation, we build a weighted spatial graph Θ_t for the proposals at the current frame, and a set of weighted spatial graphs $\Xi_{t-1} = \{\xi_{t-T}, \xi_2, \ldots, \xi_{t-1}\}$ of the tracked objects at the previous T frames. The spatial-temporal graph neural network utilizes these graphs to build an efficient spatial-temporal graph transformer that models the relationship between the tracked objects and newly generated proposals. It generates an assignment matrix $\bar{A}_t$ to track the objects and model the special events, such as entering, exiting, or occlusion, as shown in Fig. 1. Our tracker depends on the assignment matrix to update the tracked targets, which will be explained in Sec. 4.4. The details of the spatial-temporal graph Transformer will be explained in Sec. 4.1 and Sec. 4.2. The two types of training losses to train TransMOT will be elaborated in Sec. 4.3.

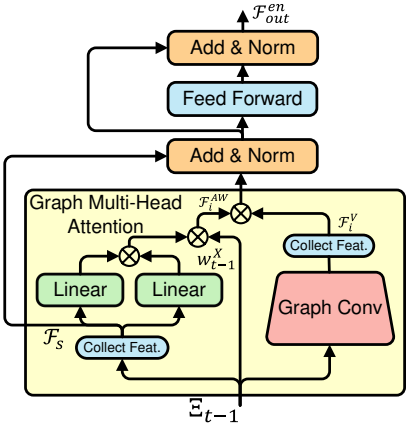

Figure 2. The spatial graph transformer encoder layer.

4. TransMOT

TransMOT uses the graphs Ξ_{t-1} and Θ_t to learn a mapping $\Phi(\cdot)$ that models the spatial-temporal correlations, and generates an assignment/mapping matrix $\bar{A}_t$. It contains three parts: a spatial graph transformer encoder layer, temporal transformer encoder layers, and a spatial graph transformer decoder layer. We propose graph multi-head attention to model spatial relationship of the tracklets and candidates using the self-attention mechanism. It is crucial for both the spatial graph transformer encoder layer and the spatial graph transformer decoder layer.

4.1. Spatial-Temporal Graph Transformer Encoder

The spatial-temporal graph encoder consists of a spatial-temporal graph transformer encoder layer to model the spatial correlation among tracklets, and two temporal transformer encoder layers to further fuse and encode the spatial and temporal information of the tracklets. We find that factoring the transformer into spatial and temporal transformers makes the model both more accurate and computationally efficient.

4.1.1 Spatial Graph Transformer Encoder Layer

The input of spatial-temporal graph encoder layer is the states of the tracklets for the past T frames. The tracklet state features are arranged using a sequence of track-

let graphs $\Xi_{t-1} = \{\xi_{t-T}, \xi_{t-T+1}, \ldots, \xi_{t-1}\}$, where $\xi_t = G(\{x_t^i\}, E_t^X, w_t^X)$ is the spatial graph[1] of the tracklets at frame t. The graph node x_t^i represents the status of i-th tracklet at frame t, two nodes are connected by an edge in E_t^X if their corresponding bounding boxes have IoU larger than 0, and the edge weight in w_t^X is set to the IoU. The weight matrix $w_t^X \in \mathbb{R}^{N_t \times N_t}$ is a sparse matrix, whose (i, j) entry is the weight of the edge connecting node i and node j, or 0 if they are not connected.

The node features for the tracklets are first embedded through a source embedding layer (a linear layer) independently for each node. All the node features are arranged into a feature tensor $\mathcal{F}_s \in \mathbb{R}^{N_{t-1} \times T \times D}$, where D is the dimension of the source embedding layer. It is passed into the spatial graph transformer encoder layer together with the graph series as shown in Fig. 2. Inside the layer, a multi-head graph attention module is utilized to generate self-attention for the input graph series. This module takes feature tensor $\mathcal{F}_s$ and the graph weights w_{t-1}^X to generate self-attention weights for the i-th head:

$$\mathcal{F}_i^{AW} = \text{softmax}\Big[\varphi(\mathcal{F}_s, W_i^Q, W_i^K) \circ w_{t-1}^X\Big], \quad (3)$$

where $\varphi(\cdot)$ is the regular scaled dot-product to obtain attention weights as in [52], W_i^Q, W_i^K are learnable linear projection matrices, and $\circ$ is the element-wise product. It can be understood as computing the spatial graph self-attention for each timestamp independently.

The multi-head graph attention utilizes the graph weights w_{t-1}^X to generate non-zero attention weights only for the tracklets that have spatial interactions, because the tracklets that are far away from each other usually have very little interaction in practice. By focusing its attention on a much smaller subset, the spatial graph transformer encoder layer models the interactions more effectively. We also apply graph convolution instead of the linear layer to aggregate information from neighboring nodes. After the graph convolution layer, the node features are collected to form a value tensor $\mathcal{F}_i^V$. Combined with the attention weights in Eq. 3, the graph multi-head attention weighted feature tensor can be written as

$$\mathcal{F}_{att}^{en} = \text{Concate}(\{\mathcal{F}_i^{AW} \otimes \mathcal{F}_i^V\}) \otimes W^O,$$

where $\{\cdot\}$ iterates and aggregates the outputs from all the attention heads, W_i^O is a learnable linear projection matrix, and $\otimes$ is the tensor mode product.[2]

The attention-weighted feature tensor is projected through a linear feed-forward and a normalization layer to get the final output of the spatial graph transformer encoder layer.

[1] We use $G(\cdot)$ to denote a graph.

[2] It performs matrix product of each slice of right and left tensors along the dimension sharing the same length.

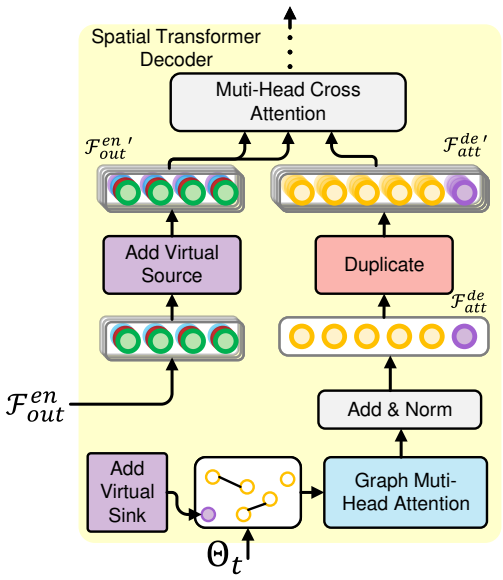

Figure 3. Illustration of the spatial graph transformer decoder.

4.1.2 Temporal Transformer Encoder Layer

The features of the tracklets are further encoded by a temporal transformer encoder layer. The temporal transformer encoder layer transposes the first two dimensions of the output tensor from the spatial graph transformer encoder, resulting in a tensor $\mathcal{F}_{tmp}^{en} \in \mathbb{R}^{T \times N_{t-1} \times D}$. The temporal transformer encoder layer employs a standard Transformer encoder layer over the temporal dimension for each tracklets independently. It calculates the self-attention weights along the temporal dimension, and computes the temporal attention-weighted feature tensor for the tracklets.

The output of the temporal transformer encoder layer is the final output of the spatial-temporal graph transformer encoder $\mathcal{F}_{out}^{en}$.

4.2. Spatial Graph Transformer Decoder

The spatial graph transformer decoder produces extended assignment matrix $\bar{A}_t$ from the candidate graph $\Theta_t = G(\{o_t^j\}, E_t^O, w_t^O)$ and the output of the spatial-temporal transformer encoder $\mathcal{F}_{out}^{en}$. The candidate graph is constructed similarly to the tracklet graphs in Sec. 4.1. Each node o_j^t represents a candidate in frame t. Two nodes are connected only if their bounding boxes' IoU is larger than 0, and the weight of the edge is set to the IoU. Besides the nodes representing the real candidates, a virtual sink node is added to the graph. The virtual sink node is responsible for exiting or occlusion events of any tracklet in the current frame. In particular, a node with a set of learnable embedding $f_{snk} \in \mathbb{R}^D$ is added to the Θ_t. The virtual sink node is connected to all the other nodes with weight 0.5.

Similar to the encoder in Sec. 4.1, the candidate node

features of input graph are embedded and collected. The f_{snk} is appended to the embedded feature set such that $\mathcal{F}_{tgt}^{de} \in \mathbb{R}^{(M_t+1)\times 1\times D}$. The spatial graph decoder first uses graph multi-head attention to encode the node features that are similar to the one Sec. 4.1, shown in Fig. 3. We denote the attention weighted candidate node features as $\mathcal{F}_{att}^{de} \in \mathbb{R}^{(M_t+1)\times 1\times D}$.

For the tracklet embedding $\mathcal{F}_{out}^{en}$ generated by the spatial-temporal graph transformer encoder, we add a virtual source to handle the candidates that either is false positives or need to initiate a new tracklet in the current frame t to form an extended tracklet embedding $\mathcal{F}_{out}^{en'} \in \mathbb{R}^{T\times(N_{t-1}+1)\times D}$. The embedding of the virtual source is a learnable parameter. Note that we only add one virtual source node compared to multiple virtual source nodes in Transform-based MOT trackers, because we find adding one virtual source node yields comparable performance as adding multiple virtual source nodes while achieving better computational efficiency. $\mathcal{F}_{att}^{de}$ is duplicated $N_{t-1}+1$ times such that $\mathcal{F}_{att}^{de} \rightarrow \mathcal{F}_{att}^{de'} \in \mathbb{R}^{(M_t+1)\times(N_{t-1}+1)\times D}$. Multi-head cross attention is calculated for $\mathcal{F}_{att}^{de'}$ and $\mathcal{F}_{out}^{en'}$ to generate unnormalized attention weights. The output is passed through a feed-forward layer and a normalization layer to generate the output tensor $\mathbb{R}^{(M_t+1)\times(N_{t-1}+1)\times D}$ that corresponds to the matching between the tracklets and the candidates.

The output of the spatial graph decoder can be passed through a linear layer and a Softmax layer to generate the assignment matrix $\bar{A}_t \in \mathbb{R}^{(M_t+1)\times(N_{t-1}+1)}$.

4.3. Training

TransMOT is trained end-to-end with the guidance of the groundtruth extended assignment matrix. The constraints in Eq. 2 need to be relaxed to allow efficient optimization. We relax the constraints so that a tracklet is always associated with a detection candidate or a virtual source. In this way, Eq. 2 can be relaxed to:

$$\text{s.t.} \quad \sum_{j}^{M_t+1} \bar{a}_{ij}^{t} = 1, i \in [1, N_{t-1}], \bar{a}_{ij}^{t} \in \{0, 1\}.$$

As a result, a row of the assignment matrix can be treated as a probability distribution over a total of M_t+1 categories, and we use the cross-entropy loss to optimize the network.

In each training iteration, a continuous sequence of $T+1$ frames is randomly sampled from the training set. The bounding boxes and their corresponding IDs are collected from each frame. The ground-truth bounding boxes are then replaced by the bounding boxes generated from the object detector by matching their IoUs. The rest unmatched detections predictions in the $T+1$ frame are false-positive predictions and need for optimizing to connect with virtual source.

For the rows that correspond to actual tracklets, a cross-entropy loss is utilized, as mentioned above. The last row of $\bar{A}_t$ represents the virtual source, and it may match to multiple detection candidates. Thus, a multi-label soft margin loss is employed to optimize this part separately.

In summary, the overall training loss can be written as

$$\begin{aligned}\mathcal{L} = &-\frac{1}{N_{t-1}}\sum_{n=1}^{N_{t-1}} y_n \log(\bar{\mathbf{a}}_n) \\ &+ \frac{\lambda}{M_t}\sum_{m=1}^{M_t} y_m^{src}\log\Big(\frac{1}{1+e^{-a'_m}}\Big) \\ &+ \frac{\lambda}{M_t}\sum_{m=1}^{M_t}(1-y_m^{src})\log\Big(\frac{e^{-a'_m}}{1+e^{-a'_m}}\Big),\end{aligned}$$

where y and y_n^{src} are association labels for real tracklets and the virtual source respectively, $\bar{\mathbf{a}}_n$ is the row element of $\bar{A}_t$, $\bar{\mathbf{a}}_{N_{t-1}+1} = \{a'_m\}$, and λ is a weighting coefficient.

4.4. Tracking Framework

This section describes how the final tracking association is generated from the association matrix $\bar{A}_t$ output by the TransMOT. The upper left part of the matrix $\hat{A}_t \in \mathbb{R}^{M_t\times N_{t-1}}$ indicates the assignment scores of the tracklets and candidate boxes. Since the elements of $\hat{A}_t \in [0, 1]$ is a soft assignment, we apply a bipartite matching Hungarian algorithm to generate the actual matching. To reduce false positive associations, only the pairs with assignment scores higher than a threshold will be matched at this step.

A re-match stage is utilized to improve the recall of the tracking association. In this stage, un-matched high confident detections are associated with the remaining tracklets. The association cost is defined as the sum of the Euclidean distance of the visual features and the normalized top distance of the tracklets and candidate boxes.

$$d_{top} = \Big\|\Big(u_i + \frac{w_i}{2} - u_j - \frac{w_j}{2}, v_i - v_j\Big)\Big\| \Big/ h_i,$$

where $[u, v]$ indicates the left upper corner of the bounding box and $[w, h]$ indicates its size[3].

Since, TransMOT only models the tracklets of the previous T frames, we have a long-term occlusion handling stage to match the tracklets that are occluded for more than T frames. For these tracklets, we store their visual features at the latest visible frames, and use them to calculate the association cost of the tracklets and candidate detections. After handling long-term occlusion, the remaining un-associated detection candidates are matched with all tracked targets again to remove potentially duplicated detections for the same target.

[3]The notation w should not be confused with the weight w in Sec. 4.1.

Method	IDF1	MOTA	MT	ML↓	FP↓	FN↓	IDS↓
DMT [25]	49.2	44.5	34.7%	22.1%	8,088	25,335	684
TubeTK [38]	53.1	58.4	39.3%	18.0%	**5,756**	18,961	854
CDADDAL [3]	54.1	51.3	36.3%	22.2%	7,110	22,271	544
TRID [32]	61.0	55.7	40.6%	25.8%	6,273	20,611	351
RAR15 [18]	61.3	56.5	45.1%	14.6%	9,386	16,921	428
GSDT [55]	64.6	**60.7**	47.0%	**10.5%**	7,334	16,358	477
Fair [65]	64.7	60.6	47.6%	11.0%	7,854	15,785	591
TransMOT	**66.0**	57.0	**64.5%**	17.8%	12,454	**13,725**	**244**

Table 1. Tracking Performance on the MOT15 benchmark test set private detection track. Best in bold.

Method	IDF1	MOTA	MT	ML↓	FP↓	FN↓	IDS↓
IoU [7]	46.9	57.1	23.6%	32.9%	5,702	70,278	2,167
CTracker [39]	57.2	67.6	32.9%	23.1%	8,934	48,305	1,897
LMCNN [2]	61.2	67.4	38.2%	19.2%	10,109	48,435	931
DeepSort [56]	62.2	61.4	32.8%	18.2%	12,852	56,668	781
FUFET [44]	68.6	76.5	52.8%	**12.3%**	12,878	28,982	1,026
LMP [50]	70.1	71.0	46.9%	21.9%	**7,880**	44,564	**434**
CSTrack [27]	73.3	75.6	42.8%	16.5%	9,646	33,777	1,121
TransMOT	**76.8**	**76.7**	**56.5%**	19.2%	14,999	**26,967**	517

Table 2. Tracking Performance on the MOT16 benchmark test set private detection track. Best in bold.

Finally, each of the remaining candidates is initialized as a new tracklet, and the unresolved tracklets that have not been updated for more than K_p frames are removed. The tracklets remaining un-updated for less than K_p frames are set to the "occluded" state.

5. Experiments

We conduct extensive experiments on four standard MOT challenge datasets for pedestrian tracking: MOT15 [26], MOT16 [34], MOT17, and MOT20 [16]. The proposed TransMOT based tracking framework is evaluated on both public and private detection tracks.

5.1. Experiment Setting and Implementation Details

The proposed approach is implemented in PyTorch, and the training and inference are performed on a machine with a 10 cores CPU@3.60GHz and an Nvidia Tesla V100 GPU. We set the number of frames for tracklets $T = 5$, the feature embedding dimension $D = 1024$, and the number of heads for all the multi-headed attention in spatial and temporal transformers to 8. For graph multi-head attention module, a single layer of ChebConv from [14] with a neighboring distance of 2 is adopted. The node features for an object at a frame are the concatenation of its visual features, normalized bounding box coordinates and detection confidence score. During training, we use vanilla SGD with an initial learning rate of 0.0015. For all the experiments in Sec. 5.2, we use the training dataset from [29] to train our TransMOT model. During inference, K_p is set 50. NMS with 0.6 IoU and 0.01 confidence threshold are employed as the loosely filtering to include as many raw detection proposals for training TransMOT, while 0.05 is used during tracking for balanced inference speed.

We trained a YOLOv5 [1](v3.0) detector with the model 'X' configuration on the combination of CrowdHuman dataset [45] and the training sets of MOT17/MOT20. The SiamFC network [43] pretrained on the ILSVRC15 dataset is adopted as our visual feature extraction sub-network. The maximum input image dimension of the tracking pipeline is set to 1920. The detector runs at 15.4 *fps* on our machine, while the TransMOT including visual feature extraction sub-network runs at 23.2 *fps*. The whole tracking pipeline runs at 9.3 *fps*. We also experimented with using TransTrack [47] as our detection and feature extraction sub-network, as well as other visual features. These comparisons will be compared in the MOT17 and ablation parts of Sec. 5.2.

To evaluate the performance of the proposed method, the standard ID score metrics [42] are reported. ID score metrics measure the long-term ID consistency and compare whole trajectories with ground truth for ID precision (IDP), ID recall (IDR), and their IDF1 scores. Therefore, IDF1 focuses more on association quality and is not sensitive to the accuracy of the individual bounding box, which is suitable for us to compare with peer trackers that use different private detectors. Following other MOT works, CLEAR MOT metrics [5], *e.g.*, multiple object tracking accuracy (MOTA) is also reported. It combines bounding box false positives (FP), false negatives (FN) and the identity switches (IDS). The percentage of mostly tracked targets (MT) and the percentage of mostly lost targets (ML) are reported.

5.2. Evaluation Results

MOT15. MOT15 [26] contains 22 different indoor and outdoor scenes for pedestrian tracking. The 22 sequences are collected from several public and private datasets, and they were recorded with different camera motion, camera angles and imaging conditions. The dataset is equally split for training and testing. We report the quantitative results of the proposed method on the private detection track in Tab. 1 and visualization on selected videos in Fig. 4. The updated results of peer trackers in the MOTChallenge leaderboard are collected for comparison. TransMOT achieves state-of-the-art performance in metrics IDF1, MT, FN, and IDS. The relatively lower MOTA score on this dataset is caused by the high FP rate, because not all the objects are exhaustively annotated for some testing sequences.

MOT16/17. MOT16 and MOT17 [34] contain the same 14 videos for pedestrian tracking. MOT17 has more accurate ground truth annotations compared to MOT16 dataset. MOT17 also evaluates the effect of object detection quality on trackers, by providing three pretrained object detectors using DPM [19], Faster-RCNN [41] and SDP [60]. We

	Method	IDF1	MOTA	MT	ML↓	FP↓	FN↓	IDS↓
Public Detection	TrctrD [58]	53.8	53.7	19.4%	36.6%	11,731	247,447	1,947
	Tracktor [4]	55.1	56.3	21.1%	35.3%	8,866	235,449	1,987
	CTTrack [67]	59.6	61.5	26.4%	31.9%	14,076	200,672	2,583
	TrackFormer [33]	60.7	62.5	29.8%	26.8%	32,828	174,921	3,917
	MPNTrack [8]	61.7	58.8	28.8%	33.5%	17,413	213,594	1,185
	LifT [21]	65.6	60.5	27.0%	33.6%	14,966	206,619	1,189
	MAT [20]	69.2	67.1	**38.9%**	**26.4%**	22,756	**161,547**	1,279
	TransMOT-P	**73.1**	**68.8**	33.1%	31.5%	**8,080**	167,174	**1,043**
Private Detection	DAN [48]	49.5	52.4	21.4%	30.7%	25,423	234,592	8,431
	TubeTK [38]	58.6	63.0	31.2%	19.9%	27,060	177,483	5,727
	TransTrack [47]	63.9	74.5	46.8%	**11.3%**	28,323	112,137	3,663
	CTTrack [67]	64.7	67.8	34.6%	24.6%	**18,498**	160,332	6,102
	Perma [51]	71.9	69.5	42.5%	17.7%	-	-	-
	TLR [53]	73.6	**76.5**	47.6%	12.7%	29,808	**99,510**	3,369
	TransMOT-D	69.3	68.5	34.2%	34.1%	22,767	153,156	1,635
	TransMOT	**76.3**	76.4	**48.7%**	21.9%	31,788	99,651	**1,623**

Table 3. Tracking Performance on the MOT17 benchmark test set. Best in bold.

	Method	IDF1	MOTA	MT	ML↓	FP↓	FN↓	IDS↓
Public Det.	SORT* [6]	45.1	42.7	16.7%	26.2%	27,521	264,694	4,470
	Tracktor [4]	52.7	52.6	29.4%	26.7%	6,930	236,680	1,648
	MPNTrack [8]	59.1	57.6	38.2%	22.5%	16,953	201,384	1,210
	LPCMOT [13]	62.5	56.3	34.1%	25.2%	11,726	213,056	1,562
	TransMOT-P	**74.2**	**73.4**	**54.7%**	**14.6%**	**11,511**	**125,029**	**1,008**
Private Det.	MLT [64]	54.6	48.9	30.9%	22.1%	45,660	216,803	2,187
	GSDT [55]	67.5	67.1	53.1%	13.2%	31,913	135,409	3,131
	Fair [65]	67.3	61.8	68.8%	**7.6%**	103,440	88,901	5,243
	CSTrack [27]	68.6	66.6	50.4%	15.5%	**25,404**	144,358	3,196
	ReMOT [59]	73.1	77.4	68.1%	9.9%	28,351	86,659	1,789
	TransMOT	**75.2**	**77.4**	**70.1%**	9.2%	32,335	**82,867**	**1,601**

Table 4. Tracking Performance on the MOT20 benchmark test set. Best in bold. Method marked with * in public detection track does not use public detection filtering mechanism. It might achieve better tracking accuracy if the mechanism is employed.

report the performance and comparisons with the state-of-the-art methods on the private detection track of MOT16 in Tab. 2. Our approach outperforms all other published trackers using the private detector in IDF1 metrics.

In MOT17, for a more complete comparison, we configure TransMOT as two additional settings: TransMOT-P and TransMOT-D. TransMOT-P uses the public detection results, and follows the filtering mechanism adopted by [4] and [67]. A new trajectory is initialized only if its bounding box at the current frame overlaps a public detection with IoU larger than 0.5. We compare TransMOT-P with other trackers adopting the same filtering mechanism on the public detection track of MOT17 in Tab. 3. Compared with regular Transformer-based tracker [33], TransMOT outperforms it in IDF1, MOTA, and IDS by a large margin. TransMOT also achieves the best IDF1 and MOTA scores among all published trackers, which demonstrates the robustness of TransMOT against detection quality variations.

TransMOT-D adopts the DETR framework as detection and visual feature extraction sub-networks. TransMOT-D takes the detection outputs of TransTrack [47] (validation model) and their Transformer embedding as visual features. For a fair comparison, the pretrained model of TransTrack is not fine-tuned in TransMOT-D. We compare TransMOT-D and TransMOT with state-of-the-art trackers on MOT17 private detection track in Tab. 3. The performance of TransMOT-D is better than TransTrack by 5.4 in IDF1. This shows that our TransMOT framework can better model the spatial-temporal relationship of the tracklets and detections than standard Transformer. TransMOT with normal configuration achieves the best IDF1, ML and IDS among all published works using the private detector.

MOT20. MOT20 consists of eight sequences for pedestrian tracking. MOT20 video sequences are more challenging because they have much higher object density, *e.g.* 170.9 *vs* 31.8 in the test set. We report the experimental results of the proposed TransMOT and the comparison with the other methods in Tab. 4. Our method establishes state-of-the-art in most metrics among all peer works. In public detection setting, TransMOT-P also demonstrates its robustness to detection noise. Compared with the private detector setting, the MOTA decreases 4.0, but the IDF1 score only drops 1.0. The experiments on both public and private detection settings demonstrate that the capability of TransMOT of modeling a large number of tracklets and detections in crowd scenes.

5.3. Ablation

We study the significance of components and hyperparameters in the proposed method through ablation study as shown in Tab. 5. The ablations are conducted on the MOT17 *training* set. To avoid overfitting, the object detector used in the ablation study is trained on only the CrowdHuman dataset.

We first evaluate the effectiveness of the TransMOT. The performance of removing TransMOT from the tracking framework is noted as *w/o* TransMOT and reported in Tab. 5. The influence of using loosely filtered detection is also investigated, where strict filtering, *i.e.* 0.35 confidence threshold, is applied to the excessive input detections to leave only high confident candidates (Filtered Det) for association. Compared with the full setting, marked as Ours in Tab. 5, these results demonstrate that TransMOT can effectively learn to association and leverage extensive loosely filtered detection predictions to improve association performance. Regarding inference speed, when TransMOT is absent, the rest logic in the tracking framework runs at 33.2 *fps* which is consumed mainly by the depending visual feature sub-net. Using loosely filtered predictions adds around 40% more candidates (100K to 139K) for association, but only reduces FPS from 26.7 to 23.2.

The importance of spatial and temporal information is also evaluated. We first set all spatial related features in

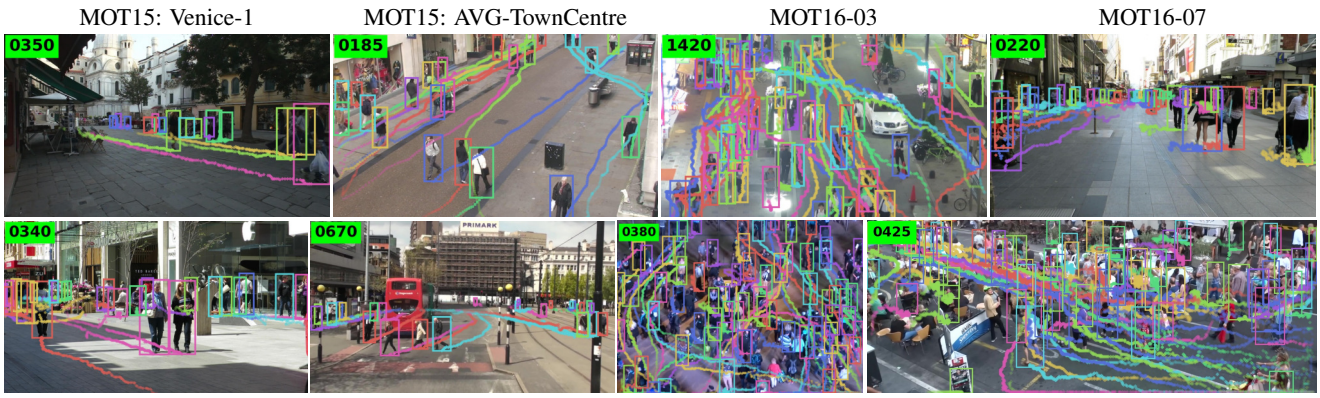

Figure 4. Results visualization of selected sequences in MOT15, MOT16, MOT17, and MOT20.

TransMOT input to zero (*w/o* Spatial), which results in isolated graph nodes and zeros in spatial features for each node. In order to exclude the temporal information, we run the tracking framework at $T = 1$ and mark it as *w/o* Temporal in Tab. 5. The decrease in IDF1 shows the importance of modeling the spatial-temporal information in tracked targets and candidates for association. The reason for less performance degradation when excluding spatial information than temporal information is that the spatial information cannot be completely excluded from the tracking framework. Other logic such as re-matching also relies on spatial information for association, which compensates for the absence of spatial features in TransMOT.

The choice of the temporal history length T is also investigated. Beside $T = 1$ in previous ablation and $T = 5$ in our full setting, $T = 10$ and $T = 20$ are also tested. We find that increasing T for more than 5 does not improve tracking performance. Including longer temporal history increases the complexity of the association task and makes the learning harder under a limited number of training data for MOT. It also significantly lowers the inference speed, *e.g.* 17.1 *fps* when $T = 20$.

Note that, MOTA metric also decreases in the above settings compared with the full setting, but not as significantly as IDF1. The reason comes in two folds: Firstly, by successfully associating targets during occlusion, the tracker keeps the ID consistent of a tracklet before and after occlusion. It will greatly improve the IDR and IDP performance, but will not significantly affect the bounding box metrics FP and FN in MOTA. Second, in MOT17, the ground truth bounding boxes with large occlusion are usually marked as ignored and excluded from evaluation for fair due to their unambiguous shape. Some of the bounding boxes TransMOT recovered from the loosely filtered detections fall in this case.

Finally, in addition to SiamFC and DETR features, we evaluate other shallow and deep visual features, including color histogram and ReID feature DGNet [66]. Benefiting from the fully trainable Transformer architecture, even using simple color histogram features, TransMOT can achieve similar performance with the one using deep ReID features but runs at a much faster inference speed. On the other hand, SiamcFC features perform better than both color histogram and ReID features, because it is trained on a large scale video dataset.

Configuration	IDF1	MOTA	FPS
w/o TransMOT	69.7	64.6	33.2
TransMOT, Filtered Det.	77.1	73.8	26.7
TransMOT, *w/o* Spatial	78.0	74.7	23.3
TransMOT@$T = 1$ (*w/o* Temporal)	76.7	74.6	25.7
TransMOT@$T = 10$	77.9	74.2	21.0
TransMOT@$T = 20$	77.1	74.1	17.1
TransMOT+Histogram	77.2	74.5	45.7
TransMOT+DGNet	77.6	74.4	9.3
TransMOT($T = 5$)+SiamcFC(**Ours**)	79.0	74.7	23.2

Table 5. Ablations on the MOT17 benchmark training set. FPS indicates the inference speed for tracker only not including detector.

6. Conclusion

We proposed a novel spatial-temporal graph Transformer for Multi-Object Tracking (TransMOT). By formulating the tracklets and candidate detections as a series of weighted graphs, the spatial and temporal relationships of the tracklets and candidates are explicitly modeled and leveraged. The proposed TransMOT not only achieves higher tracking accuracy, but also is more computationally efficient than the transitional Transformer-based methods. Experiments on MOT15, MOT16, MOT17, and MOT20 challenge datasets show that the proposed approach achieves state-of-the-art performance on all the benchmark datasets.

Acknowledgements. We would like to thank Yumao Lu for his support. Ling was supported in part by US National Science Foundation Grant 2006665.

References

[1] Yolov5. https://github.com/ultralytics/yolov5/tree/v3.0.

[2] Maryam Babaee, Zimu Li, and Gerhard Rigoll. A dual cnn–rnn for multiple people tracking. *Neurocomputing*, 368:69–83, 2019.

[3] Seung-Hwan Bae and Kuk-Jin Yoon. Confidence-based data association and discriminative deep appearance learning for robust online multi-object tracking. *TPAMI*, 2018.

[4] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019.

[5] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *JIVP*, 2008.

[6] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016.

[7] Erik Bochinski, Volker Eiselein, and Thomas Sikora. High-speed tracking-by-detection without using image information. In *AVSS*, 2017.

[8] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020.

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[10] Jiahui Chen, Hao Sheng, Yang Zhang, and Zhang Xiong. Enhancing detection model for multiple hypothesis tracking. In *CVPRw*, 2017.

[11] Peng Chu and Haibin Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6172–6181, 2019.

[12] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. *ICCV*, 2017.

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

[14] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NIPS*, 2016.

[15] Afshin Dehghan, Yicong Tian, Philip HS Torr, and Mubarak Shah. Target identity-aware network flow for online multiple target tracking. In *CVPR*, 2015.

[16] 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.

[17] Loïc Fagot-Bouquet, Romaric Audigier, Yoann Dhome, and Frédéric Lerasle. Improving multi-frame data association with sparse representations for robust near-online multi-object tracking. In *ECCV*, 2016.

[18] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *WACV*, 2018.

[19] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 2010.

[20] Shoudong Han, Piao Huang, Hongwei Wang, En Yu, Donghaisheng Liu, Xiaofeng Pan, and Jun Zhao. Mat: Motion-aware multi-object tracking. *arXiv preprint arXiv:2009.04794*, 2020.

[21] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. In *International Conference on Machine Learning*, pages 4364–4375. PMLR, 2020.

[22] Chang Huang, Bo Wu, and Ramakant Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008.

[23] Margret Keuper, Siyu Tang, Bjorn Andres, Thomas Brox, and Bernt Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *TPAMI*, 2018.

[24] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *ICCV*, 2015.

[25] Han-Ul Kim and Chang-Su Kim. Cdt: Cooperative detection and tracking for tracing multiple objects in video sequences. In *European Conference on Computer Vision*, pages 851–867. Springer, 2016.

[26] Laura Leal-Taixé, Anton Milan, Ian Reid, Stefan Roth, and Konrad Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942*, 2015.

[27] Chao Liang, Zhipeng Zhang, Yi Lu, Xue Zhou, Bing Li, Xiyong Ye, and Jianxiao Zou. Rethinking the competition between detection and reid in multi-object tracking. *arXiv preprint arXiv:2010.12138*, 2020.

[28] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9131–9140, 2020.

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

[30] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14668–14678, 2020.

[31] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple object tracking: A literature review. *Artificial Intelligence*, page 103448, 2020.

[32] Santiago Manen, Michael Gygli, Dengxin Dai, and Luc Van Gool. Pathtrack: Fast trajectory annotation with path

supervision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 290–299, 2017.

[33] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021.

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

[35] Anton Milan, Seyed Hamid Rezatofighi, Anthony R Dick, Ian D Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *AAAI*, 2017.

[36] Anton Milan, Konrad Schindler, and Stefan Roth. Multi-target tracking by discrete-continuous energy minimization. *TPAMI*, 2016.

[37] Peter Ondruska and Ingmar Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. In *AAAI*, 2016.

[38] Bo Pang, Yizhuo Li, Yifan Zhang, Muchen Li, and Cewu Lu. Tubetk: Adopting tubes to track multi-object in a one-step training model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[39] 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 *European Conference on Computer Vision*, pages 145–161. Springer, 2020.

[40] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, 2011.

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

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

[43] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv:1701.01909*, 2017.

[44] Chaobing Shan, Chunbo Wei, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, Xiaoliang Cheng, and Kewei Liang. Fgagt: Flow-guided adaptive graph tracking. *arXiv preprint arXiv:2010.09015*, 2020.

[45] 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.

[46] Xinchu Shi, Haibin Ling, Junliang Xing, and Weiming Hu. Multi-target tracking by rank-1 tensor approximation. In *CVPR*, 2013.

[47] 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.

[48] ShiJie Sun, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah. Deep affinity network for multiple object tracking. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):104–119, 2019.

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

[50] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, 2017.

[51] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. *ICCV*, 2021.

[52] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[53] Qiang Wang, Yun Zheng, Pan Pan, and Yinghui Xu. Multiple object tracking with correlation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3876–3886, 2021.

[54] Xinchao Wang, Engin Türetken, Francois Fleuret, and Pascal Fua. Tracking interacting objects using intertwined flows. *TPAMI*, 2016.

[55] Yongxin Wang, Xinshuo Weng, and Kris Kitani. Joint detection and multi-object tracking with graph neural networks. *arXiv preprint arXiv:2006.13164*, 2020.

[56] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017.

[57] Yu Xiang, Alexandre Alahi, and Silvio Savarese. Learning to track: Online multi-object tracking by decision making. In *ICCV*, 2015.

[58] Yihong Xu, Aljosa Osep, Yutong Ban, Radu Horaud, Laura Leal-Taixé, and Xavier Alameda-Pineda. How to train your deep multi-object tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6787–6796, 2020.

[59] Fan Yang, Xin Chang, Sakriani Sakti, Yang Wu, and Satoshi Nakamura. Remot: A model-agnostic refinement for multiple object tracking. *Image and Vision Computing*, 106:104091, 2021.

[60] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *CVPR*, 2016.

[61] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *European Conference on Computer Vision*, pages 507–523. Springer, 2020.

[62] Amir Roshan Zamir, Afshin Dehghan, and Mubarak Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *ECCV*. 2012.

[63] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.

[64] Yang Zhang, Hao Sheng, Yubin Wu, Shuai Wang, Wei Ke, and Zhang Xiong. Multiplex labeling graph for near-online tracking in crowded scenes. *IEEE Internet of Things Journal*, 7(9):7892–7902, 2020.

[65] Yifu Zhang, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *arXiv e-prints*, pages arXiv–2004, 2020.

[66] Zhedong Zheng, Xiaodong Yang, Zhiding Yu, Liang Zheng, Yi Yang, and Jan Kautz. Joint discriminative and generative learning for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2138–2147, 2019.

[67] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *European Conference on Computer Vision*, pages 474–490. Springer, 2020.

[68] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *ECCV*, 2018.