TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking

Peng Chu 1
Jiang Wang 1
Quanzeng You 1
Zicheng Liu 1
Haibin Ling 2

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-
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 $l_{i}^{t-1}$ 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 $O_t = \{o^t_j\}_{j=1}^{M_t}$, 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 $l_i^{t-1}$, its matching score to an object proposal $o^t_j$ is measured by the affinity $\phi(l_i^{t-1}, o^t_j)$, 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

$$\max_{A_t=(a_{ij}^t)} \sum_{i=1}^{N_{t-1}} \sum_{j=1}^{M_t} a_{ij}^t \phi(l_i^{t-1}, o_j^t), \quad (1)$$

s.t. $\left\{ \begin{array}{l}
\sum_{j=1}^{M_t} a_{ij}^t = 1, \quad \forall i = 1, \ldots, N_{t-1} \\
\sum_{i=1}^{N_{t-1}} a_{ij}^t = 1, \quad \forall j = 1, \ldots, M_t \\
a_{ij}^t \in \{0, 1\}, \quad \forall i = 1, \ldots, N_{t-1}; \quad j = 1, \ldots, M_t
\end{array} \right. \quad (2)$

where $A_t = (a_{ij}^t)$ indicates the association between tracklets $L_{t-1} = \{l_i^{t-1}\}_{i=1}^{N_{t-1}}$ and detected candidates $O_t$. 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 $A_t = \Phi(L_{t-1}, O_t)$, where $L_{t-1}$ and $O_t$ consist of all the tracklets and candidates, respectively.

To model the spatial-temporal object correlation, we build a weighted spatial graph $\Theta_t$ for the proposals at the current frame, and a set of weighted spatial graphs $\Xi_{t-1} = \{\xi_{t-T}, \xi_{t-2}, \ldots, \xi_{t-1}\}$ 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 $A_t$ 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.

4. TransMOT

TransMOT uses the graphs $\Xi_{t-1}$ and $\Theta_t$ to learn a mapping $\Phi(\cdot)$ that models the spatial-temporal correlations, and generates an assignment/mapping matrix $A_t$. 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 $\xi_t = G(\{x_t^1\}, E_t^X, w_t^X)$ 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 $F_s \in \mathbb{R}^{N_t \times T \times D}$, 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 $F_s$ and the graph weights $w_{t-1}^X$ to generate self-attention weights for the $i$-th head:

$$F_{i,AW}^s = \text{softmax}\left[\varphi(F_s, W^Q_i, W^K_i \circ w_{t-1}^X)\right], \quad (3)$$

where $\varphi(\cdot)$ is the regular scaled dot-product to obtain attention weights as in [52], $W^Q_i, W^K_i$ 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 $F_{i,V}^s$. Combined with the attention weights in Eq. 3, the graph multi-head attention weighted feature tensor can be written as

$$F_{i,\text{att}}^s = \text{Concat}(\{F_{i,AW}^s \otimes F_{i,V}^s\}) \otimes W^O,$$

where $\{\cdot\}$ iterates and aggregates the outputs from all the attention heads, $W^O_i$ 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.

---

**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 $F_{\text{temp}}^s \in \mathbb{R}^{T \times N_t \times D}$. 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 $F_{\text{out}}^s$.

**4.2. Spatial Graph Transformer Decoder**

The spatial graph transformer decoder produces extended assignment matrix $A_t$ 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 $F_{\text{out}}^s$. The candidate graph is constructed similarly to the tracklet graphs in Sec. 4.1. Each node $o_t^j$ 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 $f_{\text{sink}} \in \mathbb{R}^D$ is added to the $\Theta_t$. 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 $f_{sink}$ is appended to the embedded feature set such that $\mathcal{F}_{sink}^t \in \mathbb{R}^{(M_t+1) \times D}$. 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 $\mathcal{F}_{att}^t \in \mathbb{R}^{(M_t+1) \times D}$.

For the tracklet embedding $\mathcal{F}_{out}^t$ 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 $\mathcal{F}_{out}^t \in \mathbb{R}^{T \times (N_{t+1}) \times D}$. 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 getting better computational efficiency. $\mathcal{F}_{att}^t$ is duplicated $N_{t+1} + 1$ times such that $\mathcal{F}_{att}^t \rightarrow \mathcal{F}_{att}^t \in \mathbb{R}^{(M_t+1) \times (N_{t+1}) \times D}$. Multi-head cross attention is calculated for $\mathcal{F}_{att}^t$ and $\mathcal{F}_{out}^t$ to generate unnormalized attention weights. The output is passed through a feed-forward layer and a normalization layer to generate the output tensor $\mathcal{R} \in \mathbb{R}^{(M_t+1) \times (N_{t+1}) \times D}$ 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 $\hat{A}_t \in \mathbb{R}^{(M_t+1) \times (N_{t+1})}$.

### 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:

$$\text{s.t. } \sum_{i, j} a_{ij} = 1, i \in [1, N_{t-1}], a_{ij} \in \{0, 1\}.$$ 

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 $T$ 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 $\hat{A}_t$ 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:

$$\mathcal{L} = -\frac{1}{N_{t-1}} \sum_{n=1}^{N_{t-1}} y_n \log(\bar{a}_n) + \lambda \frac{M_t}{M_t} \sum_{m=1}^{M_t} y_{m}^{src} \log\left(\frac{1}{1 + e^{-a_m}}\right) + \lambda \frac{M_t}{M_t} \sum_{m=1}^{M_t} (1 - y_m^{src}) \log\left(\frac{e^{-a_m}}{1 + e^{-a_m}}\right),$$

where $y$ and $y_m^{src}$ are association labels for real tracklets and the virtual source respectively, $\bar{a}_n$ is the row element of $\hat{A}_t$, $\hat{a}_{N_{t-1}+1} = \{a_m\}$, and $\lambda$ is a weighting coefficient.

### 4.4. Tracking Framework

This section describes how the final tracking association is generated from the association matrix $\hat{A}_t$ 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:

$$d_{top} = \left\| \left( u_i + \frac{w_i}{2} - u_j - \frac{w_j}{2}, v_i - v_j \right) \right\|_2 / h_i,$$

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.
Table 1. Tracking Performance on the MOT16 benchmark test set private detection track. Best in bold.

<table>
<thead>
<tr>
<th>Method</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMT [25]</td>
<td>49.2</td>
<td>44.5</td>
<td>34.7%</td>
<td>22.1%</td>
<td>8,088</td>
<td>25,335</td>
<td>684</td>
</tr>
<tr>
<td>TubeTK [36]</td>
<td>53.1</td>
<td>58.4</td>
<td>39.3%</td>
<td>18.0%</td>
<td>5,756</td>
<td>18,961</td>
<td>854</td>
</tr>
<tr>
<td>CDADDAL [13]</td>
<td>54.1</td>
<td>51.3</td>
<td>36.3%</td>
<td>22.2%</td>
<td>7,110</td>
<td>22,271</td>
<td>544</td>
</tr>
<tr>
<td>TRID [32]</td>
<td>61.0</td>
<td>55.7</td>
<td>40.6%</td>
<td>25.8%</td>
<td>6,273</td>
<td>20,611</td>
<td>351</td>
</tr>
<tr>
<td>RAR15 [18]</td>
<td>61.3</td>
<td>56.5</td>
<td>45.1%</td>
<td>14.6%</td>
<td>9,386</td>
<td>16,921</td>
<td>428</td>
</tr>
<tr>
<td>GSDT [55]</td>
<td>64.6</td>
<td>60.7</td>
<td>47.0%</td>
<td>10.5%</td>
<td>7,334</td>
<td>16,358</td>
<td>477</td>
</tr>
<tr>
<td>Fair [65]</td>
<td>67.4</td>
<td>62.6</td>
<td>47.6%</td>
<td>11.0%</td>
<td>7,854</td>
<td>15,785</td>
<td>591</td>
</tr>
<tr>
<td>TransMOT</td>
<td>66.0</td>
<td>57.0</td>
<td>64.5%</td>
<td>17.8%</td>
<td>12,454</td>
<td>13,725</td>
<td>244</td>
</tr>
</tbody>
</table>

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

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

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
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 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 crowded scenes.

### 5.3. Ablation

We study the significance of components and hyper-parameters 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 settings that the capability of TransMOT of modeling a large number of tracklets and detections in crowded scenes.

<table>
<thead>
<tr>
<th>Public Detection</th>
<th>Method</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TrckrD [58]</td>
<td>53.8</td>
<td>53.7</td>
<td>19.4%</td>
<td>36.6%</td>
<td>11,731</td>
<td>247,447</td>
<td>1,947</td>
<td></td>
</tr>
<tr>
<td>Trckr [4]</td>
<td>55.1</td>
<td>56.3</td>
<td>21.1%</td>
<td>35.5%</td>
<td>8,866</td>
<td>235,449</td>
<td>1,987</td>
<td></td>
</tr>
<tr>
<td>CTTra [67]</td>
<td>59.6</td>
<td>61.5</td>
<td>26.4%</td>
<td>31.9%</td>
<td>14,076</td>
<td>200,672</td>
<td>2,583</td>
<td></td>
</tr>
<tr>
<td>TrackFormer [33]</td>
<td>60.7</td>
<td>62.5</td>
<td>29.8%</td>
<td>26.8%</td>
<td>32,828</td>
<td>174,921</td>
<td>3,917</td>
<td></td>
</tr>
<tr>
<td>MPNTrack [8]</td>
<td>61.7</td>
<td>58.8</td>
<td>28.8%</td>
<td>33.5%</td>
<td>17,413</td>
<td>213,594</td>
<td>1,185</td>
<td></td>
</tr>
<tr>
<td>LiT [21]</td>
<td>65.6</td>
<td>60.5</td>
<td>27.0%</td>
<td>33.6%</td>
<td>14,966</td>
<td>206,619</td>
<td>1,189</td>
<td></td>
</tr>
<tr>
<td>MAT [20]</td>
<td>69.2</td>
<td>67.1</td>
<td>38.9%</td>
<td>26.4%</td>
<td>22,756</td>
<td>161,547</td>
<td>1,279</td>
<td></td>
</tr>
<tr>
<td>TransMOT-P</td>
<td>73.1</td>
<td>68.8</td>
<td>33.1%</td>
<td>31.5%</td>
<td>8,080</td>
<td>167,174</td>
<td>1,043</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Private Detection</th>
<th>Method</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAN [48]</td>
<td>49.5</td>
<td>52.4</td>
<td>21.4%</td>
<td>30.7%</td>
<td>25,423</td>
<td>234,592</td>
<td>8,431</td>
<td></td>
</tr>
<tr>
<td>TubeTK [38]</td>
<td>58.6</td>
<td>63.0</td>
<td>31.2%</td>
<td>19.9%</td>
<td>27,060</td>
<td>177,483</td>
<td>5,727</td>
<td></td>
</tr>
<tr>
<td>TransTrack [47]</td>
<td>63.9</td>
<td>74.5</td>
<td>46.8%</td>
<td>11.3%</td>
<td>28,323</td>
<td>112,137</td>
<td>3,663</td>
<td></td>
</tr>
<tr>
<td>CTTra [67]</td>
<td>64.7</td>
<td>67.8</td>
<td>34.6%</td>
<td>24.6%</td>
<td>18,498</td>
<td>160,332</td>
<td>6,102</td>
<td></td>
</tr>
<tr>
<td>Perma [51]</td>
<td>71.9</td>
<td>69.5</td>
<td>42.5%</td>
<td>17.7%</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>TLR [53]</td>
<td>73.6</td>
<td>76.5</td>
<td>47.6%</td>
<td>12.7%</td>
<td>29,808</td>
<td>99,510</td>
<td>3,369</td>
<td></td>
</tr>
<tr>
<td>TransMOT-D</td>
<td>69.3</td>
<td>68.5</td>
<td>32.2%</td>
<td>34.1%</td>
<td>22,767</td>
<td>153,156</td>
<td>1,635</td>
<td></td>
</tr>
<tr>
<td>TransMOT</td>
<td>76.3</td>
<td>76.4</td>
<td>48.7%</td>
<td>21.9%</td>
<td>31,788</td>
<td>99,651</td>
<td>1,623</td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Public Detection</th>
<th>Method</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORT* [6]</td>
<td>45.1</td>
<td>42.7</td>
<td>16.7%</td>
<td>26.2%</td>
<td>27,521</td>
<td>264,694</td>
<td>4,470</td>
<td></td>
</tr>
<tr>
<td>Trckr [4]</td>
<td>52.7</td>
<td>52.6</td>
<td>29.4%</td>
<td>26.7%</td>
<td>6,930</td>
<td>236,680</td>
<td>1,648</td>
<td></td>
</tr>
<tr>
<td>MPNTrack [8]</td>
<td>59.1</td>
<td>57.6</td>
<td>38.2%</td>
<td>22.5%</td>
<td>16,953</td>
<td>201,384</td>
<td>1,210</td>
<td></td>
</tr>
<tr>
<td>LCMOT [43]</td>
<td>62.5</td>
<td>56.3</td>
<td>34.1%</td>
<td>25.2%</td>
<td>11,726</td>
<td>213,056</td>
<td>1,562</td>
<td></td>
</tr>
<tr>
<td>TransMOT-P</td>
<td>74.3</td>
<td>73.4</td>
<td>43.7%</td>
<td>14.6%</td>
<td>11,511</td>
<td>125,029</td>
<td>1,008</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Private Detection</th>
<th>Method</th>
<th>IDF1</th>
<th>MOTA</th>
<th>MT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLT [64]</td>
<td>54.6</td>
<td>48.9</td>
<td>30.9%</td>
<td>22.1%</td>
<td>45,660</td>
<td>216,803</td>
<td>2,187</td>
<td></td>
</tr>
<tr>
<td>GSDT [55]</td>
<td>67.5</td>
<td>67.1</td>
<td>53.1%</td>
<td>13.2%</td>
<td>31,913</td>
<td>135,409</td>
<td>3,131</td>
<td></td>
</tr>
<tr>
<td>Fair [55]</td>
<td>67.3</td>
<td>61.8</td>
<td>68.8%</td>
<td>7.6%</td>
<td>103,440</td>
<td>88,901</td>
<td>5,243</td>
<td></td>
</tr>
<tr>
<td>CSTra [27]</td>
<td>68.6</td>
<td>66.6</td>
<td>50.4%</td>
<td>15.5%</td>
<td>25,404</td>
<td>144,358</td>
<td>3,196</td>
<td></td>
</tr>
<tr>
<td>ReMOT [59]</td>
<td>73.1</td>
<td>77.4</td>
<td>68.1%</td>
<td>9.9%</td>
<td>28,351</td>
<td>86,659</td>
<td>1,789</td>
<td></td>
</tr>
<tr>
<td>TransMOT</td>
<td>75.2</td>
<td>77.4</td>
<td>76.1%</td>
<td>9.2%</td>
<td>32,335</td>
<td>82,867</td>
<td>1,601</td>
<td></td>
</tr>
</tbody>
</table>

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.
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.

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

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


