Global Tracking Transformers

Xingyi Zhou[1] Tianwei Yin[1] Vladlen Koltun[2] Phillip Krähenbühl[1]
[1]The University of Texas at Austin [2]Apple

Abstract

We present a novel transformer-based architecture for global multi-object tracking. Our network takes a short sequence of frames as input and produces global trajectories for all objects. The core component is a global tracking transformer that operates on objects from all frames in the sequence. The transformer encodes object features from all frames, and uses trajectory queries to group them into trajectories. The trajectory queries are object features from a single frame and naturally produce unique trajectories. Our global tracking transformer does not require intermediate pairwise grouping or combinatorial association, and can be jointly trained with an object detector. It achieves competitive performance on the popular MOT17 benchmark, with 75.3 *MOTA and* 59.1 *HOTA. More importantly, our framework seamlessly integrates into state-of-the-art large-vocabulary detectors to track any objects. Experiments on the challenging TAO dataset show that our framework consistently improves upon baselines that are based on pairwise association, outperforming published work by a significant* 7.7 *tracking mAP. Code is available at* https://github.com/xingyizhou/GTR.

1. Introduction

Multi-object tracking aims to find and follow all objects in a video stream. It is a basic building block in application areas such as mobile robotics, where an autonomous system must traverse dynamic environments populated by other mobile agents. In recent years, *tracking-by-detection* has emerged as the dominant tracking paradigm, powered by advances in deep learning and object detection [20, 36]. Tracking-by-detection reduces tracking to two steps: detection and association. First, an object detector independently finds potential objects in each frame of the video stream. Second, an association step links detections through time. Local trackers [4, 5, 54, 55, 60, 66] primarily consider pairwise associations in a greedy way (Figure 1a). They maintain a status of each trajectory based on location [5, 68] and/or identity features [55, 66], and associate current-frame detections with each trajectory based on its last visible status. This pairwise association is efficient, but lacks an explicit model of trajectories as a whole, and sometimes struggles with heavy occlusion or strong appearance change. Global trackers [3, 6, 44, 63, 65] run offline graph-based combinatorial optimization over pairwise associations. They can resolve inconsistently grouped detections and are more robust, but can be slow and are usually detached from the detector.

(a) Local trackers

(b) Our global tracker

Figure 1. **Local trackers (top) vs. our global tracker (bottom).** Local trackers associate objects frame-by-frame, optionally with a external track status memory (not show in the figure). Our global tracker take a short video clip as input, and associates objects across all frames using global object queries.

In this work, we show how to represent global tracking (Figure 1b) as a few layers in a deep network (Figure 2). Our network directly outputs trajectories and thus sidesteps both pairwise association and graph-based optimization. We show that detectors [20, 36, 70] can be augmented by transformer layers to turn into joint detectors and trackers. Our Global TRacking transformer (GTR) encodes detections from multiple consecutive frames, and uses *trajectory queries* to group them into trajectories. The queries are detection features from a single frame (e.g., the current frame in an online tracker) after non-maximum suppression, and are transformed by the GTR into trajectories. Each trajectory query produces a single global trajectory by assigning to it a detection from each frame using a softmax distribution. The outputs of our model are thus detections and their associations through time. During training, we explicitly supervise the output of our global tracking transformer

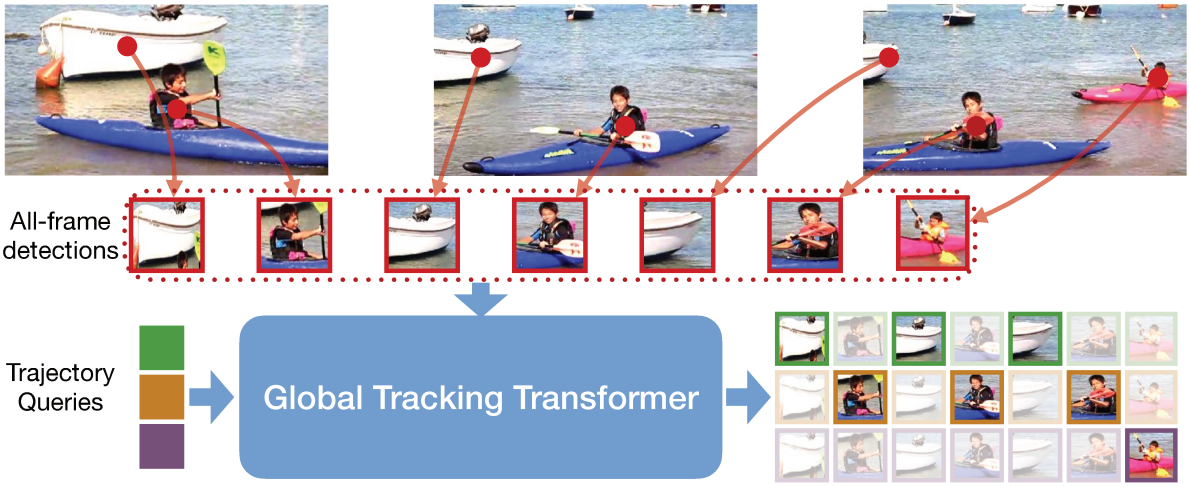

Figure 2. **Overview of our joint detection and tracking framework.** An object detector first independently detects objects in all frames. Object features are concatenated and fed into the encoder of our global Tracking transformer (GTR). The GTR additionally takes trajectory queries as decoder input, and produces association scores between each query and object. The association matrix links objects for each query. During testing, the trajectory queries are object features in the last frame. The structure of the transformer is shown in Figure 3.

using ground-truth trajectories and their image-level bounding boxes. During inference, we run GTR in a sliding window manner with a moderate temporal size of 32 frames, and link trajectories between windows online. The model is end-to-end differentiable within the temporal window.

Our framework is motivated by the recent success of transformer models [49] in computer vision in general [14, 25, 47, 67] and in object detection in particular [8, 53]. The cross-attention structure between queries and encoder features mines similarities between objects and naturally fits the association objective in multi-object tracking. We perform cross-attention between trajectory queries and object features within a temporal windows, and explicitly supervise it to produce a query-to-detections assignment. Each assignment directly corresponds to a global trajectory. Unlike transformer-based detectors [8, 30, 40, 53] that learn queries as fixed parameters, our queries come from existing detection features and adapt with the image content. Furthermore, our transformer operates on detected objects rather than raw pixels [8]. This enables us to take full advantage of well-developed object detectors [20, 69].

Our framework is end-to-end trainable, and easily integrates with state-of-the-art object detectors. On the challenging large-scale TAO dataset, our model reaches 20.1 tracking mAP on the test set, significantly outperforming published work, which achieved 12.4 tracking mAP [32]. On the MOT17 [31] benchmark, our entry achieves competitive 75.3 MOTA and 59.1 HOTA, outperforming most concurrent transformer-based trackers [30, 61, 64], and on-par with state-of-the-art association-based trackers.

2. Related work

Local multi-object tracking. Many popular trackers operate locally and greedily [4, 5, 46, 54, 55, 61, 66, 68]. They maintain a set of confirmed tracks, and link newly detected objects to tracks based on pairwise object-track distance metrics. SORT [5] and DeepSORT [55] model tracks with Kalman filters, and update the underlying locations [5] or deep features [55] in every step. Tracktor [4] feeds tracks to a detector as proposals, and directly propagates the tracking ID. CenterTrack [68] conditions detection on existing tracks, and associates objects using their predicted locations. TransCenter [61] builds upon CenterTrack by incorporating deformable DETR [72]. JDE [54] and FairMOT [66] train the detector together with an instance-classification branch, and associate via pairwise ReID features similar to SORT [5]. STRN [60] learns a dedicated association feature considering the spatial and temporal cues, but again performs pairwise association. In contrast, we do not rely on pairwise association, but instead associates to all objects across the full temporal window via a transformer.

Global tracking. Traditional trackers first detect objects offline, and consider object association across all frames as a combinatorial optimization problem [6, 12, 34, 44, 65]. Zhang et al. [65] formulate tracking as a min-cost max-flow problem over a graph, where nodes are detections and edges are valid associations. MPN [6] simplifies the graph construction and proposes a neural solver that performs the graph optimization. LPC [12] additionally considers a classification module on the graph. Lif_T [44] incorporates person ReID and pose features in the graph optimization. These methods are still based on pairwise associations and

use the combinatorial optimization to select globally consistent assignments. Our method directly outputs consistent long-term trajectories without combinatorial optimization. This is done by a single forward pass in a relatively shallow network.

Transformers in tracking. Trackformer [30] augments DETR [8] with additional object queries from existing tracks, and propagates track IDs as in Tracktor [4]. TransTrack [40] uses features from historical tracks as queries, but associates objects based on updated bounding box locations. MOTR [64] follows the DETR [8] structure and iteratively propagates and updates track queries to associate object identities. MO3TR [71] additionally uses a temporal attention module to update the status of each track over a temporal window, and feeds updated track features as queries in DETR. The common idea behind these works is to use the object query mechanism in DETR [8] to extend existing tracks *frame-by-frame*. We use transformers in a different way. Our transformer uses queries to generate entire trajectories at once. Our queries do not generate new boxes, but group already-detected boxes into trajectories.

Video object detection. Applying attention blocks on object features over a video is a successful idea in video object detection [37]. SELSA [57] feeds region proposals of randomly sampled frames to a self-attention block to provide global context. MEGA [9] builds a hierarchical local and global attention mechanism with a large temporal receptive field. ContextRCNN [2] uses an offline long-term feature bank [56] to integrate long-range temporal features. These methods support our idea of using transformers to analyze object relations. The key difference is they do not use object identity information, but implicitly use object correlations to improve detection. We explicitly learn object association in a supervised way for tracking.

3. Preliminaries

We start by formally defining object detection, tracking, and tracking by detection.

Object detection. Let I be an image. The goal of object detection is to identify and localize all objects. An object detector [8,36,45,70] takes the image I as input and produces a set of objects $\{p_i\}$ with locations $\{b_i\}, b_i \in \mathbb{R}^4$ as its output. For multi-class object detection, a second stage [20,36] takes the object features and produce a classification score $s_i \in \mathbb{R}^C$ from a set of predefined classes C and a refined location $\tilde{b}_i$. For single-class detection (e.g., pedestrian detection [31]), the second stage can me omitted.

Tracking. Let $I^1, I^2, \ldots, I^T$ be a series of images. The goal of a tracker is to find trajectories $\boldsymbol{\tau}_1, \boldsymbol{\tau}_2, \ldots, \boldsymbol{\tau}_K$ of all objects over time. Each trajectory $\boldsymbol{\tau}_k = [\tau_k^1, \ldots, \tau_k^T]$ describes a tube of object locations $\tau_k^t \in \mathbb{R}^4 \cup \{\emptyset\}$ through time t. $\tau_k^t = \emptyset$ indicates that the object k cannot be located in frame t. The tracker may optionally predict the object class score s_k [13] for each trajectory, usually as the average class of its per-frame slices.

Tracking by detection decomposes the tracking problem into per-frame detection and inter-frame object association. Object detection first finds N_t candidate objects $b_1^t, b_2^t, \ldots$ as bounding boxes $b_i^t \in \mathbb{R}^4$ in each frame I^t. Association then links existing tracks $\boldsymbol{\tau}_k$ to current detected objects using an object indicator $\alpha_k^t \in \{\emptyset, 1, 2, \ldots, N_t\}$ at each frame t :

$$\tau_k^t = \begin{cases} \emptyset & \text{if } \alpha_k^t = \emptyset \\ b^t_{\alpha_k^t} & \text{otherwise} \end{cases}$$

Most prior works define the association greedily through pairwise matches between objects in adjacent or nearby frames [4, 5, 66, 68], or rely on offline combinatorial optimization for global association [6, 16, 65]. In this work, we show how to perform joint detection and global association within a single forward pass through a network. The network learns global tracking within a video clip of 32 frames in an end-to-end fashion. We leverage a probabilistic formulation of the association problem and show how to instantiate tracking in a transformer architecture in Section 4.

4. Global tracking transformers

Out global tracking transformer (GTR) associates objects in a probabilistic and differentiable manner. It links objects p_i^t in each frame I^t to a set of trajectory queries q_k. Each trajectory query q_k produces an object association score vector $g \in \mathbb{R}^N$ over objects from all frames. This association score vector then yields a per-frame object-level association $\alpha_k^t \in \{\emptyset, 1, \ldots, N_t\}$, where $\alpha_k^t = \emptyset$ indicates no association and N_t is the number of detected objects in frame I^t. The combination of associations then produces a trajectory $\boldsymbol{\tau}_k$. Figure 2 provides an overview. The association step is differentiable and can be jointly trained with the underlying object detector.

4.1. Tracking transformers

Let $p_1^t, \ldots, p_{N_t}^t$ be a set of high-confidence objects for image I^t. Let $B^t = \{b_1^t, \ldots, b_{N_t}^t\}$ be their corresponding bounding boxes. Let $f_i^t \in \mathbb{R}^D$ be the D-dimentional features extracted from boxes b_i^t. For convenience let $F^t = \{f_1^t, \ldots, f_{N_t}^t\}$ be the set of all detection features of image I^t, and $F = F^1 \cup \ldots \cup F^T$ be the set of all features through time. The collection of all object features $F \in \mathbb{R}^{N \times D}$ is the input to our tracking transformer, where $N = \sum_t^T N_t$ is the total number of detections in all frames. The tracking transformer takes features F and a trajectory query $q_k \in \mathbb{R}^D$, and produces a trajectory-specific association score $g(q_k, F) \in \mathbb{R}^N$.

Formally, let $g_i^t(q_k, F) \in \mathbb{R}$ be the score of the i-th object in the t-th frame. A special output token $g_\emptyset^t(q_k, F) = 0$ indicates no association at time t. The tracking transformer

then predicts a distribution of associations over all objects i in frame I^t for each trajectory k. We model this as an independent softmax activation for each time-step t:

$$P_A(\alpha^t = i|q_k, F) = \frac{\exp\left(g_i^t(q_k, F)\right)}{\sum_{j\in\{\emptyset,1,\dots N_t\}} \exp\left(g_j^t(q_k, F)\right)} \quad (1)$$

Since a detector produces a single bounding box b_i^t for each object p_i^t, there is a one-to-one mapping between the association distribution P_A and a distribution P_t over bounding boxes for trajectory k at time t: $P_t(b|q_k, F) = \sum_{i=1}^{N_t} 1_{[b=b_i^t]} P_A(\alpha^t = i|q_k, F)$, where the indicator $1_{[\cdot]}$ assigns an output bounding box to each associated query. In practice, a detector's non-maximum suppression (NMS) ensures that there is also a unique mapping from P_t back to P_A. The distribution over bounding boxes in turn leads to a distribution over entire trajectories $P_T(\boldsymbol{\tau}|q_k, F) = \prod_{t=1}^{T} P_t(\tau^t|q_k, F)$.

During training, we maximize the log-likelihood of the ground-truth trajectories. During inference, we use the likelihood to produce long-term tracks in an online manner.

4.2. Training

Given a set of ground-truth trajectories $\hat{\boldsymbol{\tau}}_1, \dots, \hat{\boldsymbol{\tau}}_K$, our goal is to learn a tracking transformer that estimates P_A, and implicitly the trajectory distribution P_T. We jointly train the tracking transformer with detection by treating the transformer as an RoI head like two-stage detectors [36]. At each training iteration, we first obtain high-confidence objects $b_1^t, \dots, b_{N_t}^t$ and their corresponding features F_t after non-maximum suppression. We then maximize $\log P_T(\boldsymbol{\tau}|q_k, F)$ for each ground-truth trajectory $\boldsymbol{\tau}$. This is equivalent to maximizing $\log P_A(\alpha^t|q_k, F)$ after assigning $\boldsymbol{\tau}$ to a set of objects. We follow object detection and use a simple intersection-over-union (IoU) assignment rule:

$$\hat{\alpha}_k^t = \begin{cases} \emptyset, & \text{if } \tau_k^t = \emptyset \text{ or } \max_i IoU(b_i^t, \tau_k^t) < 0.5 \\ \text{argmax}_i IoU(b_i^t, \tau_k^t), & \text{otherwise} \end{cases} \quad (2)$$

We use this assignment to both train the bounding box regression of the underlying two-stage detector, and our assignment likelihood P_A. However, this assignment likelihood further depends on the trajectory query q_k, which we define next.

Trajectory queries are key to our formulation. Each query q_k generates a trajectory. In prior work [8], object queries were learned as network parameters and fixed during inference. This makes queries image-agnostic and requires a near-exhaustive enumeration of them [26, 42, 72]. For objects this is feasible [8], as anchors [23] or proposals [41] showed. Trajectories, however, live in the exponentially larger space of potential moving objects than simple boxes, and hence require many more queries to cover that space. Furthermore, tracking datasets feature many fewer annotated instances, and learned trajectories easily overfit and remember the dataset.

We instead directly use object features f_i^t as the object queries. Specifically, let $\hat{\alpha}_k$ be the matched objects for a ground-truth trajectory $\boldsymbol{\tau}_k$ according to Equation (2). Any feature $\{f_{\hat{\alpha}_k^1}^1, f_{\hat{\alpha}_k^2}^2, \dots\}$ can serve as the trajectory query for trajectory $\boldsymbol{\tau}_k$. In practice, we use all object features F in the all the T frames as queries and train the transformer for a sequence length of T. Any unmatched features f_i^t are used as background queries and supervised to produce $\emptyset$ for all frames. We allow multiple queries to produce the same trajectory, and do not require a one-to-one match [8]. During inference, we only use object features from one single frame as the queries to avoid duplicate outputs. All object features within a frame are different (after standard detection NMS) and hence produce different trajectories.

Training objective. The overall training objective combines the assignment in Equation (2) and trajectory queries to maximize the log-likelihood of each trajectory under its assigned queries. For each trajectory τ_k we optimize the log-likelihood of its assignments $\hat{\alpha}_k$:

$$\ell_{asso}(F, \hat{\boldsymbol{\tau}}_k) = -\sum_{s\in\{1\dots T|\hat{\alpha}_k^s\neq\emptyset\}} \sum_{t=1}^{T} \log P_A(\hat{\alpha}_k^t|F_{\hat{\alpha}_k^s}^s, F) \quad (3)$$

For any unassociated features, we produce empty trajectories:

$$\ell_{bg}(F) = -\sum_{s=1}^{T} \sum_{j:\nexists\hat{\alpha}_k^s=j} \sum_{t=1}^{T} \log P_A(\alpha^t = \emptyset|F_j^s, F) \quad (4)$$

The final loss simply combines the two terms:

$$L_{asso}(F, \{\hat{\boldsymbol{\tau}}_1, \dots, \hat{\boldsymbol{\tau}}_K\}) = \ell_{bg}(F) + \sum_{\hat{\boldsymbol{\tau}}_k} \ell_{asso}(F, \hat{\boldsymbol{\tau}}_k) \quad (5)$$

We train L_{asso} jointly with standard detection losses [70], including classification and bounding-box regression losses, and optionally second stage classification and regression losses for multi-class tracking [13],

4.3. Online Inference

During inference, we process the video stream online in a sliding-window manner with window size $T = 32$ and stride 1. For each individual frame t, we feed the image to the network before the tracking transformer and obtain N_t bounding boxes B^t and object features F^t. We keep a temporal history buffer of T frames, i.e., $B = \{B^{t-T+1}, \cdots, B^t\}$ and $F = \{F^{t-T+1}, \cdots, F^t\}$, and run the tracking transformer for each sliding window. We use object features from the current frame t as trajectory queries $q_k = F_k^t$ to produce N_t trajectories. For the first frame,

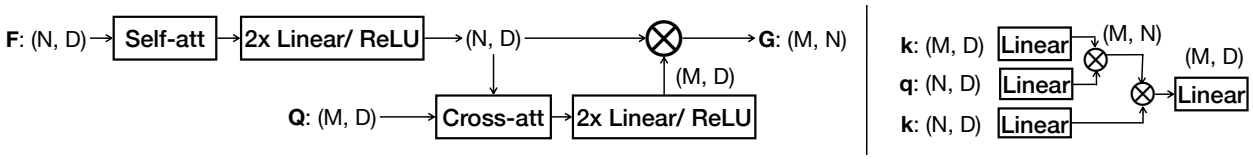

Figure 3. Left: detailed network architecture of GTR. Right: detailed structure of both self-att and cross-att blocks. We omit multi-head [49] in the figure for simplicity. For self-attention, $q = k = F$. For cross attention, $q = Q$, $k = F$. We list data dimensionalities in parentheses. $\otimes$ indicates matrix multiplication (transpose when needed).

we initialize all detections as trajectories. For any subsequent frame, we link current predicted trajectories to existing tracks using the average assignment likelihood P_A as a distance metric. Since the current trajectories share up to $T-1$ boxes and features with past trajectories, the overlap can be quite large. We use a Hungarian algorithm to ensure that the mapping from current long-term trajectories to existing tracks is unique. If the average association score with any prior trajectory is lower than a threshold θ, we start a new track. Otherwise we append the underlying current detection (query) that generates the trajectory to the matched existing track.

4.4. Network architecture

The global tracking transformer takes a stack of object features $F \in \mathbb{R}^{N \times D}$ as the encoder input, a matrix of queries $Q \in \mathbb{R}^{M \times D}$ as the decoder input, and produces an association matrix $G \in \mathbb{R}^{M \times N}$ between queries and objects. The detailed structure of the tracking transformer is shown in Figure 3 (left). It follows DETR [8] but only uses a one-layer encoder and a one-layer decoder. Empirically, we observe that self-attention for queries and Layer Normalization [1] were not required. See Section 5.5 for an ablation. The resulting network structure is lightweight, with 10 linear layers in total. It runs in a fraction of the runtime of the backbone detector, even for hundreds of queries.

4.5. Connection to embedding learning and ReID

Consider a variation of GTR with just a dot-product association score $g_i^t(q_k, F) = q_k \cdot F_i^t$. Further consider learning all trajectory queries $Q = \{q_1, \ldots, q_k\}$ as free parameters, one per training trajectory τ_k. In this variation, the softmax assignment in Equation (1) reduces to a classification problem. For each object feature, we classify it as a specific training instance or as background. This is exactly the objective of classification-based embedding learning in person-ReID [29], as used in ReID-based trackers [54, 66].

The two key differences between embedding learning and GTR are: first, our transformer does not assume any factorization of g_i^t and allows the model to reason about all boxes at once when computing associations. A dot-product-based ReID network on the other hand assumes that all boxes independently produce a compatible embedding. See Section 5.5 for an ablation of this transformer structure. Second, our trajectory queries are not learned. This allows our transformer to produce long-term associations in a single forward pass, while ReID-based trackers rely on a separate cosine-distance-based grouping step [54, 66].

5. Experiments

We evaluate our method on two tracking benchmarks: TAO [13] and MOT17 [31].

TAO [13] tracks a wide variety of objects. The images are adopted from 6 existing video datasets, including indoor, outdoor, and driving scenes. The dataset requires tracking objects with a large vocabulary of 488 classes in a long-tail setting. It contains 0.5k, 1k, and 1.5k videos for training, validation, and testing, respectively. Each video contains $\sim$ 40 annotated frames at 1 annotated frame per second. There is significant motion between adjacent annotated frames. The training annotations are incomplete. We thus do not use the training set and solely train on LVIS [19] and use the TAO validation and test set for evaluation.

MOT [31] tracks pedestrians in crowd scenes. It contains 7 training sequences and 7 test sequences. The sequences contain 500 to 1500 frames, recorded and annotated at 25-30 FPS. We follow CenterTrack [68] and split each training sequence in half. We use the first half for training and the second half for validation. We perform ablation studies mainly on this validation set, and compare to other approaches on the official hidden test set. We evaluate under the private detection protocol.

5.1. Evaluation metrics

We evaluate under the official metrics for each dataset. TAO [13] uses tracking mAP@0.5 as the official metric, which is based on standard object detection mAP [24] but changes the 2D bounding box IoU to 3D temporal-spatial IoU between the predicted trajectory and the ground-truth trajectory. The overall tracking mAP is averaged across all classes. MOT [31] uses Multi-Object Tracking Accuracy (MOTA) as the official metric. MOTA $= 1 - \frac{\sum_t (FP_t + FN_t + IDSW_t)}{\sum_t GT_t}$, where GT_t is the number of ground truth objects in frame t, and FP_t, FN_t, and $IDSW_t$ measure the errors of false positives, false negatives, and ID switches, respectively.

As suggested by the MOT benchmark, we additionally report HOTA, a new tracking metric [28]. HOTA is defined as the geometric mean of detection accuracy (DetA) and association accuracy (AssA). Both DetA and AssA have the form $\frac{|TP|}{|TP|+|FN|+|FP|}$, with their respective true/false cri-

teria. In our experiments, we mainly use AssA to access tracking performance.

5.2. Training and inference details

TAO training. Our implementation is based on detectron2 [59]. For TAO [13] experiments, we use Res2Net [17] with deformable convolution [11] as the backbone. We adopt CenterNet2 [69] as the detector, which uses CenterNet [70] as proposal network and cascaded RoI heads [7] for classification. Following the guideline of TAO dataset [13], we train the object detector on the combination of LVISv1 [19] and COCO [24]. We additionally incorporate a federated loss [69] to improve long-tail detection. We first train a single-frame detector. The training uses SGD with learning rate 0.04 and batch size 32 for 180K iterations (the $4\times$ schedule [59]). We use training resolution 896×896 following the scale-and-crop augmentation of EfficientDet [43]. The detector yields 37.1 mAP on the LVISv1 validation set and 27.3 mAP on the TAO validation set.

TAO only provides a small training set for tuning tracking hyperparameters, but not for training the tracker. We empirically observed that training on the TAO training set hurts detection performance, and overall does not yield good tracking accuracy. We find that training only on static image datasets [19] with data augmentation is sufficient for tracking. Our training strategy follows CenterTrack [68]. Specifically, we apply two different data augmentations to an image, and use them as the starting and ending frame of a video. We then interpolate the images and annotations linearly to generate a smooth video for training.

With the synthetic video, we fine-tune the network with the tracking transformer head end-to-end from the single-frame detector. Our fine-tuning protocol follows DETR [8] and uses the AdamW optimizer [27], multiplies the backbone learning rate by a factor of 0.1, and clamps the gradient norm at 0.1. We use a base learning rate of 0.0001. We generate video clips of length $T = 8$ and train with a batch size of 8 videos on 8 GPUs, resulting in an effective batch size of 64. We fine-tune the network for 22,500 iterations (a $2\times$ schedule). The fine-tuning takes around 8 hours on 8 Quadro RTX 6000 GPUs.

MOT training. For our MOT model, we follow past works [66, 68] to use CenterNet [70] with a DLA-34 backbone [62] as the object detector. We use BiFPN [43] as upsampling layers instead of the original deformable-convolution-based [11] upsampling [62]. We use RoIAlign [20] to extract features for our global tracking transformer. We do not refine bounding boxes from the RoI feature and use the CenterNet detections as-is. We use a training size of 1280×1280 and a test size of 1560 (longer edge). Following CenterTrack [68], we pretrain the detector on Crowdhuman [39] for 96 epochs. We then fine-tune with the GTR head on Crowdhuman (with augmentation) and the MOT training set in a $1:1$ ratio [40]. We again use $T = 8$ frames for a video clip with a batch size of 8 clips. We fine-tune the network for 32K iterations, which corresponds to $\sim$36 epochs of Crowdhuman and 64 epochs of MOT. This takes $\sim$6 hours on 8 Quadro RTX 6000 GPUs.

Inference. During testing, we set the output score threshold to 0.55 for MOT and the proposal score threshold to 0.4 for TAO, based on a sweep on the validation set. We do not set an output threshold for TAO. For both datasets, we set the new-track association threshold to $\theta = 0.2$. Since the MOT dataset has high frame rate, we find it beneficial to use location information during association. We associate tracks based on the maximum of the trajectory association score and the box-trajectory IoU. This is examined in a controlled experiment in Section 5.5. We further remove trajectories [6] of length < 5.

Tracking-conditioned classification. Our global association module is applied to object features before classification. This allows us to to classify objects using temporal cues from tracking. On our TAO experiments, we assign a single classification score to the trajectory by averaging the per-box classification scores within the trajectory to a global classification score offline.

Runtime. We measure the runtime on our machine with an Intel Core i7-8086K CPU and a Titan Xp GPU. On the MOT17 our backbone detector runs in 47ms and the tracking transformer in 4ms per frame. On TAO the backbone runs in 86ms, the transformer in 3ms.

5.3. Global versus local association

We first validate our main contribution: global association. We compare to baseline local trackers based on location (SORT [5]), identity, or joint location and identity (FairMOT [66]). To make a direct comparison of trackers, we apply all baseline trackers to the detection outputs of the same model to ensure the same detections (Rows 1-3 in Table 1). The ReID features are trained with our association loss (see discussions in Section 4.5), we also include baselines with the original instance-classification-based ReID losses (row 4 in Table 1). We adopt the implementation from FairMOT [66] with the default hyperparameters[1] and tricks, including a track-rebirth mechanism for up to 30 frames, for all baselines.

Table 1 shows the results on the TAO [13] and MOT17 [31] validation sets. First, despite close MOTA and DetA, ReID-based methods (FairMOT [66] and ours) generally achieve higher tracking accuracy than the location-only baseline [5]. For our method, when $T = 2$ it reduces to a local tracker that only associates across consecutive pairs of frames. This tracker cannot recover from any occlusion or missing detection, yielding a relatively low AssA. However, when we gradually increase the temporal window T,

[1] We tuned the hyperparameters, but observed that the default settings performed best.

#		TAO				MOT17				
		Track mAP	HOTA	DetA	AssA	MOTA	IDF1	HOTA	DetA	AssA
1	IoU [5]	8.8	32.7	30.5	35.4	68.9	65.0	57.4	59.2	56.1
2	ReID	10.9	35.0	31.4	39.5	70.9	74.0	61.7	60.0	63.7
3	IoU+ReID [66]	11.0	34.9	31.2	39.5	71.1	74.2	62.1	60.2	64.4
4	IoU+ReID (retrained)	6.7	23.4	18.8	29.5	69.9	73.0	60.9	59.4	62.5
5	GTR (T=2)	13.6	42.0	35.8	49.8	71.3	65.1	57.8	60.6	55.8
6	GTR (T=4)	17.7	44.0	36.4	53.6	**71.6**	69.6	59.9	**60.8**	59.6
7	GTR (T=8)	19.5	45.6	36.8	56.8	71.3	72.2	61.1	60.7	62.0
8	GTR (T=16)	**22.5**	**45.8**	**36.8**	**57.4**	71.4	75.1	62.5	60.6	65.0
9	GTR (T=32)	22.1	44.9	35.9	56.7	71.3	**75.9**	**63.0**	60.4	**66.2**

Table 1. **Effectiveness of global tracking.** We compare greedy trackers [5, 66] (top block) with our global tracker(GTR) under different temporal windows on the TAO and MOT17 validation sets. We show the official metrics (Track mAP for TAO and MOTA/ IDF1 for MOT17) as well as HOTA metrics. All metrics are higher better. All rows except row 4 are the same model trained with our losses (evaluated with different tracking algorithms). Row 4 is a different model retrained with the original instance-classification loss [66]. Our global tracker benefits from longer temporal windows, and outperforms local trackers.

	Validation				Test	
	mAP50	HOTA	DetA	AssA	mAP50	FPS
SORT_TAO [13]	13.2	-	-	-	10.2	**15.2**
QDTrack [32]	16.1	35.8	24.3	53.5	12.4	5.4
GTR w. QDTrack det.	20.4	40.7	30.1	55.6	-	-
GTR	**22.5**	**45.8**	**36.8**	**57.5**	**20.1**	11.2
AOA [15]	25.8	-	-	-	27.5	1.0

Table 2. **Results on TAO dataset [13].** We show the HOTA metrics on the validation set and the official metric tracking mAP50. We show the frame-per-second tested on our machine in the last column. We show the 2020 TAO challenge winner which are based on a separate ReID network for *per-box* in the last row.

we observe a consistent increase in association accuracy. On MOT17 with $T = 32$, our method outperforms FairMOT [66] by a healthy 1.8 AssA and 1.7 IDF1, showing the advantage of our global tracking formulation. On TAO the performance saturates at $T = 16$. This may due to the much lower frame-rate in TAO dataset which results in drastic layout change within a long temporal window.

5.4. Comparison to the state-of-the-art

Next we compare to other trackers with different detections on the corresponding test sets. Table 2 shows the results on TAO validation and test sets. TAO [13] is a relatively new benchmark with few public entries [13, 32]. Our method substantially outperforms the official SORT baseline [13] and the prior best result (QDTrack [32]), yielding a relative improvement of 62% in mAP on the test set. While part of the gain is from our stronger detector, this highlights one of the advantages of our model: it is end-to-end jointly trainable with state-of-the-art detection systems. Table 2 3rd row show GTR with the detections from QDTrack [32]. We show GTR displays a 4.3 mAP and 1.9 AssA gain over QDTrack using the same detector.

Our model underperforms the 2020 TAO Challenge winner AOA [15], which trains separate ReID networks on large single-object tracking datasets [21, 35, 37]. They feed all detected boxes separately to the ReID networks in a slow-RCNN [18] fashion. On our machine, AOA's full detection and tracking pipeline takes 989ms per image on average. Our model is more than $10\times$ faster than AOA [15], and uses a single forward pass for each frame with a light-weighted per-object head.

Table 3 compares our tracker with other entries on the MOT17 leaderboard. Our entry achieves a 74.1 MOTA, 71.1 IDF1, and 59.0 HOTA. This is better than most concurrent transformer-based trackers, including Trackformer [30], MOTR [64], TransCenter [61], and TransTrack [40]. Our model currently underperforms TransMOT [10] in both MOTA and IDF1. There are several implementation differences between TransMOT and ours, including the use of additional data (TransMOT uses additional ReID data), detector architecture (TransMOT uses YOLOv5 [48] as detector and uses a separate tracker), and training and testing parameters (Code not released). Our tracker is 1.4 MOTA lower, but runs $2\times$ faster.

5.5. Design choice experiments

Here we ablate our key design choices. All experiments are conducted under the best setting of Table 1, with $T = 32$. The random noise across different runs is within 0.2 MOTA and 0.5 AssA.

Attention structure. We first verify the necessity of using a transformer structure for the association head. As the counterpart, we remove both the self-attention layer and the cross-attention layer in Figure 3, and directly dot-product the object features after the linear layers. Table 4a shows that this decreases AssA considerably. Further adding self-attention layers in the decoder as in DETR [8] does not improve performance, thus we just use encoder attention.

Positional embedding. Positional embedding is a com-

	MOTA↑	IDF1↑	HOTA↑	DetA↑	AssA↑	FP↓	FN↓	IDS↓	FPS ↑
Trackformer [30]	65.0	63.9	-	-	-	70,443	123,552	3,528	-
MOTR [64]	65.1	66.4	-	-	-	45,486	149,307	**2,049**	-
ChainedTracker [33]	66.6	57.4	49.0	53.6	45.2	22,284	160,491	5,529	6.8
CenterTrack [68]	67.8	64.7	52.2	53.8	51.0	**18,498**	160,332	3,039	17.5
QDTrack [32]	68.7	66.3	53.9	55.6	52.7	26,589	146,643	3,378	20.3
TraDeS [58]	69.1	63.9	52.7	55.2	50.8	20,892	150,060	3,555	**66.9**
TransCenter [61]	73.2	62.2	54.5	60.1	49.7	23,112	123,738	4,614	1.0
GSDT [52]	73.2	66.5	55.2	60.0	51.0	26,397	120,666	3,891	4.9
FairMOT [66]	73.7	72.3	59.3	60.9	58.0	27,507	117,477	3,303	25.9
TransTrack [40]	74.5	63.9	53.9	60.5	48.3	28,323	112,137	3,663	59.2
CSTrack [22]	74.9	72.6	59.3	61.1	57.9	23,847	114,303	3,567	15.8
FUFET [38]	76.2	68.0	57.9	**62.9**	53.6	32,796	98,475	3,237	6.8
CorrTracker [50]	76.5	73.6	**60.7**	62.8	**58.9**	29,808	99,510	3,369	15.6
TransMOT [10]	**76.7**	**75.1**	-	-	-	36,231	**93,150**	2,346	9.6
GTR (ours)	75.3	71.5	59.1	61.6	57.0	26,793	109,854	2,859	19.6

Table 3. **Comparison to the state-of-the-art on the MOT17 test set (private detection).** We show the official metrics from the leaderboard. ↑: higher better and ↓: lower better. FPS is taken from the leaderboard or paper. GTR achieves top-tier performance on MOT17.

	HOTA	DetA	AssA	MOTA
Direct dot product	61.3	59.5	63.6	70.5
*Encoder attention	63.0	60.4	66.2	71.3
Enc.+ Dec. attention	62.3	60.5	64.5	71.2

(a) **With/without attention layers.** Encoder attention improves tracking.

	HOTA	DetA	AssA	MOTA
*no embedding	63.0	60.4	66.2	71.3
w. positional emb.	62.5	60.7	65.0	71.7
w.pos.+ temp. emb.	62.4	60.7	64.6	71.7

(b) **Different positional/temporal embeddings.** Possitional embeddings do not help.

Enc.	Dec.	HOTA	DetA	AssA	MOTA
*1	1	63.0	60.4	66.2	71.3
1	2	62.7	60.4	65.0	71.2
2	1	63.0	60.9	66.0	71.7

(c) **Number of transformer layers.** One layer is sufficient for both.

	MOT17				TAO
	HOTA	DetA	AssA	MOTA	mAP50
w/o location	61.7	60.6	63.3	71.3	22.5
*w/ location	63.0	60.4	66.2	71.3	22.5

(d) **With/without using location during testing.** Location helps MOT17 but not TAO.

Table 4. **Design choice experiments on the MOT17 validation set.** * means our default setting. We ablate the effectiveness of attention layers, effectiveness of positional embeddings, number of transformer layers, and use of localizations in testing.

mon component in transformers. We have implemented a learned positional embedding as well as a learned temporal embedding. However, we didn't observe an improvement in association accuracy from these, as shown in Table 4b. We thus don't use any positional embedding in our final model.

Transformer layers. Table 4c shows the results of using different numbers of attention layers in the encoder and decoder. While most transformer-based trackers [30, 40] require 6 encoder and decoder layers, we observe that 1 layer in each is sufficient in our model. One possible reason is that other trackers take pixel features as input, while we use detected object features, which makes the task easier.

Using location in testing. As described in Section 4.3, we combine the trajectory probability and location-based IoU during inference. Table 4d examines this choice. On MOT17, using location improves AssA by 3, due to the high frame-rate on the dataset. On TAO, where frame-rate is low, using our predicted association alone works fine.

6. Conclusion

We presented a framework for joint object detection and tracking. The key component is a global tracking transformer that takes object features from all frames within a temporal window and groups objects into trajectories. Our model performs competitively on the MOT17 and TAO benchmarks. We hope that our work will contribute to robust and general object tracking in the wild.

Limitations. Currently, we use a temporal window size of 32 due to GPU memory limits, and rely on a sliding windows inference to aggregate identities across larger temporal extents. It can not recover from missing detection or occlusions larger than 32 frames. In addition, our TAO model is currently trained only on static images, due to a lack of publicly available multi-class multi-object tracking training sets. Training our model on emerging large-scale datasets such as UVO [51] is an exciting next step.

Acknowledgments. This material is based upon work supported by the National Science Foundation under Grant No. IIS-1845485 and IIS-2006820. Xingyi is supported by a Facebook Fellowship.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv:1607.06450*, 2016. 5

[2] Sara Beery, Guanhang Wu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Context r-cnn: Long term temporal context for per-camera object detection. In *CVPR*, 2020. 3

[3] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *TPAMI*, 2011. 1

[4] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *ICCV*, 2019. 1, 2, 3

[5] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 1, 2, 3, 6, 7

[6] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *CVPR*, 2020. 1, 2, 3, 6

[7] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 6

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2, 3, 4, 5, 6, 7

[9] Yihong Chen, Yue Cao, Han Hu, and Liwei Wang. Memory enhanced global-local aggregation for video object detection. In *CVPR*, 2020. 3

[10] Peng Chu, Jiang Wang, Quanzeng You, Haibin Ling, and Zicheng Liu. Transmot: Spatial-temporal graph transformer for multiple object tracking. *arXiv:2104.00194*, 2021. 7, 8

[11] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017. 6

[12] Peng Dai, Renliang Weng, Wongun Choi, Changshui Zhang, Zhangping He, and Wei Ding. Learning a proposal classifier for multiple object tracking. *CVPR*, 2021. 2

[13] Achal Dave, Tarasha Khurana, Pavel Tokmakov, Cordelia Schmid, and Deva Ramanan. Tao: A large-scale benchmark for tracking any object. In *ECCV*, 2020. 3, 4, 5, 6, 7

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[15] Fei Du, Bo Xu, Jiasheng Tang, Yuqi Zhang, Fan Wang, and Hao Li. 1st place solution to ECCV-TAO-2020: Detect and represent any object for tracking. *arXiv:2101.08040*, 2021. 7

[16] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3D tracking by detection. In *ICRA*, 2018. 3

[17] Shanghua Gao, Ming-Ming Cheng, Kai Zhao, Xin-Yu Zhang, Ming-Hsuan Yang, and Philip HS Torr. Res2net: A new multi-scale backbone architecture. *TPAMI*, 2019. 6

[18] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 7

[19] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 5, 6

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1, 2, 3, 6

[21] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *TPAMI*, 2019. 7

[22] 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:2010.12138*, 2020. 8

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 4

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 5, 6

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv:2103.14030*, 2021. 2

[26] Ze Liu, Zheng Zhang, Yue Cao, Han Hu, and Xin Tong. Group-free 3D object detection via transformers. *arXiv:2104.00678*, 2021. 4

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2017. 6

[28] Jonathon Luiten, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. Hota: A higher order metric for evaluating multi-object tracking. *IJCV*, 2021. 5

[29] Hao Luo, Youzhi Gu, Xingyu Liao, Shenqi Lai, and Wei Jiang. Bag of tricks and a strong baseline for deep person re-identification. In *CVPR Workshops*, 2019. 5

[30] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv:2101.02702*. 2, 3, 7, 8

[31] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831*, 2016. 2, 3, 5, 6

[32] Jiangmiao Pang, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, 2021. 2, 7, 8

[33] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *ECCV*, 2020. 8

[34] Jinlong Peng, Tao Wang, Weiyao Lin, Jian Wang, John See, Shilei Wen, and Erui Ding. Tpm: Multiple object tracking with tracklet-plane matching. *Pattern Recognition*, 2020. 2

[35] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *CVPR*, 2017. 7

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 3, 4

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 3, 7

[38] Chaobing Shan, Chunbo Wei, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, Xiaoliang Cheng, and Kewei Liang. Tracklets predicting based adaptive graph tracking. *arXiv:2010.09015*, 2020. 8

[39] 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:1805.00123*, 2018. 6

[40] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple-object tracking with transformer. *arXiv:2012.15460*, 2020. 2, 3, 6, 7, 8

[41] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse r-cnn: End-to-end object detection with learnable proposals. *CVPR*, 2021. 4

[42] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris Kitani. Rethinking transformer-based set prediction for object detection. *arXiv:2011.10881*, 2020. 4

[43] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020. 6

[44] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *CVPR*, 2017. 1, 2

[45] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *ICCV*, 2019. 3

[46] Pavel Tokmakov, Jie Li, Wolfram Burgard, and Adrien Gaidon. Learning to track with object permanence. In *ICCV*, 2021. 2

[47] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv:2012.12877*, 2020. 2

[48] ultralytics. Yolov5. https://github.com/ultralytics/yolov5, 2020. 7

[49] 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. 2, 5

[50] Qiang Wang, Yun Zheng, Pan Pan, and Yinghui Xu. Multiple object tracking with correlation learning. *CVPR*, 2021. 8

[51] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. *arXiv:2104.04691*, 2021. 8

[52] Yongxin Wang, Kris Kitani, and Xinshuo Weng. Joint object detection and multi-object tracking with graph neural networks. In *ICRA*, 2021. 8

[53] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, 2021. 2

[54] Zhongdao Wang, Liang Zheng, Yixuan Liu, and Shengjin Wang. Towards real-time multi-object tracking. In *ECCV*, 2020. 1, 2, 5

[55] Nicolai Wojke and Alex Bewley. Deep cosine metric learning for person re-identification. In *WACV*, 2018. 1, 2

[56] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *CVPR*, 2019. 3

[57] Haiping Wu, Yuntao Chen, Naiyan Wang, and Zhaoxiang Zhang. Sequence level semantics aggregation for video object detection. In *ICCV*, 2019. 3

[58] Jialian Wu, Jiale Cao, Liangchen Song, Yu Wang, Ming Yang, and Junsong Yuan. Track to detect and segment: An online multi-object tracker. In *CVPR*, 2021. 8

[59] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6

[60] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *ICCV*, 2019. 1, 2

[61] Yihong Xu, Yutong Ban, Guillaume Delorme, Chuang Gan, Daniela Rus, and Xavier Alameda-Pineda. Transcenter: Transformers with dense queries for multiple-object tracking. *arXiv:2103.15145*, 2021. 2, 7, 8

[62] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *CVPR*, 2018. 6

[63] Qian Yu, Gérard Medioni, and Isaac Cohen. Multiple target tracking using spatio-temporal markov chain monte carlo data association. In *CVPR*, 2007. 1

[64] Fangao Zeng, Bin Dong, Tiancai Wang, Cheng Chen, Xiangyu Zhang, and Yichen Wei. End-to-end multiple-object tracking with transformer. *arXiv:2105.03247*, 2021. 2, 3, 7, 8

[65] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008. 1, 2, 3

[66] 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:2004.01888*, 2020. 1, 2, 3, 5, 6, 7, 8

[67] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020. 2

[68] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. *ECCV*, 2020. 1, 2, 3, 5, 6, 8

[69] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. In *arXiv:2103.07461*, 2021. 2, 6

[70] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv:1904.07850*, 2019. 1, 3, 4, 6

[71] Tianyu Zhu, Markus Hiller, Mahsa Ehsanpour, Rongkai Ma, Tom Drummond, and Hamid Rezatofighi. Looking beyond two frames: End-to-end multi-object tracking using spatial and temporal transformers. *arXiv:2103.14829*, 2021. 3

[72] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2, 4