Rethinking Training Strategies and Improving Model Structures for Multiple-Object Tracking

Yihong Xu

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ GRENOBLE ALPES

Spécialité : **Mathématiques et Informatique**

Arrêté ministériel : 25 mai 2016

Présentée par

Yihong XU

Thèse dirigée par **Xavier ALAMEDA-PINEDA**, Université Grenoble Alpes
et codirigée par **Radu HORAUD**, Directeur de recherche, Université Grenoble Alpes

préparée au sein du **Laboratoire Jean Kuntzmann**
dans **l'École Doctorale Mathématiques, Sciences et technologies de l'information, Informatique**

Amélioration de stratégies d'entrainements et développement de modèles pour le suivi d'objets multiple

Rethinking Training Strategies and Improving Model Structures for Multiple-Object Tracking

Thèse soutenue publiquement le **8 juin 2022,**
devant le jury composé de :

Monsieur PATRICK PEREZ
Ingénieur docteur, VALEO, Rapporteur
Monsieur VINCENT LEPETIT
Chercheur HDR, ECOLE NATIONALE DES PONTS ET CHAUSSEES, Examinateur
Monsieur NICULAE SEBE
Professeur, Università degli Studi di Trento, Rapporteur
Monsieur ERIC GAUSSIER
Professeur des Universités, UNIVERSITE GRENOBLE ALPES, Examinateur, Président du jury

ABSTRACT

Multiple-object tracking (MOT) aims to provide all object trajectories (accurate positions and consistent identities) in a given scene. The predominant way of performing MOT is *tracking-by-detection*. It first detects the positions of objects in video frames and then associates them despite object occlusions, (re-)occurring and disappearing. Therefore, MOT relies on both *object detection* and *object temporal association*. The two steps are often treated as two computer-vision sub-tasks: one aims to robustly detect objects and the other focuses on optimally associating the frame-to-frame object positions. Within this paradigm, this thesis investigates three different problems in MOT: (i) Deep learning-based MOT methods have been proposed but they are still trained with separate objective functions, directly transferred from their corresponding sub-tasks: typically, a bounding-box regression loss for the detection and an object-identity classification loss for the association. Instead, standard MOT evaluations are unified metrics considering object miss detections, false detections, and identity changes. To close this train-evaluation gap, we propose a MOT training framework through a deep Hungarian proxy–*deepMOT* (CVPR2020), leveraging standard evaluation metrics as objective functions for training any deep MOT method; (ii) MOT becomes more challenging with the recent introduction of very crowded scenes (e.g.MOT20). The challenging scenario with dense interactions among objects motivates us to leverage the global dependency of transformers. To this end, we question the current MOT model structures and conceive a novel transformer-based MOT tracking method–*TransCenter* (TPAMI revision), exhibiting state-of-the-art performance; (iii) Current MOT methods are mostly trained with the supervision of abundant object-position and -identity annotations. However, this is usually not possible in real-world applications where we only have some annotated videos for training while we deploy the MOT method to unlabeled scenes. We tackle this issue and propose an unsupervised domain adaptation MOT training framework–*DAUMOT* (CVIU submission), to overcome the limitation of unlabeled data. To conclude, (i) we address the supervised MOT training problem by unifying the evaluation metrics and the training losses; (ii) Facing the challenging crowded scenes, we rethink the MOT model structures and propose a state-of-the-art transformer-based MOT method; (iii) To tackle the unavailability of annotations in real-world data, we explore an unsupervised domain adaptation framework for training MOT methods without labels in the target scenes. The code of our works is publicly available for further research.

RÉSUMÉ

Le suivi d'objets multiple (MOT) vise à fournir les trajectoires des objets présents dans une séquence vidéo donnée. Le suivi d'objets multiple est un problème complexe. Dans une trajectoire, l'objet doit conserver une identité cohérente et inclure toutes ses positions au court du temps. Une approche prédominante de MOT est appelée *suivi par détection*: elle détecte d'abord toutes les positions d'objets dans la vidéo, puis tente de les associer temporellement en dépit du risque d'occlusion, de (ré)apparition et disparition d'objets. Par conséquent, le MOT s'appuie à la fois sur sa capacité à bien *détecter les objets* et sur la cohérence de son *association temporelle*. Ces problèmes sont généralement traités comme deux sous-tâches de la vision par ordinateur : l'une visant à détecter les objets et l'autre à trouver l'association temporelle entre detections. Avec le développement rapide des méthodes basées sur l'apprentissage profond, des méthodes MOT ont été proposé avec succès. Cependant, elles sont toujours entraînées avec des fonctions de coût distinctes, directement transférées de leurs sous-tâches respectives : une fonction de coût pour la régression de la position des detections, et une pour la classification/identification d'objets afin de résoudre l'association temporelle. Au contraire, les évaluations standard de MOT utilisent des métriques unifiées qui prennent en compte à la fois les détections manquées, les fausses détections et les changements d'identité des objets. Pour combler ce fossé entre entraînement et évaluation, nous proposons une approche novatrice pour le MOT -deepMOT- utilisant ces métriques d'évaluation comme fonctions de coût, et généralisable à n'importe quelle méthode de MOT profonde. Ces dernières années, le MOT est devenu plus complexe à cause de la densité importante des objets trackés dans les scènes issues des nouveaux datasets. Ces scénarios aux interactions complexes nous a incité à tirer parti de la capacité à saisir les dépendances globales des architectures transformers. À cette fin, nous proposons une nouvelle méthode MOT -TransCenter-, et présentant des performances de tracking supérieures à celles de l'état de l'art. Toutes les méthodes ci-dessus sont construites sur des bases de données d'entraînement fournissant les annotations d'identité et de position des objets. Dans un contexte plus réaliste d'entraînement de ces méthodes à un environnement cible spécifique (laboratoire, lieux publique ect), ces annotations ne sont disponibles que pour un sous-ensemble de données. Pour résoudre ce problème, nous proposons une approche non supervisée - DAUMOT - qui surmonte cette limitation par le biais de stratégies d'adaptation de domaine non supervisé.

Acknowledgments

Firstly, I would like to thank my supervisor Xavier who chose me as his Ph.D. student and gave me plenty of opportunities during the journey. His guidance shaped my way of thinking and solving challenging scientific problems. I should also thank Aljosa and Laura for welcoming me during my stay at TUM, teaching me with their greatest patience very basic scientific qualities, and making my visit unforgettable. Secondly, I owe my gratitude to Yutong, a former Ph.D. student in the team and a friend of my Ph.D. life. He cheered me up when doing research became tedious and his patience calmed me down when problems got tough. Thirdly, I also want to thank Xiaofei, Radu, and all the other colleagues in the team RobotLearn (former Perception). Their words of wisdom keep inspiring me. Especially, I should thank Guillaume, my best collaborator. His scientific rigor made our collaborations successful and I truly admire this quality that shapes a real scientist.

I feel grateful to my family who supported me to return back to France for the challenging but meaningful Ph.D. life. Indeed, one can hardly "survive" from the Ph.D. without supportive supervisors, colleagues, friends, and family. Particularly for me, the support is also from my flatmate taking care of our daily life, especially during my submission deadlines.

Finally, I would like to show my greatest thank to all the reviewers (Patrick and Nicu), examiners (Eric, Vincent) and the invited guests (Laura, Jakob) for the review of my thesis and their participation for the defense.

Yihong Xu
June 2022

To all the brave souls pursuing a Ph.D. degree.

Contents

Chapter 1

Introduction

1.1 General Context

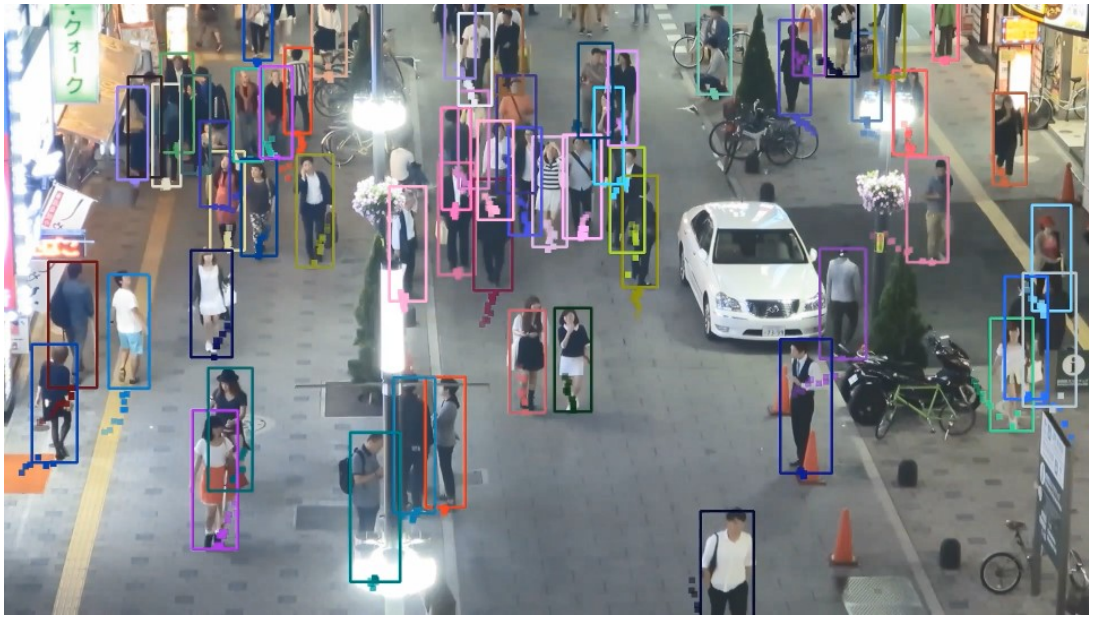


Figure 1.1: MOT aims to estimate for each object/person a trajectory (bounding boxes with trajectory tails in the figure) with a consistent identity (different colors), as shown in the above example from MOT17-04 [124].

In computer vision, researchers develop methods enabling machines to perceive and understand the world from visual data. These data are often expressed as

digitized images quantified into discrete values. Machines are taught with analytic formulations or algorithms to process the raw discrete values and to produce the desired output. Examples of some common computer vision tasks are image classification [92], segmentation [66] and object detection [59, 137, 139, 164]. Using such methods as building blocks, researchers also boost the development of higher-level computer vision tasks such as single-object tracking (SOT).

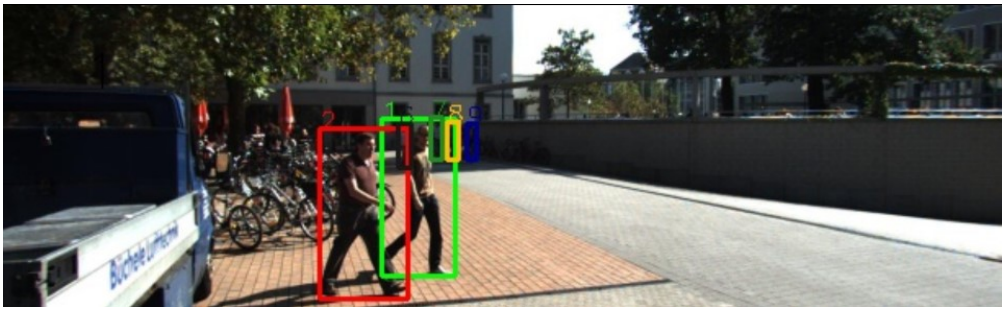
Works on SOT tackle one object at a time. Notably in 2010, correlation-filter-based methods, firstly introduced in [13] to SOT, have dominated the state-of-the-art performance leaderboards [91, 182, 183] for years. After that, we see learning-based methods enter to the domain since 2016 such as the siamese deep neural networks, firstly used in SOT by SiamFC [10], made real-time SOT feasible. Since then, SiamFC has drawn much attention to SOT with convolutional neural networks [100, 101]. Importantly, the development was boosted by the release of widely-used datasets such as OTB-13 (Object Tracking Benchmark) [182] in 2013, OTB-15 [183] in 2015, and also the VOT 15-21 datasets [91].

More complex than SOT, multiple-object tracking (MOT) aims to perceive *multiple* objects and depict their trajectories. This enables machines to follow all the objects in a given video sequence, which is useful in applications such as security surveillance, robot perception, and autonomous driving. As illustrated in Fig. 1.1, MOT tackles the problem of finding the trajectories $\mathbf{X} = \{\mathbf{X}^1, \mathbf{X}^2, ..., \mathbf{X}^i, ...\mathbf{X}^N\}$ of objects $i = 1, 2, .., N$ in a video sequence. A trajectory $\mathbf{X}^i$ consists of L_i frames of positions $\boldsymbol{x}^i_{t_l} \in \mathbb{R}^4$ (2D coordinate and object size) with $l = 1..., L_i$ and an unique identity i. From this formulation, we can see that an ideal MOT method should (i) provide complete, correct, and identity consistent trajectories for all the objects; (ii) handle the varying number of objects that (re-)appear and disappear in the given (video) sequence. In this thesis, we address the multiple-person tracking problem, which is also commonly denoted as MOT in the community and the methodologies can be generalized to objects.

Before going into details of our contributions, we present the preliminaries to help the understanding of MOT and our works: First, we introduce the commonly-used MOT datasets. Second, we summarize state-of-the-art MOT methods with a focus on the learning-based MOT methods. Third, we introduce the widely-used MOT metrics and describe the process of the predominant MOT metrics. Finally,

we summarize the existing problems and our solutions. The structure of the thesis is presented to ease the reading.

1.2 MOT Datasets



(a) MOT15-KITTI-17

(b) MOT17-13

(c) MOT20-05

Figure 1.2: Examples of the widely-used MOT datasets: MOT15 [97], MOT16/17 [124] and MOT20 [38]. Different colors represent different object identities.

Despite the MOT benchmark being introduced with the PETS dataset [44] in 2009 and developed in 2010, the video data were divided into different tasks and the ones for MOT remained simple compared to crowded real-world scenes. Moreover, for comparing to state-of-the-art methods, we usually submit our tracking results to an official remote server provided by the dataset owner. For PETS [44], the submissions were however limited to once a year during its conference workshops, making the comparisons more difficult and less efficient. In 2015, a more standardized and diverse benchmark was released–MOT15 [97]. It provides real-world scenarios

(street view, campus, etc.) for multiple-person tracking and the user-friendly challenge organization attracted greatly the research efforts in this domain. Concretely, MOT15 contains 5,500 frames in 11 videos in the train set and 5,783 frames in 11 videos for the test set. The dataset consists of 39,905 bounding boxes distributed over 500 trajectories in the train set. MOT15 is challenging since most of the video sequences were captured with severe camera motion, causing the rapid movements of pedestrians between frames. Like MOT15, the MOTChallenge team released afterward MOT16/17 [124] (MOT17 uses the same video sequences as MOT16 but with more public detection options, see below). MOT17 has videos with both crowded scenes (e.g. MOT17-04) and moving-camera scenes (e.g. MOT17-13). It contains 1,638 trajectories over 15,948 frames for the train set and 2,355 trajectories distributed in 17,757 frames for the test set. Following MOT17, the recently released MOT20 tackles very crowded scenes with different lighting conditions from only static camera views (e.g. a dim night view from the roof of a stadium). It has 2,332 trajectories in 8,931 frames for the train set and 1,501 trajectories within only 4,479 frames for its test set, making it be a much more challenging dataset. Fig. 1.2 shows some examples from these datasets.

Moreover, the MOT datasets offer public detections for a fair comparison of the data association ability among different MOT methods. Specifically, MOT17 provides three types of detections from different state-of-the-art detectors, namely DPM [49], FRCNN [139] and SDP [190]. Besides, a private detection mode is also available if the MOT method uses its own detections. The MOTChallenge website has official leaderboards comparing different submissions of state-of-the-art methods using standard MOT metrics, mainly MOTA. It provides a fair and standardized platform for the research in MOT, greatly facilitating the comparisons among MOT methods.

Even MOT15, 16/17, and 20 remain the mostly-used datasets for multiple-person tracking, there exist other datasets tackling different scenarios in MOT. Among them, the KITTI dataset offers 2D and 3D multiple-object (vehicle and person) tracking data in an autonomous driving context; A similar but larger-scale vehicle-driving video dataset–BDD100K [195] provides 100 videos with annotations for 10 computer-vision tasks including MOT. DAVIS17-19 [18, 19, 133] and Youtube-VIS [193] provide segmentation masks for multiple instances such as humans,

animals, vehicles, and objects in videos of common life. Similarly, MOTS [170] offers segmentation annotations for MOT17 videos. Both allow the exploration of MOT via segmentation; The STEP [177] dataset steps further to segment every pixel in some MOT17 sequences.

Differently, DukeMTMC [141] provides a multi-camera tracking scenario for pedestrian tracking. CroHD [158] raises interest in head tracking, and TAO dataset [36] targets the tracking for any object with overall 800 categories, being a building block for the open-world object tracking problem [116].

1.3 MOT Methods

In general, MOT methods can be grouped into online and offline methods. In online methods, we produce the predictions at t only based on information from $t-1$ and before while offline methods do not have this constraint and usually consider that the whole tracking information is available. Obviously, online methods are more adapted to real-world applications due to their causality while offline methods usually serve for the post-processing or the labeling process. In this thesis, we focus on the online modality even if the extension to offline methods is intuitive and feasible.

MOT is challenging because it should handle trajectories of multiple objects simultaneously. The predominant methodology to address this complex task is called *tracking by detection*. With this, the MOT task can be respectively divided into two sub-tasks: *object detection* and *object temporal association*. On one hand, MOT should detect *all* the objects at each frame. The detection ability has a great impact on finding adequate and accurate object trajectories. Moreover, without sufficient detections, the object association can hardly be performed. For this reason, we see the MOT performance evolve with the progress in object detection [7, 122, 131, 203, 209]. On the other hand, multiple objects interact with each other with mutual occlusions and having sufficient detection candidates means a more difficult association process. Given the detections, a MOT method also needs to correctly associate them through frames and assign them with consistent identities. This association ability often relies on the bounding-box geometry,

typically Intersection over Union (IoU) of object bounding boxes [11, 202, 209], and appearance characteristics of objects [24, 26, 79, 94, 96, 157, 197] such as color histograms, histograms of oriented gradient [35], and deep appearance features [96]. In modern methods, the bounding-box geometry and object-appearance cues are often used jointly [7, 12, 104, 196, 203].

To model the data association process, early works [3, 5, 140] use probabilistic models to optimally solve the association problem in MOT. These traditional probabilistic models often suppose that the detections are provided, such as from the public detections in MOTChallenge. The data association problem is also modeled as graphs. Specifically, [73, 86, 159, 160, 161] leverage traditional graph methods to model the positions of objects as nodes and the temporal connection of the objects as edges. This graph-based approach is further developed by using Graph Neural Networks (GNNs) where traditional graphs are replaced by learnable GNNs [14, 65, 130, 176, 180, 181] to model the complex interactions of the objects.

While sufficient data [144, 167] and the development of computing hardware helped deep-learning-based methods achieve much better performance in many computer vision tasks such as image classification [92] and object detection [59, 137, 139], it is not trivial to construct a deep MOT method. The reasons lie not only in the aforementioned difficulty in managing the detection and association of multiple objects but also in the fact that objects can appear, disappear and reappear at any frame of the given video sequence. The multiple sub-tasks and the varying number of objects make MOT difficult to be formulated with neural networks in an end-to-end (differentiable) and efficient manner. In previous works, deep learning was only leveraged to tackle either the object detection [99] or the feature extraction for object association [96, 149]. Not until 2017, [33, 125, 155] formulated the end-to-end deep MOT method as multiple SOT instances or several concatenated sub-networks using recurrent neural networks (RNNs) or convolutional neural networks (CNNs). Nevertheless, the object association, occurrence and termination (i.e. birth and death) processes remain not end-to-end.

The proof of concept of modeling MOT with neural networks [33, 125, 155] makes deep MOT methods become popular. Notably, [7] advances the MOT performance significantly by introducing a simple *tracking by detection* variant named *tracking by regression* leveraging the FRCNN detector [139]. The detector provides reliable

positions and the positions at $t-1$ are used as region proposals [139] to regress (using the FRCNN regression head) their corresponding positions at t, i.e. object association. Following such significant improvement, [209] then [203] replaces the proposal-based detector with a keypoint-based detector [210] outputting dense heatmaps, which greatly reduces the ambiguity in the bounding-box representation of object positions, especially in crowded scenes. Similar to the *tracking by regression* framework, [154] detects new objects from a region-proposal network while associating objects using a siamese network comparing the object appearances at different time steps.

In recent years, the focus on the detection ability of MOT also moves towards the interpolation of occluded objects that are usually ignored by common detectors. Motion interpolation is a performance booster in MOT since a well-designed motion model can compensate for missing detections due to occlusions. To this end, [64, 147] focus on the motion-based interpolation using traditional probabilistic models. Very recently, [202] benefits from the efficient and powerful YOLOX detector [57] and suggests making use of low-score detections to discover (partially) occluded objects.

1.4 MOT Evaluation

To assess the performance of a MOT method, we commonly use CLEAR-MOT [9] as the standard evaluation metric. It considers both the detection and the object association processes by comparing the trajectories between the ground-truth and predicted objects. Recall that MOT tackles the problem of finding the trajectories $\mathbf{X} = \{\mathbf{X}^1, \mathbf{X}^2, ..., \mathbf{X}^i, ..., \mathbf{X}^N\}$ of objects $i = 1, 2, .., N$ in a video sequence. A trajectory of L_i frames can be formulated as $\mathbf{X}^i = \{\boldsymbol{x}^i_{t_1}, \boldsymbol{x}^i_{t_2}, ..., \boldsymbol{x}^i_{t_{L_i}}\}$ where $\boldsymbol{x}^i_{t_l} \in \mathbb{R}^4$ is the 2D position and object size at time t_l of object i. Importantly, at evaluation time, CLEAR-MOT calculates the metrics frame-by-frame. At time t, the N_t predicted bounding boxes, $\mathbf{x}^{i_1}_t, \ldots, \mathbf{x}^{i_{N_t}}_t$ must be compared to the M_t ground-truth objects, $\mathbf{y}^{j_1}_t, \ldots, \mathbf{y}^{j_{M_t}}_t$. We first need to compute the correspondence between predicted bounding boxes and ground-truth objects. This is a non-trivial problem as multiple ground-truth boxes may overlap and thus can be fit to several track hypotheses. Moreover, the numbers of objects N_t and M_t vary with time t.

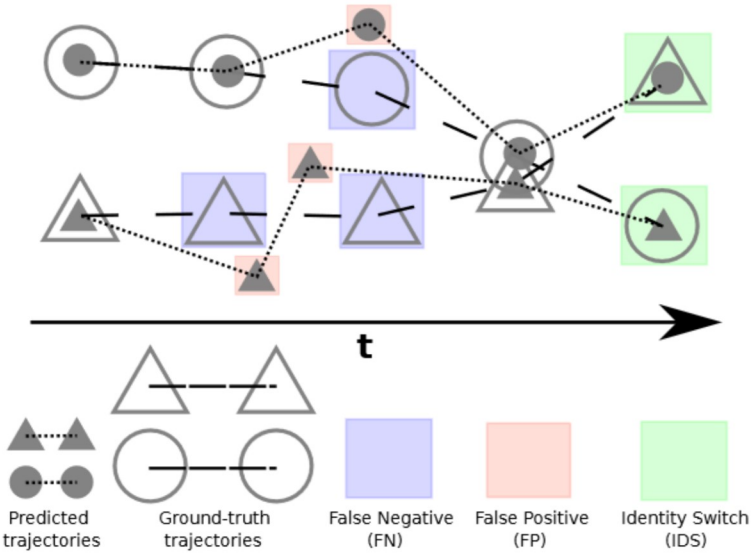


Figure 1.3: An illustration of false negatives (FN), false positives (FP) and identity switches (IDS) used in the CLEAR-MOT [9] metric. Triangles and circles represent two different trajectories with different identities.

In the following, we will omit temporal index t to ease the reading. All expressions will be evaluated with respect to time index t unless specified otherwise.

The standard metric–CLEAR-MOT, proposed in [9], tackles this association problem using bi-partite matching. First, a prediction-to-ground-truth distance matrix $\mathbf{D} \in \mathbb{R}^{N \times M}$,[1] $d_{nm} \in [0, 1]$ is computed. For vision-based tracking, an IoU (Intersection over Union)-based distance is commonly used. Then, the optimal prediction-to-ground-truth assignment binary matrix is obtained by solving the following integer program using the Hungarian (Kuhn–Munkres) algorithm (HA) [93]:

$$\mathbf{A}^* = \underset{\mathbf{A} \in \{0,1\}^{N \times M}}{\operatorname{argmin}} \sum_{n,m} d_{nm} a_{nm}, \quad \text{s.t.} \sum_{m} a_{nm} \leq 1, \forall n; \tag{1.1}$$
$$\sum_{n} a_{nm} \leq 1, \forall m; \quad \sum_{m,n} a_{nm} = \min\{N, M\}.$$

By solving this integer program we obtain a mutually consistent association between

[1]The distance matrix $\mathbf{D}$ is considered without those objects/tracks that are thresholded-out, i.e., too far from any possible assignment.

ground-truth objects and track predictions. The constraints ensure that all rows and columns of the assignment should sum to 0 or 1 – one-to-one assignment, thus avoiding multiple assignments between the two sets. After finding the optimal association $\mathbf{A}^*$, we can compute MOTA and MOTP using $\mathbf{A}^*$ and $\mathbf{D}$:[2]

$$\text{MOTA} = 1 - \frac{\sum_t (\text{FP}_t + \text{FN}_t + \text{IDS}_t)}{\sum_t M_t}, \tag{1.2}$$

$$\text{MOTP} = 1 - \frac{\sum_t \sum_{n,m} d_{tnm} a^*_{tnm}}{\sum_t |\text{TP}_t|}, \tag{1.3}$$

where a^*_{tnm} is the (n, m)-th entry of $\mathbf{A}^*$ at time t. The true positives (TP) correspond to the number of matched predicted tracks and false positives (FP) correspond to the number of non-matched predicted tracks. False negatives (FN) denote the number of ground-truth objects without a match. Finally, to compute ID switches (IDS) we need to keep track of past-frame assignments. Whenever the track assigned to a ground truth object changes, we increase the number of IDS and update the assignment structure. Fig. 1.3 illustrates some examples of FP, FP, and IDS, and the rest of predicted-ground-truth pairs are TP. As an extension of MOTA, PR-MOTA [178] and AMOTA [179] consider confidence ranked detection/tracking results instead of equally considering all tracking results with different confidence scores as in MOTA.

Besides the CLEAR-MOT metric, another common MOT metric called IDF1 [142] is proposed in 2016. It was formerly proposed to tackle the multiple-camera MOT tracking problem in datasets such as DukeMTMC [141] but can be easily extended to single-camera MOT datasets [38, 124]. The major difference from CLEAR-MOT is the one-to-one matching step. IDF1 does not perform the matching at the per-frame-detection but at the trajectory level. It matches the ground-truth trajectories and the predicted ones by maximizing the number of matched detections for each matched trajectory. By doing this, each ground-truth trajectory can only be matched to at most one predicted trajectory and inversely. After the matching, the number of matched detections are noted as identity true positives (IDTP), the rest of ground-truth and predicted detections are identity false negatives (IDFN) and identity false positives (IDFP), respectively. From the matching strategy, we see

[2] Accounting also for the objects/tracks that were left out.

that IDF1 focuses more on the association consistency than CLEAR-MOT. Even IDF1 is usually paired with MOTA to assess the MOT performance, CLEAR-MOT is still the dominant metric in most of the MOT leaderboards since the detection ability is often the principle performance bottleneck for a MOT method. Moreover, MOTA and IDF1 have a strong correlation, reflected in results shown in Chap. 2.

1.5 Contributions

Deep MOT methods have led the MOT performance since 2017 and the CLEAR-MOT [9] is widely used to evaluate different aspects of the MOT performance (detection and association). Nevertheless, training a deep MOT method using standard metrics as objective functions is not properly addressed: previous MOT methods are usually trained with separate losses dealing independently with the object detection and association. For object detection, a common loss is the L1 distance regression between the ground-truth and predicted bounding boxes, without considering their identities. For object association, an object-identity classifier, trained with triplet loss [50] or cross-entropy loss, is used to extract object identity features that are used to associate objects temporally based on the feature (euclidean or cosine) distances. Furthermore, the motivation to make standard MOT metrics trainable is not intuitive because it needs differentiable approximations: as defined in Sec. 1.4, The HA, MOTA and MOTP contain non-differentiable operations and thus cannot be directly used for tracker optimization with the gradient descent technique. To close this discrepancy between training and evaluation, deepMOT proposes a differentiable version of HA–Deep Hungarian Net (DHN) as the gradient proxy and approximates MOTA and MOTP to be loss functions, optimizing jointly a deep MOT method both in object detection and association.

A novel MOT-specific training framework like DeepMOT can indeed ease the MOT training process and improve the detection and association performance of deep MOT methods. However, the performance bottleneck lies also in the deep MOT structures. To this end, efforts have been made in probabilistic models [2, 15], CNN-based [7, 96, 203, 209], RNN-based [125, 155] and graph-based [14, 149]

methods. In spite of the efforts being made, the recent MOT20 [38], containing very crowded tracking scenarios and frequently-present occlusions, makes MOT much more challenging for most of the modern MOT methods. Since 2021, attention has been driven to transformers [169] where its global dependency is intuitively suitable for dealing with dense object interactions in MOT. In parallel to our solution, works like [122, 131] directly extend the transformer-based object detection framework–DETR [20, 214] to the MOT task. Differently, we redesign a transformer-based MOT method with pixel-level dense queries–TransCenter, setting a new state-of-the-art MOT baseline while running with a high efficiency.

The above-mentioned methods are built under the hypothesis that we possess sufficient and accurate object positions and identities for training, like in the standard MOT datasets [38, 124]. However we often only have an annotated subset of training videos (source) while the real-world video scenes are not annotated (target). This makes the deployment of a MOT method to real-world applications difficult. The difficulty is mainly because the background and the object appearances vary from one video sequence/dataset/camera angle to another, i.e. domain shifts. In this case, the traditional supervised training can no longer be used since the annotations in the target set are not available. Therefore, it is important to think of an effective way to train a deep MOT method without available annotations–unsupervised MOT. To address this, we see efforts such as *Tracking by animation* [69, 76] and SimpleReID [83]. The former is proposed to tackle the unsupervised MOT problem but the training data are synthesized floating objects that are far from being realistic. The latter, though claimed to be an unsupervised MOT method, uses trajectories produced by an existing tracker that has been trained on standard MOT datasets [38, 124] in a supervised manner.

Different from them, the existence of domain shifts among videos motivates us to the solution with domain adaption so that we break these domain shifts to improve the MOT performance (in the target set). Indeed, previous works in the field like the domain adaptive object detection [23, 28, 126, 146, 187, 213] have highlighted the drop of detection performance in the case of a significant domain shift, and the detection domain adaptation has been addressed in [28] using a binary domain discriminator. The unsupervised training with domain adaptation is also well-addressed in the unsupervised person re-ID (URID) [46, 51, 56, 113, 200].

However, to our best knowledge, unsupervised training of MOT with domain adaptation remains unsolved. Inspired by these works and considering the complex nature of MOT, we propose an unsupervised MOT training framework using domain adaptation–DAUMOT. It reduces the domain gap presented in a source-pre-trained MOT method applied to the unlabeled target set and in consequence significantly improves its performance in the unlabeled target set.

To summarize, we address the MOT training and the MOT network designing issues with the following solutions:

- DeepMOT: unifying the supervised MOT training with standard MOT metrics with novel differentiable proxy and approximations;
- TransCenter: tackling MOT with dense object interactions using a novel and efficient transformer-based MOT network structure;
- DAUMOT: alleviating the performance drop in real-world unlabeled scenarios by finetuning with the proposed unsupervised domain adaptation MOT training framework.

1.6 Thesis Structure

In this thesis, we address the online MOT both in the training and the network designing aspects: (i) Chap. 2 illustrates in detail the proposed MOT training framework in the supervised setting–deepMOT; (ii) Chap. 3 studies the MOT network designs and proposes a novel deep MOT model structure using transformers with dense queries–TransCenter, demonstrating state-of-the-art performance compared to modern MOT methods; (iii) We address the unsupervised domain adaptation MOT problem in Chap. 4 and propose our solution using domain adaptation techniques–DAUMOT; (iv) Chap. 5 recalls our contributions in MOT and points out the potential research directions for the future.

Chapter 2

DeepMOT: Rethinking MOT Supervised Training

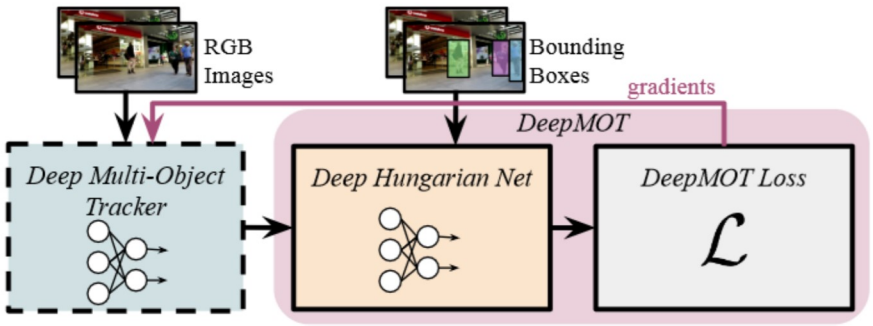


Figure 2.1: We propose DeepMOT, a general framework for training deep multiple-object trackers including the DeepMOT loss that directly correlates with established tracking evaluation measures [9]. The key component in our method is the Deep Hungarian Net (DHN) that provides a soft approximation of the optimal prediction-to-ground-truth assignment, and allows to deliver the gradient, back-propagated from the approximated tracking performance measures, needed to update the tracker weights.

2.1 Introduction

Vision-based MOT is a long-standing research problem with applications in mobile robotics and autonomous driving. It is through tracking that we become

aware of surrounding object instances and anticipate their future motion. The majority of existing methods for pedestrian tracking follow the tracking-by-detection paradigm and mainly focus on the association of detector responses over time. A significant amount of research investigated combinatorial optimization techniques for this challenging data association problem [16, 17, 63, 108, 138, 149].

Recent data-driven trends in MOT leverage the representational power of deep networks for learning identity-preserving embeddings for data association [80, 96, 171], learning the appearance model of individual targets [134, 211] and learning to regress the pose of the detected targets [7]. However, these methods train individual parts of the MOT pipeline using proxy losses (e.g. triplet loss [50] for learning identity embeddings), that are only indirectly related to the MOT evaluation measures [9]. The main difficulty in defining loss functions that resemble standard tracking evaluation measures is due to the need of computing the optimal matching between the predicted object tracks and the ground-truth objects. This problem is usually solved by using the Hungarian (Kuhn–Munkres) algorithm (HA) [93], which contains non-differentiable operations.

The significant contribution of this work is a novel, differentiable framework for the training of deep multiple-object trackers, as illustrated in Fig. 2.1: it proposes a differentiable variant of the standard CLEAR-MOT [9] evaluation measures, which we combine into a novel loss function, suitable for end-to-end training of MOT methods. In particular, we introduce a differentiable network module – Deep Hungarian Net (DHN) – that approximates the HA and provides a soft approximation of the optimal prediction-to-ground-truth assignment. The proposed approximation is based on a bi-directional recurrent neural network (Bi-RNN) that computes the (soft) assignment matrix based on the prediction-to-ground-truth distance matrix. We then express both the MOTA and MOTP [9] as differentiable functions of the computed (soft) assignment matrix and the distance matrix. Through DHN, the gradients from the approximated tracking performance measures are back-propagated to update the tracker weights. In this way, we can train object trackers in a data-driven fashion using losses that directly correlate with standard MOT evaluation measures. In summary, this work makes the following contributions:

(i) We propose novel loss functions that are directly inspired by standard MOT evaluation measures [9] for end-to-end training of deep multiple-object trackers.

(ii) To back-propagate losses through the network, we propose a new network module – Deep Hungarian Net – that learns to match predicted tracks to ground-truth objects in a differentiable manner.

(iii) We demonstrate the merit of the proposed loss functions and differentiable matching module by training the powerful MOT method–Tracktor [7] using our proposed framework. We demonstrate improvements over the baseline and establish a new state-of-the-art result (by the time of submission) on MOTChallenge benchmark datasets [97, 124].

2.2 Related Works

Tracking as Discrete Optimization. With the emergence of reliable object detectors [6, 35, 48] tracking-by-detection has become the leading tracking paradigm. These methods first perform object detection in each image and associate detections over time, which can be performed online via frame-to-frame bi-partite matching between tracks and detections [93]. As early detectors were noisy and unreliable, several methods search for the optimal association in an offline or batch fashion, often posed as a network flow optimization problem [16, 17, 63, 108, 149].

Alternatively, tracking can be posed as a maximum-a-posteriori (MAP) estimation problem by seeking an optimal set of tracks as a conditional distribution of sequential track states. Several methods perform inference using conditional random fields (CRFs) [1, 31, 128], Markov chain Monte Carlo (MCMC) [127] or a variational expectation-maximization [2, 4, 5]. These methods in general, use hand-crafted descriptors for the appearance model, such as color histograms [1, 29], optical flow-based descriptors [31] and/or motion models [6, 128] as association cues. Therefore typically only a few parameters are trainable and are commonly learned using grid/random search or tree of parzen window estimators [8, 128]. In the case of CRF-based methods, the weights can be trained using structured SVM [162, 174].

Deep Multiple-Object Tracking. Recent data-driven trends in MOT leverage representational power of deep neural networks. [185] learns track birth/death/association policy by modeling them as Markov Decision Processes (MDP). As the standard evaluation measures [9] are not differentiable, they learn the policy by reinforcement learning.

Several existing methods train parts of their tracking methods using losses, not directly related to tracking evaluation measures [9]. [88] leverages pre-learned CNN features or a bilinear LSTM [89] to learn the long-term appearance model. Both are incorporated into (Multiple Hypothesis Tracking) MHT framework [138]. Other methods [54, 80, 96, 171] learn identity-preserving embeddings for data association using deep neural networks, trained using contrastive [62], triplet [50] or quadruplet loss [80]. At inference time, these are used for computing data association affinities. Approaches by [134, 211] learn the appearance model of individual targets using an ensemble of single-object trackers that share a convolutional backbone. A spatiotemporal mechanism (learned online using a cross-entropy loss) guides the online appearance adaptation and prevents drifts. All these methods are only partially trained, and sometimes in various stages. Moreover, it is unclear how to train these methods to maximize established tracking metrics.

Most similar to our objective, [174] proposes a framework for learning parameters of linear cost association functions, suitable for network flow optimization [108] based multiple-object trackers. They train parameters using structured SVM. Similar to our method, they devise a loss function, that resembles MOTA: the intra-frame loss penalizes false positives (FP) and missed targets while the inter-frame component of the loss penalizes false associations, ID switches, and missed associations. However, their loss is not differentiable and is only suitable for training parameters within the proposed min-cost flow framework. [32] proposes an end-to-end training framework that jointly learns feature, affinity, and multi-dimensional assignment. However, their losses are not directly based on MOTA and MOTP. [149] parameterizes (arbitrary) cost functions with neural networks and trains them end-to-end by optimizing them w.r.t. the min-flow training objective. Different from [149], our approach goes beyond learning the association function and can be used by any learnable tracking method.

[7] proposes a tracking-by-regression approach to MOT. The method is trained

for the object detection task using a smooth L_1 loss for the bounding box regressor. Empirically, their method can regress bounding boxes in high-frame-rate video sequences with no significant camera motion. Apart from the track birth and death management, this approach is fully trainable, and thus it is a perfect method for demonstrating the merit of our training framework. Training this approach on sequence-level data using our proposed loss further improves the performance and establishes a new state of the art on the MOTChallenge benchmark [97].

Hungarian (Kuhn–Munkres) Algorithm (HA). HA is designed in 1957 to solve the minimum sum cost matching problem by optimally assigning (at most) one edge for connecting a pair of vertices from two independent sets in a bipartite graph. The cost (weighted) of the edges linking two vertices from two different sets can be defined as the value of an entry in a $N \times M$ matrix–distance matrix $\mathbf{D}$, where N and M are the number of vertices in two sets (e.g. N is the number of predicted bounding boxes (tracks) and M is the number of ground-truth objects in the context of MOT.) The original algorithm takes a square cost matrix (i.e. $N = M$) as the input and outputs a binary assignment matrix $\mathbf{A}^*$ where each row and column has at most one assignment (i.e. the entry value is 1). It can be extended to deal with rectangular matrices. The steps of HA are described as follows:

(i) Construct a $N \times M$ matrix by calculating the costs (distances) between vertices;

(ii) Find the minimum value for each row and subtract it from each entry in the row;

(iii) Repeat step (ii) but for each column;

(iv) Draw a minimum number of lines through rows and columns so that all the zeros are covered;

(v) If the number of drawn lines equals $min(N, M)$, the final assignment matrix can be produced by assigning one to entries having a value of zero so that each row and column has at most one assignment, and assigning zero for the others; Otherwise, go to the next step;

(vi) Find the minimum value of uncovered entries (i.e. entries that are crossed by the drawn lines) and subtract it from uncovered rows (i.e. rows that are not (fully) covered by lines) and then add it back to the covered columns, go to step (iv).

As we can see, the HA contains several non-differentiable steps, which are problematic for training with the gradient descent technique. For this reason, we

propose in the following a differentiable proxy, inspired by HA, to approximate the assignment solution in a differentiable way.

2.3 DeepMOT

As described in Sec. 1.4, the first step to compute the CLEAR-MOT [9] tracking evaluation measures is to perform bi-partite matching, using Hungarian Algorithm [93], between the sets of ground-truth objects and of predicted tracks, as described in Eq. 1.1. Once the correspondence between the two sets is established, we can count the number of TP, FP, FN, and IDS needed to express MOTA and MOTP (Eq. 1.2 and Eq. 1.3). As the main contribution, we propose a differentiable loss inspired by these measures, following the same two-step strategy. We first propose to perform a soft matching between the two sets using a differentiable function, parameterized as a deep neural network– Deep Hungarian Net (DHN). Once we establish the matching, we design a loss, approximating the CLEAR-MOT measures, as a combination of differentiable functions of the (soft) assignment matrix and the distance matrix. Alternative measures such as IDF1 [142] focus on how long the tracker correctly identifies targets instead of how often mismatches occur. However, MOTA and IDF1 have a strong correlation. This is reflected in our results – by optimizing our loss, we also improve the IDF1 measure (Sec. 2.4.4). In the following, we discuss both the differentiable matching module (Sec. 2.3.1) and the differentiable version of the CLEAR-MOT measures [9] (Sec. 2.3.2).

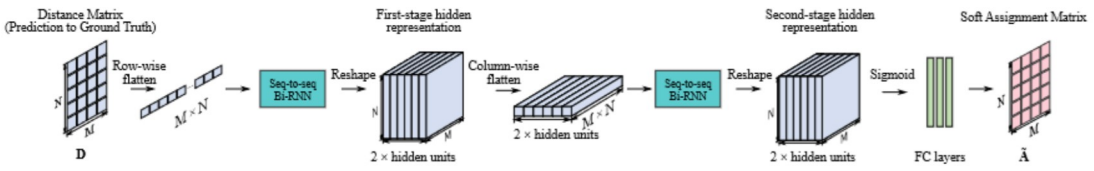


Figure 2.2: Structure of the proposed DHN. The row-wise and column-wise flattening are inspired by the original HA, while the Bi-RNN allows for all decisions to be taken globally, thus is accounting for all input entries.

2.3.1 Deep Hungarian Net: DHN

Recall the notations defined in Sec. 1.4, in this section we introduce DHN, a fundamental block in our DeepMOT framework. DHN produces a proxy $\tilde{\mathbf{A}}$ that is differentiable w.r.t. $\mathbf{D}$. Thus DHN provides a bridge to deliver gradient from the loss (to be described later on) to the tracker. We formalize DHN with a non-linear mapping that inputs $\mathbf{D}$ and outputs the proxy soft assignment matrix $\tilde{\mathbf{A}}$. DHN is modeled by a neural network $\tilde{\mathbf{A}} = g(\mathbf{D}, \omega_d)$ with parameters ω_d. Importantly, the DHN mapping must satisfy several properties: (i) the output $\tilde{\mathbf{A}}$ must be a good approximation to the optimal assignment matrix $\mathbf{A}^*$, (ii) this approximation must be differentiable w.r.t. $\mathbf{D}$, (iii) both input and output matrices are of equal, but varying size and (iv) g must take global decisions as the HA does.

While (i) will be achieved by setting an appropriate loss function when training the DHN (see Sec. 2.4.1), (ii) is ensured by designing DHN as a composite of differentiable functions. The requirements (iii) and (iv) push us to design a network that can process variable (but equal) input and output sizes, where every output neuron has a receptive field equal to the entire input. We opt for bi-directional recurrent neural networks (Bi-RNNs). Alternatively, one could consider the use of fully convolutional networks, as these would be able to process variable input/output sizes. However, large assignment problems would lead to partial receptive fields, and therefore, to local assignment decisions.

We outline our proposed DHN architecture in Fig. 2.2. In order to process a 2D distance matrix $\mathbf{D}$ using RNNs, we perform row-wise (column-wise) flattening of $\mathbf{D}$. This is inspired by the original HA that performs sequentially row-wise and column-wise reductions and zero-entry verifications and fed it to Bi-RNNs (see details below), opening the possibility for $g(\cdot)$ to make global assignment decisions.

Precisely, we perform flattening sequentially, i.e., first row-wise followed by column-wise. The row-wise flattened $\mathbf{D}$ is input to a first Bi-RNN that outputs the first-stage hidden representation of size $N \times M \times 2h$, where h is the size of the Bi-RNN hidden layers. Intuitively the first-stage hidden representation encodes the row-wise intermediate assignments. We then flatten the first-stage hidden representation column-wise, to input to a second Bi-RNN that produces the second-stage hidden representation of size $N \times M \times 2h$. The two Bi-RNNs have

the same hidden size, but they do not share weights. Intuitively, the second-stage hidden representation encodes the final assignments. To translate these encodings into the final assignments, we feed the second-stage hidden representation through three fully-connected layers (along the $2h$ dimension, i.e., independently for each element of the original $\mathbf{D}$). Finally, a sigmoid activation produces the optimal $N \times M$ soft-assignment matrix $\tilde{\mathbf{A}}$. Note that in contrast to the binary output of the Hungarian algorithm, DHN outputs a (soft) assignment matrix $\tilde{\mathbf{A}} \in [0,1]^{N \times M}$.

Distance Matrix Computation. The most common metric for measuring the similarity between two bounding boxes is the Intersection-over-Union (IoU). Note that, in principle, the input $\mathbf{D}$ can be any (differentiable) distance function. However, if two bounding boxes have no intersection, the distance $1 - \text{IoU}$ will always be a constant value of 1. In that case, the gradient from the loss will be 0, and no information will be back-propagated. For this reason, our distance is an average of the Euclidean center-point distance and the Jaccard distance $\mathcal{J}$ (defined as $1 - \text{IoU}$):

$$d_{nm} = \frac{f(\mathbf{x}^n, \mathbf{y}^m) + \mathcal{J}(\mathbf{x}^n, \mathbf{y}^m)}{2}. \tag{2.1}$$

f is the Euclidean distance normalized w.r.t the image size:

$$f(\mathbf{x}^n, \mathbf{y}^m) = \frac{\|c(\mathbf{x}^n) - c(\mathbf{y}^m)\|_2}{\sqrt{H^2 + W^2}}, \tag{2.2}$$

where function $c(\cdot)$ computes the center point of the bounding box and H and W are the height and the width of the video frame, respectively. Both the normalized Euclidean distance and Jaccard distance have values in the range of $[0, 1]$, so do all entries d_{nm}. Our framework admits any distance that is expressed as a composition of differentiable distance functions. In the experimental section, we demonstrate the benefits of adding a term that measures the cosine distance between two learned appearance embeddings. In the following, we explain how we compute a differentiable proxy of MOTA and MOTP as functions of $\mathbf{D}$ and $\tilde{\mathbf{A}}$.

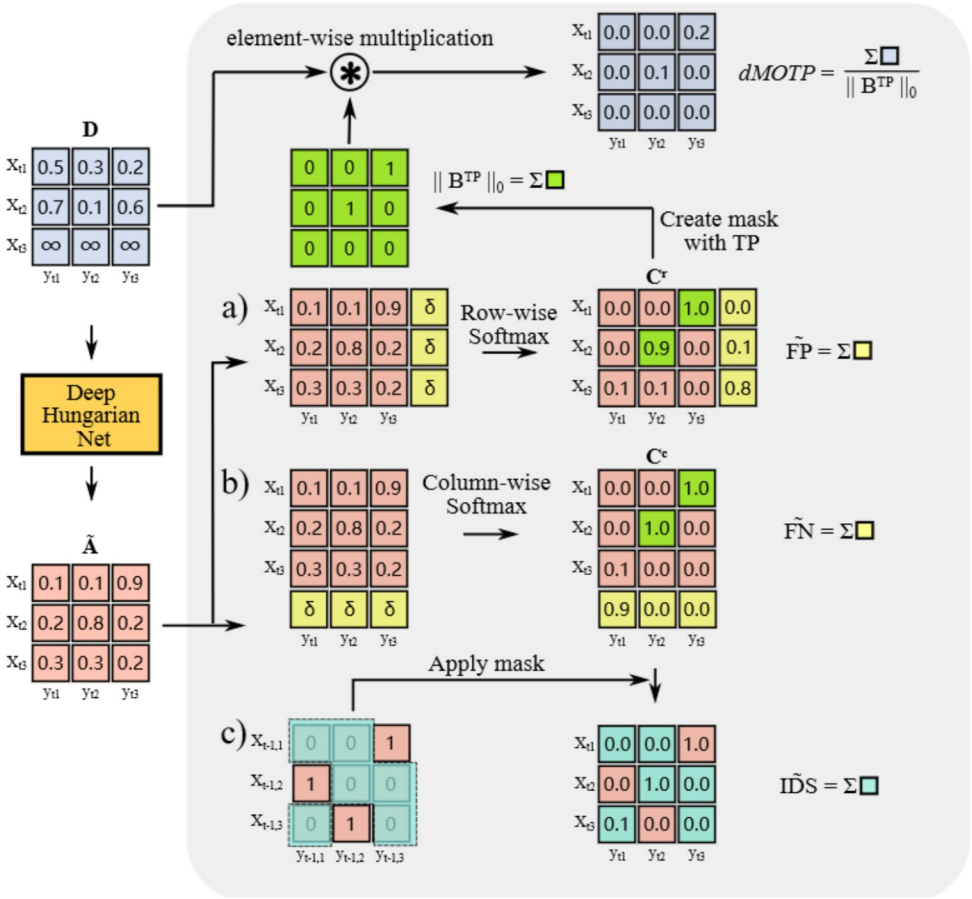


Figure 2.3: DeepMOT loss: $dMOTP$ (top) is computed as the average distance of matched tracks and $dMOTA$ (bottom) is composed with $\tilde{\text{FP}}$, $\tilde{\text{IDS}}$ and $\tilde{\text{FN}}$.

2.3.2 Differentiable MOTA and MOTP

In this section, we detail the computation of two components of the proposed DeepMOT loss: differentiable MOTA ($dMOTA$) and MOTP ($dMOTP$). As discussed in Sec. 1.4, to compute the classic MOTA and MOTP evaluation measures, we first find the optimal matching between predicted tracks and ground-truth objects. Based on $\mathbf{A}^*$, we count FN, FP and IDS. The latter is computed by comparing assignments between the current frame and previous frames. To compute the proposed $dMOTA$ and $dMOTP$, we need to express all these as differentiable functions of $\mathbf{D}$ and $\tilde{\mathbf{A}}$ computed using DHN (see Sec. 2.3.1).

The operations described in the following are illustrated in Fig. 2.3. First, we

need to count FN and FP. Therefore, we need to obtain a count of non-matched tracks and non-matched ground-truth objects. To this end, we first construct a matrix $\mathbf{C}^r$ by appending a column to $\tilde{\mathbf{A}}$, filled with a threshold value δ (e.g., $\delta = 0.5$), and perform row-wise softmax (Fig. 2.3a). Analogously, we construct $\mathbf{C}^c$ by appending a row to $\tilde{\mathbf{A}}$ and perform column-wise softmax (Fig. 2.3b). Then, we can express a soft approximation of the number of FP and FN as:

$$\tilde{\text{FP}} = \sum_n \mathbf{C}^r_{n,M+1}, \quad \tilde{\text{FN}} = \sum_m \mathbf{C}^c_{N+1,m}\,. \tag{2.3}$$

Intuitively, if all elements in $\tilde{\mathbf{A}}$ are smaller than the threshold δ, then entries of $\mathbf{C}^r_{n,M+1}$ and $\mathbf{C}^c_{N+1,m}$ will be close to 1, signaling we have a FP or FN. Otherwise, the element with the largest value in each row/column of $\mathbf{C}^r$ and $\mathbf{C}^c$ (respectively) will be close to 1, signaling that we have a match. Therefore, the sum of the $N+1$-th row of $\mathbf{C}^c$ (Fig. 2.3b) and of the $M+1$-th column of $\mathbf{C}^r$ (Fig. 2.3a) provide a soft estimate of the number of FN and the number of FP, respectively. We will refer to these as $\tilde{\text{FN}}$ and $\tilde{\text{FP}}$.

To compute the soft approximations $\tilde{\text{IDS}}$ and $dMOTP$, we additionally need to construct two binary matrices $\mathbf{B}^{\text{TP}}$ and $\mathbf{B}^{\text{TP}}_{\text{-}1}$, whose non-zero entries signal true positives at the current and previous frames respectively. Row indices of these matrices correspond to indices assigned to our tracks and column indices correspond to ground truth object identities. We need to pad $\mathbf{B}^{\text{TP}}_{\text{-}1}$ for element-wise multiplication because the number of tracks and objects varies from frame to frame. We do this by filling-in rows and columns of $\mathbf{B}^{\text{TP}}_{\text{-}1}$ to adapt the matrix size for the newly-appeared objects at the current frame by copying their corresponding rows and columns from $\mathbf{B}^{\text{TP}}$. Note that we do not need to modify $\mathbf{B}^{\text{TP}}$ to compensate for newly appearing objects as these do not cause IDS. By such construction, the sum of $\mathbf{C}^c_{1:N,1:M} \odot \overline{\mathbf{B}}^{\text{TP}}_{\text{-}1}$ (where $\overline{\mathbf{B}}$ is the binary complement of $\mathbf{B}$) yields the (approximated) number of IDS (Fig. 2.3c):

$$\tilde{\text{IDS}} = \| \mathbf{C}^c_{1:N,1:M} \odot \overline{\mathbf{B}}^{\text{TP}}_{\text{-}1} \|_1, \tag{2.4}$$

where $\|\cdot\|_1$ is the L_1 norm of a flattened matrix. With these ingredients, we can

evaluate $dMOTA$:

$$dMOTA = 1 - \frac{\tilde{\mathrm{FP}} + \tilde{\mathrm{FN}} + \gamma\, \tilde{\mathrm{IDS}}}{M}. \tag{2.5}$$

γ controls the penalty we assign to $\tilde{\mathrm{IDS}}$. Similarly, we can express $dMOTP$ as:

$$dMOTP = 1 - \frac{\| \mathbf{D} \odot \mathbf{B}^{\mathrm{TP}} \|_1}{\| \mathbf{B}^{\mathrm{TP}} \|_0}. \tag{2.6}$$

Intuitively, the L_1 norm expresses the distance between the matched tracks and ground-truth objects, and the zero-norm $\| \cdot \|_0$ counts the number of matches. Since we should train the tracker to maximize MOTA and MOTP, we propose the following DeepMOT loss:

$$\mathcal{L}_{\mathrm{DeepMOT}} = (1 - dMOTA) + \lambda(1 - dMOTP), \tag{2.7}$$

where λ is a loss balancing factor. By minimizing our proposed loss function $\mathcal{L}_{\mathrm{DeepMOT}}$, we are penalizing FP, FN and IDS–all used by the CLEAR-MOT measures [9]. Same as for the standard CLEAR-MOT measures, $dMOTA$, $dMOTP$ must be computed at every time frame t.

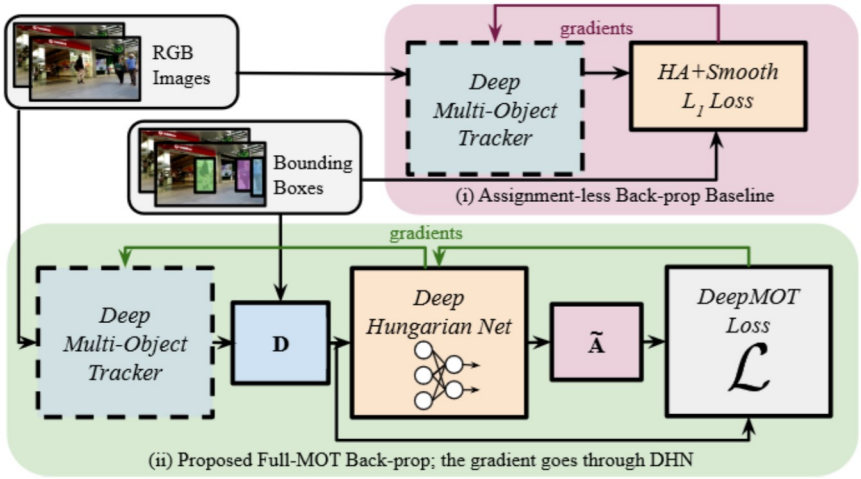


Figure 2.4: The proposed MOT training strategy (bottom) accounts for the track-to-object assignment problem, solved by the proposed DHN, and approximates the standard MOT losses, as opposed to the classical training strategies (top) using the *non-differentiable* HA.

2.3.3 How To Train Your Deep Multiple-Object Tracker

The overall tracker training procedure is shown in Fig. 2.4. We randomly sample a pair of consecutive frames from the training video sequences. These two images together with their ground-truth bounding boxes constitute one training instance. For each such instance, we first initialize the tracks with ground-truth bounding boxes (at time t) and run the forward pass to obtain the track's bounding-box predictions in the following video frame (time $t+1$). To mimic the effect of imperfect detections, we add random perturbations to the ground-truth bounding boxes.

We then compute $\mathbf{D}$ and use our proposed DHN to compute $\tilde{\mathbf{A}}$ (Sec. 2.3.1). Finally, we compute our proxy loss based on $\mathbf{D}$ and $\tilde{\mathbf{A}}$ (Sec. 2.3.2). This provides us with gradient that accounts for the (soft) assignment, and that is used to update the weights of the tracker.

2.4 Experimental Evaluation

In this section, we first detail the DHN implementation and experimentally verify that our proposed DHN is a good approximation to HA [93] for bipartite matching, as required by MOT evaluation measures (Sec. 2.4.1 and Sec. 2.4.2). We then provide the implementation details of DeepMOT and DeepMOT-trackers (Sec. 2.4.3). To show the merit of the proposed framework–DeepMOT, we conduct several experiments on different datasets for evaluating pedestrian tracking performance (Sec.2.4.4). Finally, we visualize the gradient back-propagated from the proposed losses (Sec.2.4.5).

2.4.1 DHN Implementation Details

We provide as follows the implementation details of our differentiable matching module–DHN.

DHN Training. To train the DHN, we create a data set with pairs of matrices ($\mathbf{D}$ and $\mathbf{A}^*$), separated into 114,483 matrices for training and 17,880 for matrices testing. The training pairs are generated as follows. We first compute distance matrices $\mathbf{D}$

using ground-truth labels (bounding boxes) and object detections provided by the MOTChallenge datasets (MOT 15-17) [97, 124]. We augment the data by setting all entries, higher than the randomly (with a uniform distribution ranging from 0 to 1) selected threshold, to a large value to discourage these assignments. This way, we obtain a rich set of various distance matrices. We then compute assignments using the Hungarian algorithm–HA (variant used in [9]) to get the corresponding (binary) assignment matrices $\mathbf{A}^*$, used as a supervisory signal.

We pose the DHN training as a 2D binary classification task using the focal loss [111]. We compensate for the class imbalance (between the number of zeros n_0 and ones n_1 in $\mathbf{A}^*$) by weighting the dominant zero-class using $w_0 = n_1/(n_0 + n_1)$. We weight the one-class by $w_1 = 1 - w_0$. We evaluate the performance of DHN by computing the weighted accuracy (WA):

$$\text{WA} = \frac{w_1 n_1^* + w_0 n_0^*}{w_1 n_1 + w_0 n_0}, \tag{2.8}$$

where n_1^* and n_0^* are the number of true and false positives, respectively. Since the values of the output of DHN are between 0 and 1, we threshold the output at 0.5.

For training the DHN, we use the RMSprop optimizer [165] with a learning rate of 0.0003, gradually decreasing by 5% every 20,000 iterations. We train DHN for 20 epochs (6 hours on a Titan XP GPU). For the focal loss, we weight zero-class by $w_0 = n_1/(n_0 + n_1)$ and one-class by $w_1 = 1 - w_0$. Here n_0 is the number of zeros and n_1 the number of ones in $\mathbf{A}^*$. We also use a modulating factor of 2 in the focal loss. Once the DHN training converges, we freeze the DHN weights and keep them fixed when training trackers with DeepMOT.

DHN Usage. Once the DHN is trained with the strategy described above, its weights are fixed: they are not updated in any way during the training of the deep trackers.

2.4.2 DHN Ablation Study

In this section, we perform DHN ablation using our test split of 17,880 DHN training instances, as explained previously in Sec. 2.4.1. We also evaluate the

generalization of DHN by evaluating performing evaluation using distance matrices, generated during the DeepMOT training process. Using these data, we provide (i) an ablation study on the choice of recurrent units, (ii) a discussion of alternative architectures; (iii) we experimentally assess how well the DHN preserves the properties of assignment matrices; (iv) we provide an analysis of the impact of the distance matrix size on the matching precision.

Discretization. To perform the evaluation, we first need to discretize the soft assignment matrix $\tilde{\mathbf{A}}$, predicted by our DHN to obtain a discrete assignment matrix $\bar{\mathbf{A}}$. There are two possibilities.

(i) For each row of $\bar{\mathbf{A}}$, we set the entry of $\bar{\mathbf{A}}$ corresponding to the largest value of the row to 1 (as long as it exceeds 0.5) and the remaining values are set to 0. We refer to this variant as *row-wise maximum*.

(ii) Analogously, we can perform *column-wise maximum* by processing columns instead of rows.

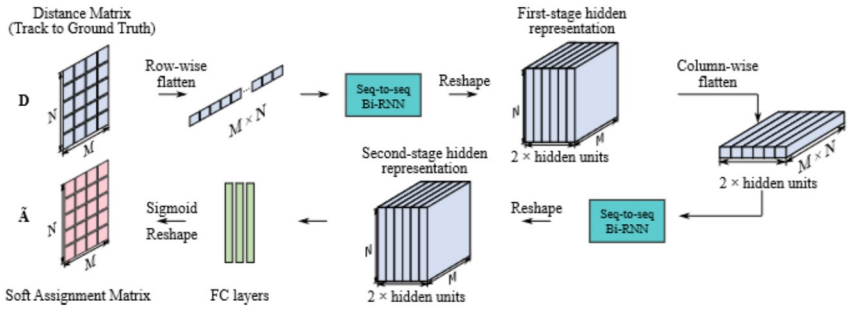


Figure 2.5: Sequential DHN: Structure of the proposed Deep Hungarian Net. The row-wise and column-wise flattening are inspired by the original Hungarian algorithm, while the Bi-RNN allows for all decisions to be taken globally, thus is accounting for all input entries.

DHN Model Variants. We compare to three different DHN architectures:

(i) Sequential DHN (**seq**, see Fig. 2.5),

(ii) Parallel DHN (**paral**, see Fig. 2.6),

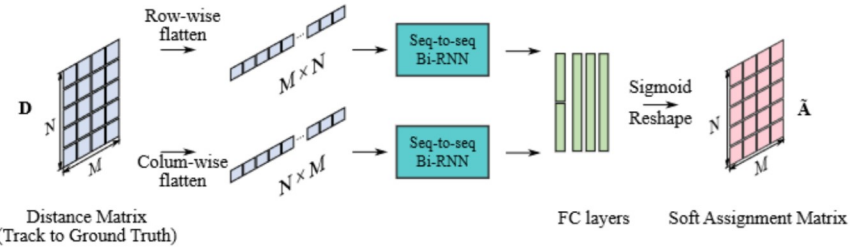


Figure 2.6: Parallel DHN variant: (i) We perform row-wise and the column-wise flattening of $\mathbf{D}$. (ii) We process the flattened vectors using two different Bi-RNNs. (iii) They then are respectively passed to an FC layer for reducing the number of channels and are concatenated along the channel dimension. (iv) After two FC layers we reshape the vector and apply the sigmoid activation.

(iii) 1D Convolutional DHN (**1d_conv**, see Fig. 2.7).

The recurrent unit of the two recurrent architectures, **seq** and **paral**, is also ablated between long-short term memory units (**lstm**) [72] and gated recurrent units (**gru**) [30].

Accuracy and Validity. We reuse Eq. 2.8 for evaluating the weighted accuracy of the outputs of our DHN. For validity, the output of the matching algorithm should be a permutation matrix; i.e., there should be at most one assignment per row/column. In the case of the HA, this is explicitly enforced via constraints on the solution. To study how well the predicted (discretized) assignment matrices preserve this property, we count the number of rows and columns by the following criteria:

- **Several Assignments (SA)** counts the number of rows/columns that have more than one assignment, after performing column-wise maximum or row-wise maximum.
- **Missing Assignments (MA)** counts the number of rows/columns that are not assigned (after performing column-wise maximum or row-wise maximum) when ground-truth assignment matrix $\mathbf{A}^*$ has an assignment or inversely when there is no assignment in $\mathbf{A}^*$ while $\bar{\mathbf{A}}$ has an assignment in the corresponding rows/columns.

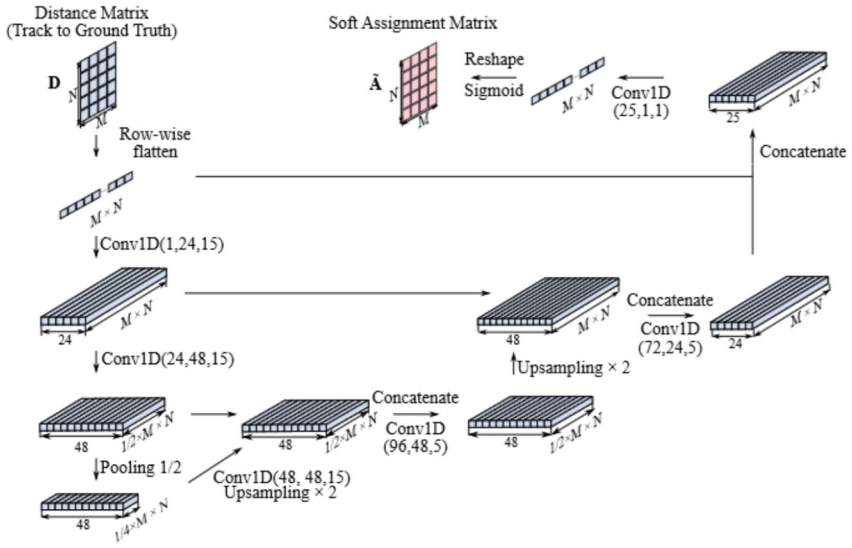


Figure 2.7: 1D convolutional DHN: Our 1D convolutional DHN variant is inspired by the U-Net [143]. The encoder consists of two 1D-convolution layers of shapes $[1, 24, 15]$ and $[24, 48, 15]$ ([#input channels, #output channels, kernel size]). The decoder consists of two 1D convolutional layers of shapes $[96, 48, 5]$ and $[72, 24, 5]$. Finally, we apply an 1D convolution and a sigmoid activation to produce $\tilde{\mathbf{A}}$.

From Tab. 2.1, we see that the proposed sequential DHN (**seq_gru**) obtains the highest WA (92.88% for row-wise maximum and 93.49% for column-wise maximum) compared to others. Compared to the 1D convolutional DHN variant (WA of 56.43% and 56.18% for row-wise and column-wise maximum, respectively), Bi-RNN shows the advantage of its global view due to the receptive field, equal to the entire input. For the sequential DHN setting, we observe in Tab. 2.1 that **gru** units consistently outperform **lstm** units with WA +9.22% (row-wise maximum) and +6.42% (column-wise maximum). Finally, the proposed sequential DHN is more accurate compared to the parallel variant of DHN (+3.32% for row-wise and +2.48% for column-wise maximum). As for the validity, the proposed **seq_gru** commits the least missing assignments (MA) (4.79% and 6.41% for row-wise and column-wise maximum, respectively), and commits only a few SA compared to other variants.

DHN is a key component of our proposed DeepMOT training framework. To evaluate how well DHN performs during training as a proxy to deliver gradients

Table 2.1: Comparison of different network structures and settings in terms of WA, MA and SA on the DHN test set.

Discretization	Network	WA% (↑)	MA% (↓)	SA% (↓)
Row-wise maximum	**seq_gru (proposed)**	**92.88**	**4.79**	3.39
	seq_lstm	83.66	13.79	5.98
	paral_gru	89.56	8.21	4.99
	paral_lstm	88.93	8.67	5.38
	1d_conv	56.43	35.06	**2.78**
Column-wise maximum	**seq_gru (proposed)**	**93.49**	**6.41**	26.57
	seq_lstm	87.07	13.54	47.04
	paral_gru	91.01	7.98	46.25
	paral_lstm	90.50	8.60	47.43
	1d_conv	56.18	79.54	**7.73**

from the DeepMOT loss to the tracker, we conduct the following experiment. We evaluate DHN using distance matrices $\mathbf{D}$, collected during the DeepMOT training process. As can be seen in Tab. 2.2, the proposed sequential DHN (**seq_gru**) outperforms the others variants, with a WA of 92.71% for row-wise and 92.36% for column-wise maximum. For validity, it also attains the lowest MA: 13.17% (row) and 12.21% (column). The SA is kept at a low level with 9.70% and 3.69% for row-wise and column-wise maximum discretizations, respectively. Based on these results, we conclude that (i) our proposed DHN generalizes well to matrices, used to train our trackers, and (ii) it produces outputs that closely resemble valid permutation matrices.

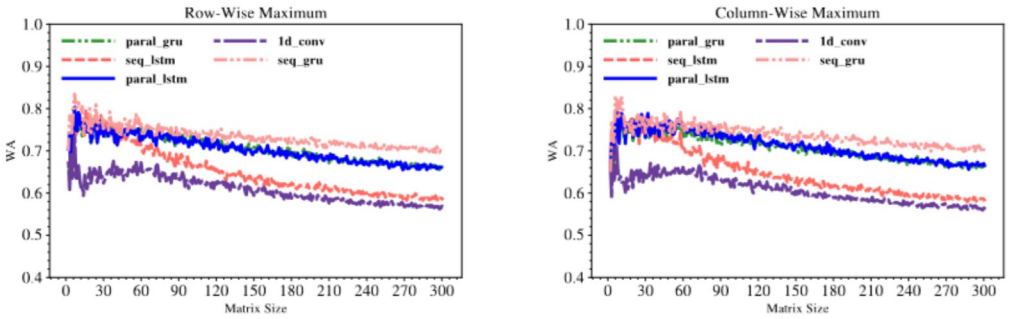


Figure 2.8: Evaluation of performance of DHN and its variants on $\mathbf{D}$ of different matrix sizes.

Table 2.2: Comparison of different network structures and settings in terms of WA, MA and SA on distance matrices during training.

Discretization	Network	WA% (↑)	MA% (↓)	SA% (↓)
Row-wise maximum	**seq_gru (proposed)**	**92.71**	**13.17**	9.70
	seq_lstm	91.64	14.55	10.37
	paral_gru	86.84	23.50	17.15
	paral_lstm	71.58	42.48	22.62
	1d_conv	83.12	32.73	**5.73**
Column-wise maximum	**seq_gru (proposed)**	**92.36**	**12.21**	3.69
	seq_lstm	91.93	13.15	4.71
	paral_gru	87.24	20.56	16.67
	paral_lstm	72.58	39.55	23.16
	1d_conv	82.74	32.94	**1.11**

Matrix Size. To provide further insights into DHN, we study the impact of the distance matrix size on the assignment accuracy. We visualize the relation between WA and the input matrix size in Fig. 2.8. For validation, we generate square matrices with sizes ranging from $[2, 300]$. Precisely, we generate $\mathbf{D}$ with a uniform distribution $[0, 1)$ and use the Hungarian algorithm implementation from [9] to generate assignment matrices $\mathbf{A}^*$. For each size, we evaluate 10 matrices, which gives us 2,990 matrices in total. As can be seen in Fig. 2.8, (i) the proposed **seq_gru** consistently outperforms the alternatives. (ii) The assignment accuracy of DHN and its variants decreases with the growth of the matrix size. Moreover, we observe a performance drop for very small matrices (i.e., $\mathrm{M} = \mathrm{N} \leqslant 6$). This may be due to the imbalance with respect to the matrix size during the training.

2.4.3 DeepMOT Experimental Settings

We demonstrate the practical interest of the proposed framework by assessing the performance of existing (deep) multiple-object trackers when trained using the proposed framework on several datasets for pedestrian tracking. We first ablate the loss terms and the tracking architectures. We also evaluate the impact of the framework w.r.t. other training alternatives. Finally, we establish a new state-of-the-art score on the MOTChallenge benchmark.

Datasets. We use the MOT15, MOT16, and MOT17 datasets for the state-of-

the-art comparisons, which provide pedestrian video sequences captured in the real-world indoor and outdoor scenarios. For the ablation studies, we divide the MOT17 into train/validation sets. We split each sequence into three parts: the first, one containing 50% of frames, the second one 25%, and the third 25%. We use the first 50% for training data and the last 25% for validation to make sure there is no overlap between the two. For MOT17, the public object detections (obtained by DPM [48], SDP [192] and Faster RCNN [139] detectors) from the MOTChallenge are used only during tracking.

Evaluation Metrics. In addition to the standard MOTP and MOTA measures [9], we report the performance using the IDF1 [142] measure, defined as the ratio of correctly identified detections over the average number of ground-truth objects and object tracks. We also report mostly tracked (MT) and mostly lost (ML) targets, defined as the ratio of ground-truth trajectories that are covered by a track hypothesis more than 80% and less than 20% of their life span respectively.

Tracktor. Tracktor [7] is an adaptation of the Faster RCNN [139] object detector to the MOT task. It uses a region proposal network (RPN) and the classification/regression heads of the detector to (i) detect objects and (ii) to follow the detected targets in the consecutive frames using a bounding box regression head. As most parts of Tracktor are trainable, this makes this method a perfect candidate to demonstrate the benefits of our framework. Note that Tracktor was originally trained only on the MOTChallenge detection dataset and was only applied to video sequences during inference. In the following, we will refer to Tracktor trained in this setting as **Vanilla Base** Tracktor. Thanks to DeepMOT, we can train Tracktor directly on video sequences, optimizing for standard MOT measures. We will refer to this variant as **DeepMOT, Base** Tracktor.

Tracktor+ReID. Vanilla Tracktor has no notion of track identity. Therefore [7] proposed to use an externally trained ReID module during inference to mitigate IDS. This external ReID module is a feature extractor with a ResNet-50 backbone, trained using a triplet loss [50] on the MOTChallenge video sequences. We will refer to this variant as **+ReIDext**. Note that this does not give Tracktor any notion of identity during training. This means that the DeepMOT loss which penalizes the number of IDS will have no significant effect on the final performance. For this

reason, we propose to replace **ReIDext** with a lightweight ReID head that we can train jointly with Tracktor using DeepMOT. This in turn allows us to utilize $\tilde{\text{IDS}}$ and to fully optimize performance to all components of CLEAR-MOT measures. We refer to this variant as **+ReIDhead**. It takes the form of a fully-connected layer with 128 units plugged into Tracktor.

Even if such a network head has been previously used in [171], it was trained externally using the triplet loss [50]. To the best of our knowledge, we are the first to optimize such an appearance model by directly optimizing the whole network for tracking evaluation measures.

For implementing the training with **ReIDhead**, we make the following changes. Instead of selecting a pair of video frames, we randomly select ten consecutive frames. This is motivated by the implementation of the external ReID mechanism in [7], where the tracker averages appearance features over ten most recent frames. At each training step, we compute representative embedding by averaging embeddings of the past video frames and use it to compute the cosine distance to the ground-truth object embeddings.

MOT-by-SOT. To demonstrate the generality of our method, we propose two additional simple trainable baselines to perform MOT by leveraging two existing off-the-shelf (trainable) single-object trackers (SOTs): GOTURN [70] and SiamRPN [102]. During inference we initialize and terminate tracks based on object detections. For each object, the SOTs take a reference image at time $t-1$ of the person and a search region in image t as input. Based on this reference box and search region, the SOTs then regress a bounding box for each object independently.

Training Details. Recall that to train object trackers, we randomly select one training instance from the sequence that corresponds to a pair of consecutive frames. Then, we initialize object trackers using ground-truth detections and predict track continuations in the next frame. At each time step, we use track predictions and ground-truth bounding boxes to compute $\mathbf{D}$, which we pass to our DHN and, finally, compute the losses and back-propagate the gradient to the tracker through DHN.

Classic bounding-box data augmentation techniques are used. We initialize trackers using ground-truth bounding boxes. To mimic the effects of imperfect object

detectors and prevent over-fitting, we perform the following data augmentations during the training:

- We randomly re-scale the bounding boxes with a scaling factor ranging from 0.8 to 1.2.
- We add random vertical and horizontal offset vectors (bounding box width and/or height scaled by a random factor ranging from 0 to 0.25).

As for training hyperparameters, when training trackers using our DeepMOT loss, we set the base value of $\delta = 0.5$, and the loss balancing factors of $\lambda = 5, \gamma = 2$, as determined on the validation set. We use the Adam optimizer [90] with a learning rate of 0.0001. We train the SOTs for 15 epochs (72h), and we train Tracktor (regression head and ReID head) for 18 epochs (12h) on a Titan XP GPU.

Track Management. In all cases, we use a simple (non-trainable) track management procedure. We (i) use detector responses to initialize object tracks in regions, not covered by existing tracks (can be either public detections or Faster RCNN detection responses in the case of Tracktor); (ii) we regress tracks from frame $t-1$ to frame t using either a SOT or Tracktor and (iii) we terminate tracks that have no overlap with detections (SOT baseline) or invoke the classification head of Tracktor, that signals whether a track is covering an object or not. As an alternative to direct termination, we can set a track as invisible for $K = 60$ frames.

2.4.4 DeepMOT Results and Discussion

Beyond Bounding Box Regression. In Tab. 2.3, we first establish the Vanilla Base Tracktor performance on our validation set and compare it to the DeepMOT Base Tracktor. This experiment (i) validates that our proposed training pipeline based on DHN delivers the gradient to the trackers and improves the overall performance, and (ii) confirms our intuition that training object trackers using a loss that directly correlates with the tracking evaluation measures has a positive impact. Note that the impact on IDS is minimal, which may be on the first sight surprising, as our proposed loss penalizes IDS in addition to FP, FN, and bounding box misalignment.

Table 2.3: Impact of the different ReID strategies for the two training strategies on Tracktor's performance.

	Method	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
Van.	Base	59.97	89.50	70.84	35.13	27.66	276	31827	326
	+ReIDext	**60.20**	**89.50**	**71.15**	**35.13**	**27.80**	**276**	**31827**	**152**
DeepMOT	Base	60.43	91.82	71.44	35.41	27.25	218	31545	309
	+ReIDext	60.62	91.82	71.66	35.41	27.39	218	31545	149
	+ReIDhead	**60.66**	**91.82**	**72.32**	**35.41**	**27.25**	**218**	**31545**	**118**

We study this by first evaluating the impact of applying external ReID module, i.e., **+ReIDext**. As can be seen in Tab. 2.3, **ReIDext** has a positive impact on the performance, as expected, in terms of MOTA (+0.23% and +0.19%) and IDS (−174 and −160) compared to **Base** for **Vanilla** and **DeepMOT** training respectively.

To further demonstrate the interest of a ReID module, we also report the **+ReIDhead** architecture trained with DeepMOT. Importantly, **+ReIDhead** cannot be trained in the Vanilla setting due to the lack of mechanisms to penalize IDS. Remarkably, **+ReIDhead** trained end-to-end with Tracktor does not only improve over the Base performance (MOTA +0.23%, IDS ↓ 191), but it also outperforms **+ReIDext** (MOTA ↑ 0.04 and IDS ↓ 31). Very importantly, the lightweight ReID head contains a significantly lower number of parameters (≈ 131 K) compared to the external ReID module (≈ 25 M).

Finally, in addition to improve the performance measures for which we optimize Tracktor, DeepMOT consistently improves tracking measures such as IDF1 (↑1.17 improvement of **DeepMOT+ReIDhead** over **Vanilla+ReIDext**). We conclude that (i) training existing trackers using our proposed loss clearly improves the performance and (ii) we can easily extend existing trackers such as Tracktor to go beyond simple bounding box regression and incorporate the appearance module directly into the network. All modules are optimized jointly in a single training.

DeepMOT, Loss Ablation. Next, we perform several experiments in which we study the impact of different components of our proposed loss (Eq. 2.7) on the performance of Tracktor (**DeepMOT+ReIDhead**). We outline our results in

Table 2.4: Ablation study on the effect the training loss on Tracktor.

Training loss	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
Vanilla	60.20	89.50	71.15	35.13	27.80	276	31827	152
Smooth L_1	60.38	91.81	71.27	34.99	27.25	294	31649	164
$dMOTP$	60.51	91.74	71.75	35.41	**26.83**	291	31574	142
$dMOTA$	60.52	88.31	71.92	35.41	27.39	254	31597	142
$dMOTA$+$dMOTP$-I$\tilde{\text{D}}$S	60.61	**92.03**	72.10	35.41	27.25	222	31579	124
$dMOTA$+$dMOTP$	**60.66**	91.82	**72.32**	**35.41**	27.25	**218**	**31545**	**118**

Tab. 2.4. In addition to **Vanilla+ReIDext** (representing the best performance trained in Vanilla settings), we also report results obtained by training the same architecture using only the Smooth L_1 loss (see Fig. 2.4). We train the regression head with Smooth L_1 loss using a similar training procedure as for DeepMOT, (see Sec. 2.3.3), to regress predicted bounding boxes to the ones at the current time step of their associated tracks. This approach is limited in the sense that we cannot (directly) penalize FP, FN, and IDS.

The Smooth L_1 training, when compared to Vanilla, has a positive impact on almost all performance measures, except for MT, FP, and IDS. However, both Vanilla and Smooth L_1 are outperformed almost systematically for all performance measures by the various variants of the DeepMOT loss. Remarkably, when using the $dMOTA$ term in our loss, we significantly reduce the number of IDS and FP. Training with $dMOTP$ has the highest impact on MOTP, as it is the case when training with Smooth L_1. When only optimizing for $dMOTA$, we have a higher impact on the MOTA and IDF1 measure. Remarkably, when training with ($dMOTA$+$dMOTP$), we obtain a consistent improvement on all tracking evaluation measures with respect to Vanilla and Smooth L_1. Finally, we assess the impact of I$\tilde{\text{D}}$S, by setting the weight γ to 0 (Eq. 2.5) (line $dMOTA$+$dMOTP$-I$\tilde{\text{D}}$S). In these settings, the trackers exhibit a higher number of IDS compared to using the full loss, confirming that the latter is the best strategy.

MOT-by-SOT Ablation. Using DeepMOT, we can turn trainable SOT methods into trainable MOT methods by combining them with the track management mechanism (as explained in Sec. 2.4.3) and optimize their parameters using our loss.

Table 2.5: DeepMOT v.s. Smooth L_1 using MOT-by-SOT baselines and Tracktor.

	Training	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
GOTURN	Pre-trained	45.99	85.87	49.83	22.27	36.51	2927	39271	1577
	Smooth L_1	52.28	90.56	63.53	**29.46**	**34.58**	2026	36180	472
	DeepMOT	**54.09**	**90.95**	**66.09**	28.63	35.13	**927**	**36019**	**261**
SiamRPN	Pre-trained	55.35	87.15	66.95	33.61	**31.81**	1907	33925	356
	Smooth L_1	56.51	**90.88**	68.38	**33.75**	32.64	925	34151	167
	DeepMOT	**57.16**	89.32	**69.49**	33.47	32.78	**889**	**33667**	**161**
Tracktor	Vanilla	60.20	89.50	71.15	35.13	27.80	276	31827	152
	Smooth L_1	60.38	91.81	71.27	34.99	27.25	294	31649	164
	DeepMOT	**60.66**	**91.82**	**72.32**	**35.41**	**27.25**	**218**	**31545**	**118**

In Tab. 2.5, we outline the results of the two MOT-by-SOT baselines (GOTURN [70] and SiamRPN [102]). For both, we show the performance when using (i) a pre-trained network, (ii) a network fine-tuned using the Smooth L_1 loss, and (iii) the one trained with DeepMOT.

Based on the results outlined in Tab. 2.5, we conclude that training using the Smooth L_1 loss improves the MOTA for both SOTs (GOTURN: +6.29%, SiamRPN: +1.16%). Moreover, compared to models trained with Smooth L_1 loss, we further improve MOTA and reduce the number of IDS when we train them using DeepMOT. For GOTURN (SiamRPN), we record a MOTA improvement of 1.81% (0.65%) while reducing the number of IDS by 211 (6). We also outline the improvements comparing **Vanilla+ReIDext** Tracktor trained with Smooth L_1 loss, and **DeepMOT+ReIDhead** Tracktor trained using DeepMOT. These results further validate the merit and generality of our method for training deep multiple-object trackers.

MOTChallenge Benchmark Evaluation. We evaluate the trackers trained using our framework on the MOTChallenge benchmark (test set) using the best-performing configuration, determined previously using the validation set. During training and inference, we use the camera motion compensation module, as proposed by [7], for the three trained trackers. We discuss the results obtained on MOT15-17, as shown in Tab. 2.6. We follow the standard evaluation practice and compare our

Table 2.6: We establish a new state-of-the-art on MOT15-17 public benchmarks by using the proposed DeepMOT.

	Method	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
MOT17	DeepMOT-Tracktor	**53.7**	77.2	53.8	19.4	36.6	**11731**	247447	1947
	Tracktor [7]	53.5	78.0	52.3	19.5	36.6	12201	248047	2072
	DeepMOT-SiamRPN	52.1	**78.1**	47.7	16.7	41.7	12132	255743	2271
	SiamRPN [102]	47.8	76.4	41.4	17.0	41.7	38279	251989	4325
	DeepMOT-GOTURN	48.1	77.9	40.0	13.6	43.5	22497	266515	3792
	GOTURN [70]	38.3	75.1	25.7	9.4	47.1	55381	282670	10328
	eHAF [153]	51.8	77.0	**54.7**	**23.4**	37.9	33212	**236772**	1834
	FWT [71]	51.3	77.0	47.6	21.4	**35.2**	24101	247921	2648
	jCC [85]	51.2	75.9	54.5	20.9	37.0	25937	247822	**1802**
	MOTDT17 [26]	50.9	76.6	52.7	17.5	35.7	24069	250768	2474
	MHT_DAM [88]	50.7	77.5	47.2	20.8	36.9	22875	252889	2314
MOT16	DeepMOT-Tracktor	**54.8**	77.5	**53.4**	**19.1**	**37.0**	**2955**	**78765**	645
	Tracktor [7]	54.4	78.2	52.5	19.0	36.9	3280	79149	682
	DeepMOT-SiamRPN	51.8	78.1	45.5	16.1	45.1	3576	83699	641
	SiamRPN [102]	44.0	76.6	36.6	15.5	45.7	18784	82318	1047
	DeepMOT-GOTURN	47.2	78.0	37.2	13.7	46.1	7230	87781	1206
	GOTURN [70]	37.5	75.4	25.1	8.4	46.5	17746	92867	3277
	HCC [121]	49.3	**79.0**	50.7	17.8	39.9	5333	86795	**391**
	LMP [161]	48.8	**79.0**	51.3	18.2	40.1	6654	86245	481
	GCRA [120]	48.2	77.5	48.6	12.9	41.1	5104	88586	821
	FWT [71]	47.8	75.5	44.3	**19.1**	38.2	8886	85487	852
	MOTDT [26]	47.6	74.8	50.9	15.2	38.3	9253	85431	792
2D MOT 2015	DeepMOT-Tracktor	**44.1**	**75.3**	46.0	17.2	26.6	6085	26917	1347
	Tracktor [7]	**44.1**	75.0	46.7	18.0	**26.2**	6477	**26577**	1318
	DeepMOT-SiamRPN	33.3	74.6	32.7	9.3	43.7	7825	32211	919
	SiamRPN [102]	31.0	73.9	30.7	12.6	41.7	10241	31099	1062
	DeepMOT-GOTURN	29.8	**75.3**	27.7	4.0	66.6	3630	38964	524
	GOTURN [70]	23.9	72.8	22.3	3.6	66.4	7021	38750	965
	AP_HWDPL_p [25]	38.5	72.6	**47.1**	8.7	37.4	**4005**	33203	586
	AMIR15 [145]	37.6	71.7	46.0	15.8	26.8	7933	29397	1026
	JointMC [85]	35.6	71.9	45.1	**23.2**	39.3	10580	28508	457
	RAR15pub [47]	35.1	70.9	45.4	13.0	42.3	6771	32717	**381**

models to methods that are officially published on the MOTChallenge benchmark and peer-reviewed. For MOT17, we average the results obtained using the three sets of provided public detections (DPM [48], SDP [40] and Faster R-CNN [139]). As in [7], we use these public detections for track initialization and termination. Importantly, in the case of Tracktor, we do not use the internal detection mechanism of the network, but only public detections.

As can be seen in Tab. 2.6, DeepMOT-Tracktor establishes a new state-of-the-art on both MOT17 and MOT16. We improve over Tracktor (on MOT17 and MOT16, respectively) in terms of (i) MOTA (0.2% and 0.4%), (ii) IDF1 (1.5% and 0.9%) and (iii) IDS (125 and 37). On both benchmarks, Vanilla Tracktor is the second best-performing method, and our simple SOT-by-MOT baseline DeepMOT-SiamRPN is the third. We observe large improvements over our MOT-by-SOT pre-trained models and models trained using DeepMOT. For GOTURN, we improve MOTA by 9.8% and 9.7% and we significantly reduce the number of IDS by 6536 and 2071, for MOT17 and MOT16 respectively. A similar impact on DeepMOT-SiamRPN is observed.

We report the results for MOT15 as a supplementary material. Our key observations are:

(i) For the MOT-by-SOT baseline, we significantly improve over the trainable baselines (SiamRPN and GOTURN). deepMOT-SiamRPN increases MOTA for +2.3%, MOTP for +0.7% and IDF1 for +2.0%. Remarkably, deepMOT-SiamRPN suppresses 2,416 FP and 143 IDS. We observe similar performance gains for deepMOT-GOTURN.

(ii) deepMOT-Tracktor obtains results, comparative to the vanilla Tracktor [7]. Different from MOT16 and MOT17 datasets, we observe no improvements in terms of MOTA, which we believe is due to the fact that labels in MOT15 are very noisy, and vanilla Tracktor already achieves impressive performance. Still, we increase MOTP for 0.3% and reduce FP for 392.

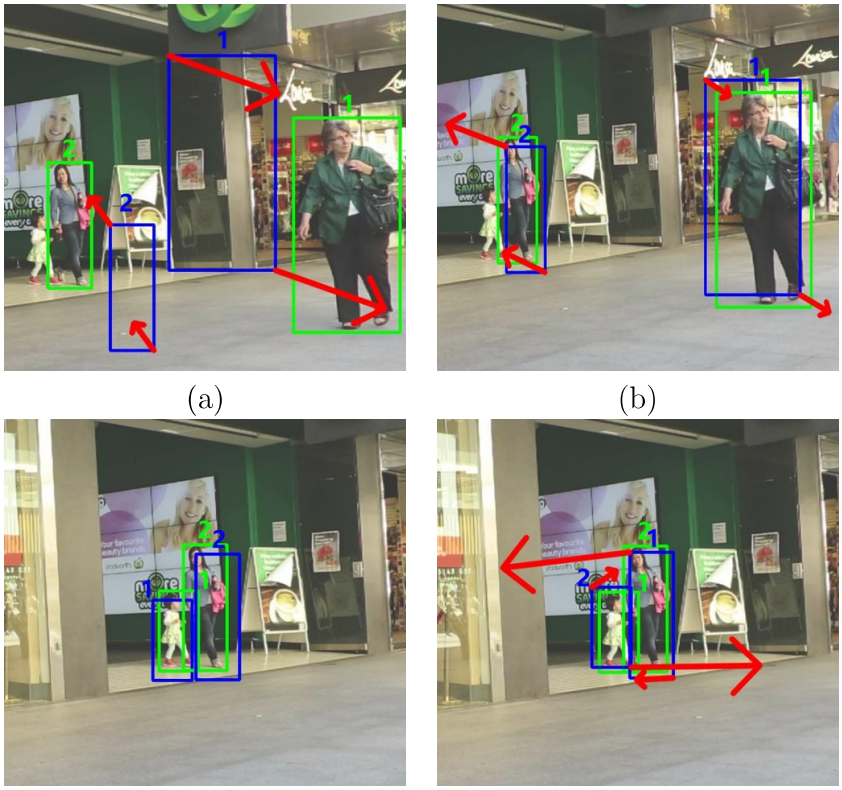


Figure 2.9: Visualization of negative gradients (direction and magnitude) from different terms in the proposed DeepMOT loss: (a) FP and FN (b) MOTP (c-d) IDS (compare (c) $t-1$ with (d) t). The predicted bounding boxes are shown in blue, the ground-truth ones are shown in green and the gradient direction is visualized using red arrows.

2.4.5 Training Gradient Visualization

The negative gradient should reflect the direction that minimizes the loss. In Fig. 2.9, we plot the negative gradient of different terms that constitute our DeepMOT loss w.r.t the coordinates of each predicted bounding box to demonstrate visually the effectiveness of our DeepMOT. In this example, we manually generated the cases that contain the FP, FN, or IDS. We observe that the negative gradient does encourage the tracks' bounding boxes to be close to those of their associated objects during the training.

2.5 Conclusion

In this work, we propose an end-to-end MOT training framework, based on a differentiable approximation of HA and CLEAR-MOT metrics. We experimentally demonstrate that our proposed MOT framework improves the performance of existing deep MOT methods. Thanks to our method, we set a new state-of-the-art score on the MOT16-MOT17 datasets. We believe that our method was the missing block for advancing the progress in the area of end-to-end learning for deep multiple-object tracking. We expect that our training module holds the potential to become a building block for training future multiple-object trackers.

2.6 Acknowledgments

For conducting this work, we gratefully acknowledge the mobility grant from the Department for Science and Technology of the French Embassy in Berlin (SST) and the French Institute for Research in Computer Science and Automation (Inria). We are grateful to the Dynamic Vision and Learning Group, the Technical University of Munich as the host institute, especially Guillem Brasó and Tim Meinhardt for the fruitful discussions. Finally, this research was partially funded by the Humboldt Foundation through the Sofja Kovalevskaja Award.

Chapter 3

TransCenter: Improving MOT Performance with Transformers

(a) Detection (b) Tracking

Figure 3.1: In our TransCenter, we propose to tackle the multiple-object tracking problem with transformers in an accurate and efficient manner: the dense non-overlapping representations provide sufficient and accurate detections through dense heatmap outputs as shown in (a); The sparse tracking queries, obtained from sampled features in object positions at the previous frame, efficiently produce the displacements of objects from the previous to the current time step, as shown in (b).

3.1 Introduction

Recent progress in computer vision using transformers [169] for tasks such as object detection [20, 110, 214], person re-identification (Re-ID) [68] or image super resolution [191], showed the benefit of the attention-based mechanism. Transformers are good at modeling simultaneously dependencies between different parts of the input and thus at taking global decisions. These advantages fit perfectly the underlying challenges of MOT, where current methods often struggle when modeling the interaction between objects, especially in crowded scenes. We, therefore, are motivated to investigate the use of a transformer-based architecture for MOT, enabling global estimations during finding trajectories thus reducing missed or noisy tracks.

Current MOT methods follow in principle the predominant tracking-by-detection paradigm where we first detect objects in the visual scene and associate them through time. The detection ability is critical for having a good MOT performance. To this end, MOT methods combined with probabilistic models [5, 112, 140] or deep convolutional architectures [7, 54, 132, 151, 176, 189, 201] usually use either an integrated or external detector to predict discrete bounding-box outputs. They are often based on overlapping predefined anchors that predict redundant outputs (which might create noisy detections) and thus need hand-crafted post-processing techniques (that might suppress correct detections) such as non-maximum suppression (NMS). Instead, concurrent MOT methods [123, 131] built on recent object detectors like DETR [20] leverage transformers where sparse noise-initialized queries are employed, which perform cross-attention with encoded images to output non-overlapping object-position predictions, thanks to the one-to-one assignment of objects and queries during training. However, with sparse fixed-number queries, the approach often suffers a lack of detections when the visual scene becomes crowded and needs to re-adjust the hyper-parameters of queries. For these reasons, we argue that having dense but non-overlapping representations for detection is beneficial in MOT, especially in crowded-scene MOT. Indeed, there exist center-based methods [203, 209] that output dense object-center heatmap predictions directly related to the input image. The center heatmap representations are pixel-related and thus inherit the one-to-one assignment of objects and queries

(pixels) without external assignment algorithms. However, the locality of CNN architecture limits the networks to explore the co-dependency of objects globally while transformers do. Therefore, we believe that the transformer-based approach with dense image-related (thus non-overlapping) representations is a better choice for building a powerful MOT method.

Designing a powerful transformer-based MOT with dense image-related representations is far from evident. The main drawback is the computational efficiency related to the quadratic complexity in transformers w.r.t the dense inputs. Therefore, both the encoder and the decoder in transformers should be carefully designed to build an efficient and powerful MOT method. For the encoder, the DETR [20] structure alleviates this issue by introducing a pre-feature extractor like ResNet-50 to extract lower-scale features before inputting the images to the transformers, but the CNN feature extractor contributes also to the network complexity. Deformable DETR reduces significantly the attention complexity but the ResNet backbone is still kept. Alternatively, recent efficient transformers like [175] discard the pre-feature extractor and input directly image patches, following the ViT structure [43]. They also reduce the attention complexity in transformers by a spatial-reduction attention, which makes building efficient transformer-based MOT possible.

Concerning the TransCenter Decoder, indeed, Deformable DETR reduces significantly in the decoder side the attention complexity but faced with dense queries, the calculation is still heavy for real-time applications. We recall the dense representations obtained from the outputs of the encoder already learn global self-dependency inside the encoder and have a size related to the input image. It is thus intuitive to directly output the dense representations for heatmap predictions that avoids the heavy dense detection cross-attention on the decoder side. Furthermore, MOT is more than detection but also needs to associate objects from one time step to another–tracking. The tracking happens in the decoder where we search the current object positions with the known previous ones. By this intuition, we design sparse tracking queries because the tracking has prior positional information and thus does not need to search every pixel for finding current object positions, which significantly speeds up the MOT method without losing accuracy.

To summarize, as roughly illustrated in Fig. 3.1, TransCenter tackles the MOT problem with transformers in an accurate while efficient way. It (1) introduces

dense image-related queries to deal with miss detections from insufficient sparse queries or over detections in overlapping anchors of current MOT methods, yielding better MOT accuracy; (2) solves the efficiency issue inherited in transformers with careful network designs, sparse tracking queries, and removal of useless external overheads, allowing TransCenter to perform MOT efficiently. Overall, this paper has the following contributions:

- We introduce TransCenter, the first center-based transformer framework for MOT, which is also among the first to show the benefits of using transformer-based architectures for MOT.
- We carefully explore different network structures to combine the transformer with center representations, specifically proposing *dense image-related multi-scale representations* that are mutually correlated within the transformer attention and produce *abundant but less noisy tracks, while keeping a good computational efficiency.*
- We extensively compare with up-to-date online MOT tracking methods, TransCenter sets *a new state-of-the-art* baseline both in MOT17 [124] (+4.0% Multiple-Object Tracking Accuracy, MOTA) and MOT20 [38] (+18.8% MOTA) by large margins, leading both MOT competitions by now in the published literature.
- Moreover, two more model options, TransCenter-Dual, which further boosts the performance in crowded scenes, and TransCenter-Lite, enhancing the computational efficiency, are provided for different requirements in MOT applications.

3.2 Related Works

3.2.1 Transformers in Multiple-Object Tracking

Transformer is first proposed by [169] for machine translation, and has shown its ability to handle long-term complex dependencies between entries in a sequence

by using multi-head attention mechanism. With its success in natural language processing, works in computer vision start to investigate transformers for various tasks, such as image recognition [43], Person Re-ID [68], realistic image generation [82], super resolution [191] and audio-visual learning [52, 53].

Object detection with Transformer (DETR) [20] can be seen as an exploration and correlation task. It is an encoder-decoder structure where the encoder extracts the image information and the decoder finds the best correlation between the object queries and the encoded image features with an attention module. The attention module transforms the inputs into *Query* (Q), *Key* (K), and *Value* (V) with fully-connected layers. Having Q, K, V, the attended features are calculated with the attention function [169]:

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{QK^T}{\sqrt{h}})V \tag{3.1}$$

where h is the hidden dimension of Q, K, and V. The attention calculation suffers from heavy computational and memory complexities w.r.t the input size: the feature maps extracted from a ResNet-50 [67] backbone are used to alleviate the problem. Deformable DETR [214] further tackles the issue by proposing a deformable attention inspired by [34], drastically speeding up the convergence ($10\times$) and reducing the complexity. The reduction of memory consumption allows in practice using multi-scale features to capture finer details, yielding better detection performance. However, the CNN backbone is still kept in Deformable DETR. This important overhead hinders the Transformer from being efficient. Alternatively, Pyramid Vision Transformer [175] (PVT) extracts the visual features directly from the input images and the attention is calculated with efficient spatial-reduction attention (SRA). Precisely, PVT follows the ViT [43] structure while the feature maps are gradually down-scaled with a patch embedding module (with convolutional layers and layer normalization). To reduce the quadratic complexity of Eq. 3.1 w.r.t the dimension d of Q, K, V, the SRA in PVT reduces beforehand the dimension of K, V from $\mathbb{R}^{d\times h}$ to $\mathbb{R}^{\frac{d}{r}\times h}$ (with $r > 1$, the scaling factor) and keeps the dimension of Q unchanged. The complexity can be reduced by r^2 times while the dimension of the output attended features remains unchanged, boosting the efficiency of transformer attention modules.

Transformers are still new in MOT. Before transformers, some attempts with simple attention-based modules have been introduced for MOT. Specifically, [61] proposes a target and distractor-aware attention module to produce more reliable appearance embeddings, which also helps suppress detection drift and [173] proposes hand-designed spatial and temporal correlation modules to achieve long-range information similar to what transformers inherit. After the success in detection using transformers, two *concurrent works* directly apply transformers on MOT based on the (deformable) DETR framework. First, Trackformer [123] builds directly from DETR [20] and is trained to propagate the queries through time. Second, Transtrack [131] extends [214] to MOT by adding a decoder that processes the features at $t-1$ to refine previous detection positions. Importantly, both methods stay in the DETR framework with sparse queries and extend it for tracking. However, recent literature [114, 203, 209] also suggests that point-based tracking may be a better option for MOT while the use of pixel-level dense queries with transformers to predict dense heatmaps for MOT has never been studied. In addition, we question the direct transfer from DETR to MOT as concurrent works do [123, 131]. Indeed, the sparse queries without positional correlations might be problematic in two folds. Firstly, the insufficient number of queries could cause severe miss detections thus false negatives (FN) in tracking. Secondly, queries are highly overlapping, and simply increasing the number of non-positional-correlated queries may end up having many false detections then false positives (FP) in tracking. All above motivate us to investigate a better transformer-based MOT framework with careful designs considering existing drawbacks. We thus introduce TransCenter, achieving state-of-the-art performance.

3.3 TransCenter

TransCenter proposes to tackle the MOT task with the TransCenter Decoder dedicated to the two main tasks: detection and temporal association. Different from previous transformer-based MOT methods, TransCenter questions the use of sparse queries without positional correlations (i.e. noise initialized), and explores the use of image-related dense queries producing dense representations within transformers. To that aim, we introduce the query learning networks (QLN) responsible for

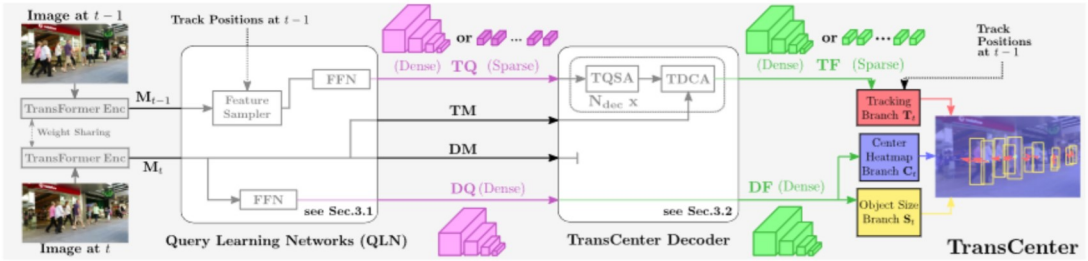


Figure 3.2: Generic pipeline of TransCenter and different variants: Images at t and $t-1$ are fed to the transformer encoder (*DETR-Encoder* or *PVT-Encoder*) to produce multi-scale memories $\mathbf{M}_t$ and $\mathbf{M}_{t-1}$ respectively. They are passed (together with track positions at $t-1$) to the *Query Learning Networks (QLN)* operating in the feature dimension channel. QLN produce (1) dense pixel-level multi-scale detection queries–**DQ**, (2) detection memory–**DM**, (3) (sparse or dense) tracking queries–**TQ**, (4) tracking memory–**TM**. For associating objects through frames, the TransCenter Decoder performs cross attention between **TQ** and **TM**, producing Tracking Features–**TF**. For detection, the TransCenter Decoder either calculates the cross attention between **DQ** and **DM** or directly outputs **DQ** (in our efficient versions, TransCenter and TransCenter-Lite, see Sec. 3.3), resulting in Detection Features–**DF** for the output branches, $\mathbf{S}_t$ and $\mathbf{C}_t$. **TF**, together with object positions at $t-1$ (sparse **TQ**) or center heatmap $\mathbf{C}_{t-1}$ (omitted in the figure for simplicity) and **DF** (dense **TQ**), are used to estimate image center displacements $\mathbf{T}_t$ indicating for each center its displacement in the adjacent frames (red arrows). We detail *our choice* (TransCenter) of QLN and TransCenter Decoder structures in the figure. Other designs of QLN and TransCenter Decoder are detailed in Fig. 3.3 and Fig. 3.4. Arrows with dotted line are only necessary for models with sparse **TQ**.

converting the output of the encoder into the input of the decoder. Different architectures for the QLN and the TransCenter Decoder are possible and their choices should take both the computational efficiency and accuracy into account.

While exploiting dense representations from dense queries (visualized in Sec. 3.5.2) can help detect sufficiently the objects, especially in the case of crowded scenes, the design of dense queries is not trivial. Indeed, the quadratic increase of calculation complexity in transformers should be solved, and randomly initialized dense queries may lead to FP. The fact that the dense queries are image-related has three prominent advantages: (1) the queries are multi-scale and exploit the multi-resolution structure of the encoder, allowing for very small targets to be captured by those queries; (2) image-related dense detection queries also make

the network more flexible since no parameter is needed to (re-)define the number of queries like in [131] based on the scenes; (3) the query-pixel correspondence discards the time-consuming Hungarian matching [93] (e.g. 0.45s per training step, i.e. taking 6.9h more for training MOT20 validation set for 50 epochs) for the query-ground-truth object association during training. Up to our knowledge, we are the first to explore the use of image-related dense query feature maps that scale with the input image size. Meanwhile, we solve the efficiency issue through careful network designs handling the dense representation of queries, which provides an accurate and efficient MOT method even in crowded-scene MOT.

A generic pipeline of TransCenter is illustrated in Fig. 3.2. The RGB images at t and $t-1$ are input to the weight-shared Transformer encoder and we obtain the dense multi-scale attended features called memories $\mathbf{M}_t$ and $\mathbf{M}_{t-1}$, respectively. They are the inputs of the QLN. The latter produces two sets of output pairs, tracking queries (**DQ**) and memory (**DM**) for detecting the objects at time t, and tracking queries (**TQ**) and memory (**TM**) for associating the objects at t with previous time step $t-1$. Furthermore, TransCenter Decoder, leveraging the deformable transformer [214], is used to correlate the detection/tracking queries with the memories (**DM**/**TM**). To elaborate, **TQ** interacts with **TM** in the cross-attention module of the TransCenter Decoder, resulting in the tracking features (**TF**). Similarly, the detection features (**DF**) are the output of the cross-attention between **DQ** and **DM**. To produce the output dense representations, **DF** are used to estimate the object size $\mathbf{S}_t$ and the center heatmap $\mathbf{C}_t$. **TF** are used to estimate the tracking displacement $\mathbf{T}_t$.

One can argue that the downside of using dense queries is the associated memory consumption and computational efficiency. A prominent problem for obtaining these dense queries is the use of the deformable DETR encoder (including the CNN feature extractor ResNet 50 [67]), as previously done in the literature. Instead, TransCenter uses PVT [175] as its encoder, *PVT-Encoder*. The reasons are 3-fold: (1) it discards the ResNet-50 backbone and uses efficient attention heads [175], reducing significantly the network complexity; (2) it has flexible scalability by modifying h and block structures (we use B0 (*PVT-Lite-Encoder*) in TransCenter-Lite and B2 (*PVT-Encoder*) for TransCenter, see details in [175] and Sec.3.4.1.); (3) its feature pyramid structure is suitable for building dense pixel-level multi-scale

queries.

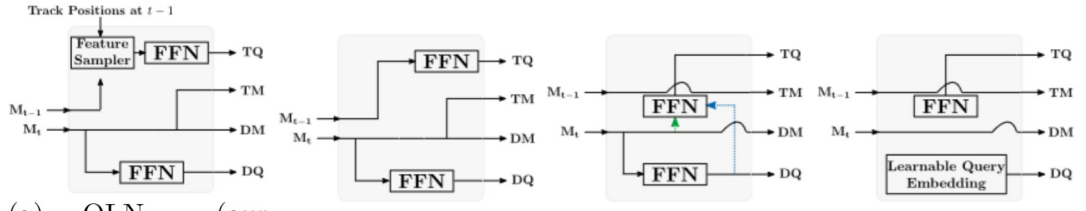
Once the TransCenter Encoder extracts the dense memory representations $\mathbf{M}_t$ and $\mathbf{M}_{t-1}$, we propose several design choices for QLN (see Section 3.3.1) and for the TransCenter Decoder (see Section 3.3.2), to alleviate the complexity issue. In particular, unlike **DQ** that are needed to represent every pixel (i.e. dense) to discover new objects, we demonstrate that **TQ** can be sparse (visualized in Sec. 3.5.3) since we have the position prior of objects at $t-1$ to search their current positions at t, i.e. to find the corresponding object given its potions at the previous frame, we do not need to search every pixel. Thus, the discretization of tracking queries based on object positions at $t-1$ (roughly from 14k to less than 500 depending on the number of tracks) is beneficial and can significantly speed up the tracking attention calculation in the TransCenter Decoder. Regarding the detection side, the cross attention module between the dense detection queries (**DQ**) and the detection memory (**DM**) is beneficial in terms of performance but with important computational loads. To this end, we study the impact on the computational efficiency and the accuracy of the detection cross-attention (introducing *TransCenter-Dual*). Finally, a lighter version (with PVT-Lite-Encoder) of TransCenter is also studied and denoted as *TransCenter-Lite*. In the next two sections, we detail the design choices of the QLN and the TransCenter Decoder, we provide the details of the final branches (see Section 3.3.3) and as well as the training losses (see Section 3.3.4).

3.3.1 QLN: Query Learning Networks

As discussed previously in Sec. 3.3, **DQ** are dense and image-related that discover object positions precisely and abundantly. Different from the detection queries,**TQ** are sparse aiming at finding links between two different frames, and thus **TQ** and **TM** should be produced by input features from different time steps. Based on these attributes, the chosen QLN_{S-}[1] produces **DQ** by $\mathbf{M}_t$ (attended features from image t) after a FFN (feed-forward network with fully-connected

[1] "S" means sparse, "−" means that the features are sampled from $t-1$. It is counter-intuitive to have QLN_S because at time t, we know neither the number of tracked objects (tracks) at t nor their positions, we cannot thus sample features with track positions.

layers). For **TQ**, QLN_{S-} samples object features (with a feature sampler using bilinear interpolation) from $\mathbf{M}_{t-1}$ using track positions at $t-1$, while outputting **TM** from features at a different time step, $\mathbf{M}_t$.



(a) QLN_{S-} (our choice) (b) QLN_{D-} (c) QLN_{D} (d) QLN_{E}

Figure 3.3: Query Learning Networks (QLN): TransCenter uses QLN_{S-} as its query learning network, producing sparse tracking queries from information at $t-1$. Different structures of QLN are studied such as QLN_{D-}, QLN_{D} (QLN_{M_t} in green arrow and QLN_{DQ} in blue arrow), and QLN_{E}, detailed in Sec. 3.3.1. Best seen in color.

To demonstrate the effectiveness of our choice (QLN_{S-}), we ablate with different QLN variants in Sec. 3.4.5. To elaborate, to demonstrate that having sparse tracking queries is possible, we study QLN_{D-} following a similar strategy as QLN_{S-} but produces *dense* (with the subscript "D") **TQ** without the feature sampler. Both **TQ** in QLN_{S-} and QLN_{D-} are from $\mathbf{M}_{t-1}$, interacting with **TM** from $\mathbf{M}_t$ in the TransCenter Decoder. Inversely, based the aforementioned intuition that **TQ** and **TM** should be from different time steps, we compare to QLN_{D}, of two variants – QLN_{M_t} and QLN_{DQ}, outputting dense **TQ** from $\mathbf{M}_t$ or **DQ**, respectively. They interact with **TM** from $\mathbf{M}_{t-1}$. Finally, for demonstrating the superior performance of our dense image-related queries, we compare to QLN_{E} similar to QLN_{DQ} but the queries are from learnable embeddings (with the subscript "E") initialized with noise as in DETR [20], instead of the memories and thus not related to the images.

In summary, QLN are carefully designed and ablated to produce dense detection queries relative to the input image and sparse tracking queries for accurate and efficient MOT with transformers.

[2]We note that discarding TDCA is impossible since the tracking queries at $t-1$ or t should interact with **TM** for searching objects at t or $t-1$.

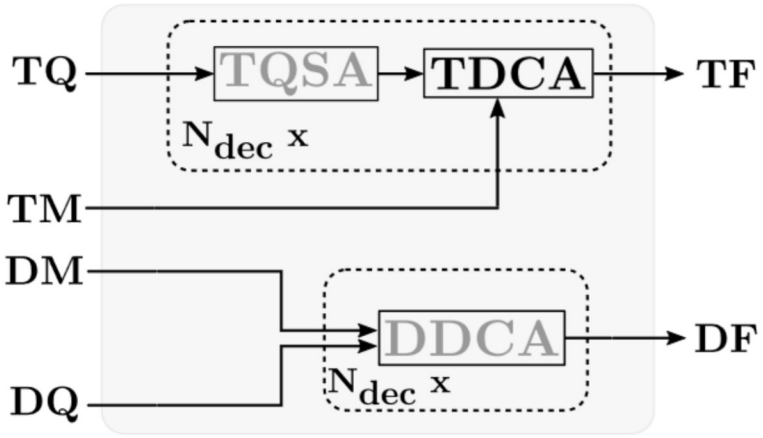


Figure 3.4: TransCenter Decoder is used to handle tracking queries **TQ** and detection queries **DQ**. The detection attention correlates **DQ** and **DM** with the attention modules to detect objects. The tracking attention correlates **TQ** and **TM** (information from different time steps) to learn the displacements between frames of the detected objects at $t-1$ (i.e. tracks). TransCenter Decoder has three main modules TQSA, DDCA and TDCA (defined in Sec. 3.3.2). Different versions of TransCenter Decoder depending on discarding the DDCA (TQSA) or not, are denoted as *Single-(TQSA)* or *Dual-(TQSA)* decoder [2]. TransCenter uses *Single-TQSA* considering the efficiency-accuracy tradeoff. The choice is based on the ablation of the aforementioned variants in Sec. 3.4.5. $\mathbf{N}_{dec}$ is the number of decoder layers.

3.3.2 TransCenter Decoder

To successfully find object trajectories, a MOT method should not only detect the objects but also associate them across frames. To do so, TransCenter Decoder tackles in parallel with the two sub-tasks: detection and tracking. TransCenter Decoder consists of Tracking Deformable Cross-Attention (TDCA), and Detection Deformable Cross-Attention (DDCA). We also introduce Tracking Query Self-Attention (TQSA) to enhance the interactions among input sparse queries through a multi-head self-attention module, knowing that the overhead is acceptable because the queries are (tracking) sparse in TransCenter. TDCA and DDCA calculate cross attention between (tracking or detection) queries–**TQ** or **DQ** and (tracking or detection) memories–**TM** or **DM**, outputting **TF** or **DF**, respectively. They leverage a deformable cross attention module in [214] with linear complexity w.r.t. input size.

From the efficiency perspective, the use of the multi-head attention modules as in traditional transformers [169] as in DETR [20] implies a complexity growth that is quadratic with the input size. Of course, this is undesirable and would limit the scalability and usability of the method. To mitigate this, we resort to the deformable multi-head attention [214]–Deformable Cross-Attention (DCA), where the queries are input to produce sampling offsets for displacing the input reference points. The reference points are from either the track position at $t-1$ for tracking in TDCA or the pixel coordinates of the dense queries for detection in DDCA [3]. The displaced coordinates are used to locate and sample features in **DM** or **TM** (Value). The input queries also produce in DCA the attention weights for merging sampled features.

However, the cost of calculating the cross attention between **DQ** and **DM** is still not negligible because of their multi-scale image resolutions. To solve this, we demonstrate in Sec. 3.4.5 (*Single* v.s. *Dual* decoder, i.e. discarding DDCA or not) that it is possible to output directly **DQ** as **DF** for output-branch predictions, thanks to the dense pixel-level nature of **DQ**, with an acceptable loss of accuracy as expected. Differently, under this sparse nature of **TQ** in the proposed TransCenter, we enhance the interaction among queries, TQSA is added before TDCA.

To conclude, we choose to use a *Single-TQSA* decoder for the cross attention of **TQ** and **TM** while we directly use **DQ** for the output branches. This is possible thanks to the *sparse* **TQ** and *dense* **DQ**, which yields a good balance between computational efficiency and accuracy. Other variants are ablated in Sec. 3.4.5 and described in Fig. 3.4.

3.3.3 The Center, the Size and the Tracking Branches

Given **DF** and **TF** as inputs from the TransCenter Decoder, we use different branches to output object center heatmap $\mathbf{C}_t$ and their box size $\mathbf{S}_t$ as well as the tracking displacements $\mathbf{T}_t$. **DF** contain feature maps of four different resolutions, namely $1/32, 1/16, 1/8$, and $1/4$ of the input image resolution. For the center heatmap and the object size, the feature maps at different resolutions are combined

[3] The reference points are omitted in the figures for simplicity.

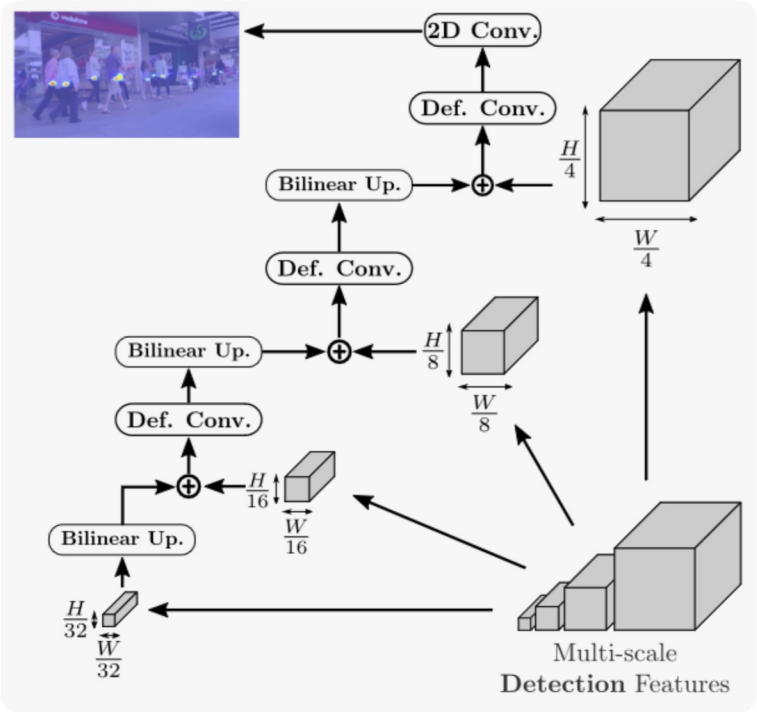


Figure 3.5: Overview of the center heatmap branch. The multi-scale detection features are up-scaled (bilinear up.) and merged via a series of deformable convolutions (Def. Conv., the ReLU activation is omitted for simplicity) [34], into the output center heatmap. A similar strategy is followed for the object size and the tracking branches.

using deformable convolutions [34] and bilinear interpolation, following the architecture shown in Fig. 3.5, into a feature map of 1/4 of the input resolution, and finally into $\mathbf{C}_t \in [0,1]^{H/4 \times W/4}$ (H and W are the input image height and width, respectively) and $\mathbf{S}_t \in \mathbb{R}^{H/4 \times W/4 \times 2}$ (the two channels of $\mathbf{S}_t$ encode the object width and height). Regarding the tracking branch, the tracking features **TF** are sparse of size depending on the number of tracks at $t-1$ where one tracking query feature corresponds to one track at $t-1$. **TF**, together with object positions at $t-1$ (sparse **TQ**) or center heatmap $\mathbf{C}_{t-1}$ and **DF** (dense **TQ**), are input to two fully-connected layers with ReLU activation. They predict the horizontal and vertical displacements $\mathbf{T}_t$ of tracks $t-1$ in the adjacent frames.

3.3.4 Model Training

The model training is achieved by jointly learning a 2D classification task for the object center heatmap and a regression task for the object size and tracking displacements, covering the branches of TransCenter. For the sake of clarity, in this section, we will drop the time index t.

Center Focal Loss. In order to train the center branch, we need first to build the ground-truth heatmap response $\mathbf{C}^* \in [0,1]^{H/4 \times W/4}$. As done in [209], we construct $\mathbf{C}^*$ by considering the maximum response of a set of Gaussian kernels centered at each of the $K > 0$ ground-truth object centers. More formally, for every pixel position (x, y) the ground-truth heatmap response is computed as:

$$\mathbf{C}^*_{xy} = \max_{k=1,\dots,K} G((x,y),(x_k,y_k);\sigma), \tag{3.2}$$

where (x_k, y_k) is the ground-truth object center, and $G(\cdot,\cdot;\sigma)$ is the Gaussian kernel with spread σ. In our case, σ is proportional to the object's size, as described in [95]. Given the ground-truth $\mathbf{C}^*$ and the inferred $\mathbf{C}$ center heatmaps, the center focal loss, L_{C} is formulated as:

$$L_{\text{C}} = \frac{1}{K} \sum_{xy} \begin{cases} (1-\mathbf{C}_{xy})^{\alpha} \log(\mathbf{C}_{xy}) & \mathbf{C}^*_{xy} = 1, \\ (1-\mathbf{C}^*_{xy})^{\beta} (\mathbf{C}_{xy})^{\alpha} \log(1-\mathbf{C}_{xy}) & \text{otherwise.} \end{cases} \tag{3.3}$$

where the scaling factors are $\alpha = 2$ and $\beta = 4$, see [203].

Sparse Regression Loss. The values of $\mathbf{S}$ is supervised only on the locations where object centers are present, i.e. $C^*_{xy} = 1$ using a L_1 loss:

$$L_{\text{S}} = \frac{1}{K} \sum_{xy} \begin{cases} \left\| \mathbf{S}_{xy} - \mathbf{S}^*_{xy} \right\|_1 & \mathbf{C}^*_{xy} = 1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.4}$$

The formulation of L_{T} for $\mathbf{T}$ is analogous to L_{S} but using the tracking output and ground-truth displacement, instead of the object size. To complete the sparsity of L_{S}, we add an extra L_1 regression loss, denoted as L_R with the bounding boxes computed from $\mathbf{S_t}$ and ground-truth centers. To summarize, the overall loss is

formulated as the weighted sum of all the losses, the weights are chosen according to the numeric scale of each loss:

$$L = L_{\mathrm{C}} + \lambda_{\mathrm{S}} L_{\mathrm{S}} + \lambda_{\mathrm{T}} L_{\mathrm{T}} + \lambda_{\mathrm{R}} L_{\mathrm{R}} \tag{3.5}$$

3.4 Experimental Evaluation

3.4.1 Implementation Details

Inference with TransCenter. Once the method is trained, we detect objects by selecting the maximum responses from the output center heatmap $\mathbf{C}_t$. Since the datasets are annotated with bounding boxes, we need to convert our estimates into this representation. In detail, we apply (after a max pooling) a threshold τ (0.3 for MOT17 and 0.4 for MOT20) to the center heatmap, thus producing a list of center positions $\{\mathbf{c}_{t,k}\}_{k=1}^{K_t}$. We extract the object size $\mathbf{s}_{t,k}$ associated to each position $\mathbf{c}_{t,k}$ in $\mathbf{S}_t$. The set of detections produced by TransCenter is denoted as $\mathbf{D}_t = \{\mathbf{c}_{t,k}, \mathbf{s}_{t,k}\}_{k=1}^{K_t}$. In parallel, for associating objects through frames (tracking), given the positions of tracks at $t-1$, we can estimate the object position in the current frame by extracting the corresponding displacement estimate $\mathbf{t}_{t,k}$ from $\mathbf{T}_t$. Therefore, we can construct a set of *tracked positions* $\tilde{\mathbf{P}}_t = \{\mathbf{c}_{t-1,k} + \mathbf{t}_{t,k}, \mathbf{s}_{t,k}\}_{k=1}^{\tilde{K}_t}$. Finally, we use the Hungarian algorithm [93] to match the tracked positions – $\tilde{\mathbf{P}}_t$ and the detection at t – $\mathbf{D}_t$. The matched detections are used to update the tracked object positions at t. The birth and death processes are naturally integrated in TransCenter: Detections not associated to any tracked object give birth to new tracks, while unmatched tracks are put to sleep for at most $T = 60$ frames before being discarded. An external Re-ID network is often used in MOT methods [7] to recover tracks in sleep, which is proven unnecessary in our experiment in Sec. 4.4.3. We also assess inference speed (fps) in the testset results either obtained from [202] or tested under the same GPU setting.

Network and Training Parameters. The input images are resized to 640×1088 with padding in TransCenter (608×1088 in TransCenter-Lite). In TransCenter, the PVT-encoder has $[3, 4, 6, 3]$ layers ($[2, 2, 2, 2]$ in PVT-Lite-Encoder) for each image

level (see Fig. 2.4) and the corresponding hidden dimension $h = [64, 128, 320, 512]$ ($h = [32, 64, 160, 256]$ in PVT-Lite-Encoder). $h = 256$ for the TransCenter Decoder with eight attention heads and six layers (four layers in TransCenter-Lite). TransCenter is trained with loss weights $\lambda_S = 0.1$, $\lambda_R = 0.5$ and $\lambda_T = 1.0$ by the AdamW optimizer [117] with learning rate 2e$-$4. The training converges at around 50 epochs, applying learning rate decay of 1/10 at the 40th epoch. The entire network is pre-trained on the pedestrian class of COCO [167] and then fine-tuned on the respective MOT dataset [38, 124]. We also present the results fine-tuning with extra data like CrowdHuman dataset [152] (see Sec. 3.4.3 for details).

3.4.2 Protocol

Datasets and Detections. We use the standard split of the MOT17 [124] and MOT20 [38] datasets and the test evaluation is obtained by submitting the results to the MOTChallenge website. The MOT17 testset contains 2,355 trajectories distributed in 17,757 frames. MOT20 testset contains 1,501 trajectories within only 4,479 frames, which leads to a much more challenging crowded-scene setting. We evaluate TransCenter both under public and private detections. When using public detections, we limit the maximum number of birth candidates at each frame to be the number of public detections per frame, as in [123, 209]. The selected birth candidates are those closest to the public detections with IOU larger than 0. When using private detections, there are no constraints, and the detections depend only on the network's detection capacity, the use of external detectors, and more importantly, the use of extra training data. For this reason, we regroup the results by the use of extra training datasets as detailed in the following. In addition, we evaluated our TransCenter in the KITTI data under the autonomous driving setting, KITTI dataset contains annotations of cars and pedestrians in 21 and 29 video sequences in the train and test set, respectively. For the results of KITTI dataset, we also used [163] as extra data.

Extra Training Data. To fairly compare with the state-of-the-art methods, we clearly denote the extra data used to train each method (including several pre-prints listed in the MOTChallenge leaderboard, which are marked with * in our result

tables):[4] **ch** for CrowdHuman [152], **pt** for PathTrack [148], **re1** for the combination of Market1501 [205], CUHK01 and CUHK03 [105] person re-identification datasets, **re2** replaces CUHK01 [105] with DukeMTMC [142], **5d1** for the use of five extra datasets (ETH [45], Caltech Pedestrian [41, 42], CityPersons [199], CUHK-SYS [186], and PRW [206]), **5d1+CH** is the same as **5d1** plus CroudHuman. (**5d1+CH**) uses the tracking/detection results of FairMOT [203] (trained within the **5d1+CH** setting), and **no** for using no extra dataset.

Table 3.1: Results on MOT17 testset: the left and right halves of the table correspond to public and private detections respectively. The cell background color encodes the amount of extra-training data: green for none, orange for one extra dataset, red for (more than) five extra datasets. Methods with * are not associated to a publication. The best result within the same training conditions (background color) is underlined. The best result among published methods is in **bold**. Best seen in color.

	Public Detections										Private Detections									
Method	Data	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓	FPS↑	Data	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓	FPS↑
MOTDT17 [27]	re1	50.9	76.6	52.7	17.5	35.7	24,069	250,768	2,474	**18.3**										
*UnsupTrack [84]	pt	61.7	78.3	58.1	27.2	32.4	16,872	197,632	1,864	<17.5										
GMT_CT [65]	re2	61.5		**66.9**	26.3	32.1	**14,059**	200,655	**2,415**											
TrackFormer [123]	ch	62.5		60.7	29.8	26.9	14,966	206,619	1,189	6.8										
SiamMOT [154]	ch	65.9		63.5	34.6	23.9	18,098	170,955	3,040	12.8										
*MOTR [198]	ch	67.4		67.0	34.6	24.5	32,355	149,400	1,992	7.5										
TrackFormer [123]											ch	65.0		63.9	45.6	13.8	70,443	123,552	3,528	6.8
CenterTrack [209]											ch	67.8	78.4	64.7	34.6	24.6	**18,489**	160,332	**3,039**	**17.5**
TraDeS [81]											ch	69.1		63.9	36.4	21.5	20,892	150,060	3,555	**17.5**
PermaTrack [166]											ch	73.8		68.9	43.8	17.2	28,998	114,104	3,699	11.9
*TransTrack [131]											ch	74.5	80.6	63.9	46.8	11.3	28,323	112,137	3,663	10.0
TransCenter	ch	**75.9**	**81.2**	65.9	**49.8**	**12.1**	30,190	**100,999**	4,626	11.7	ch	**76.2**	**81.1**	**65.5**	**53.5**	**7.9**	40,101	**88,827**	5,394	11.8
GSDT [176]											5d1	66.2	79.9	68.7	40.8	18.3	43,368	144,261	3,318	4.9
SOTMOT [114]	5d1	62.8		**67.4**	24.4	33.0	**6,556**	201,319	**2,017**	**16.0**	5d1	71.0		71.9	42.7	15.3	39,537	118,983	5,184	16.0
GSDT_V2 [176]											5d1	73.2		66.5	41.7	17.5	**26,397**	120,666	3,891	4.9
CorrTracker [173]											5d1	**76.5**		73.6	47.6	12.7	29,808	99,510	3,369	15.6
FairMOT [203]											5d1+CH	73.7	81.3	72.3	43.2	17.3	27,507	117,477	3,303	**25.9**
*RelationTrack [194]											5d1+CH	73.8	81.0	74.7	41.7	23.2	27,999	118,623	1,374	7.4
*CSTrack [109]											5d1+CH	74.9	80.9	72.6	41.5	17.5	23,847	114,303	3,567	15.8
MLT [201]											(5d1+CH)	75.3	**81.7**	**75.5**	49.3	19.5	27,879	109,836	**1,719**	5.9
*FUFET [151]											(5d1+CH)	76.2	81.1	68.0	51.1	13.6	32,796	98,475	3,237	6.8
TransCenter	5d1+CH	**76.0**	**81.4**	65.6	**47.3**	**15.3**	28,369	**101,988**	4,972	11.7	5d1+CH	76.4	81.2	65.4	**51.7**	**11.6**	37,005	**89,712**	6,402	10.9
TretrD17 [189]	no	53.7	77.2	53.8	19.4	36.6	11,731	247,447	1,947	<2.0										
Tracktor [7]	no	53.5	78.0	52.3	19.5	36.6	12,201	248,047	2,072	<2.0										
Tracktor++ [7]	no	56.3	78.8	55.1	21.1	35.3	8,866	235,449	1,987	<2.0										
GSM_Tracktor [115]	no	56.4	77.9	57.8	22.2	34.5	14,379	230,174	1,485	8.7										
TADAM [61]	no	59.7		58.7			**9,676**	216,029	1,930											
CenterTrack [209]	no	61.5	78.9	59.6	26.4	31.9	14,076	200,672	2,583	17.5										
*FUFET [151]	no	62.0		59.5	27.8	31.5	15,114	196,672	2,621	6.8										
ArTIST-C [147]	no	62.3		59.7	29.1	34.0	19,611	191,207	2,062	<17.5										
MAT [64]	no	67.1	**80.8**	**69.2**	38.9	26.4	22,756	161,547	**1,279**	9.0										
MTP [22]	no	51.5		54.9	20.5	35.5	29,623	241,618	2,563	20.1	no	55.9		60.4	20.5	36.7	**8,653**	238,853	**1,188**	20.1
ChainedTracker [132]											no	66.6	78.2	57.4	32.2	24.2	22,284	160,491	5,529	6.8
QDTrack [129]	no	64.6	79.6	65.1	32.3	28.3	14,103	18,2998	2,652	**20.3**	no	68.7	79.0	**66.3**	40.6	21.9	26,589	14,6643	3,378	**20.3**
TransCenter	no	**71.9**	80.5	64.1	**44.4**	**18.6**	27,356	**126,860**	4,118	11.9	no	**72.7**	**80.3**	64.0	**48.7**	**14.0**	33,807	**115,542**	4,719	11.8

Metrics. Standard MOT metrics such as MOTA (Multiple Object Tracking Accuracy) and MOTP (Multiple Object Tracking Precision) [9] are used: MOTA is mostly used since it reflects the average tracking performance including the number

[4] COCO [167] and ImageNet [144] are not considered as extra data according to the MOTchallenge [38, 124].

of FP (False positives, predicted bounding boxes not enclosing any object), FN (False negatives, missing ground-truth objects) and IDS [106] (Identities of predicted trajectories switch through time). MOTP evaluates the quality of bounding boxes from successfully tracked objects. Moreover, we also evaluate on IDF1 [142] (the ratio of correctly identified detections over the average number of ground-truth objects and predicted tracks), MT (the ratio of ground-truth trajectories that are covered by a track hypothesis more than 80% of their life span), and ML (less than 20% of their life span).

3.4.3 Testset Results and Discussion

MOT17. Tab. 3.1 presents the results obtained on the MOT17 testset. The first global remark is that most state-of-the-art methods do not evaluate under both public and private detections, and under different extra-training data settings, while we do. Secondly, TransCenter *sets new state-of-the-art performance compared to all other methods*, in terms of MOTA, under **CH** and no-extra training data conditions, both for public and private detections. Precisely, the increase of MOTA w.r.t. the state-of-the-art methods is of 8.5% and 4.8% (both including unpublished methods by now) for the public detection setting under **CH** and no-extra training data, and of 1.7% and 4.0% for the private detection setting, respectively. The superiority of TransCenter is remarkable in most of the metrics. We can also observe that TransCenter trained with no extra-training data outperforms, not only the methods trained with no extra data but also some methods trained with one extra dataset (in terms of MOTA for both public and private detections). Similarly, TransCenter trained on **ch** performs better than seven methods trained with five or more extra datasets in the private setting, comparable to the best result in **5d1+CH** (-0.3% MOTA), showing that TransCenter is less data-hungry. Moreover, trained with **5d1+CH**, the performance is further improved while running at around 11 fps. Overall, these results confirm our hypothesis that TransCenter with dense detection representations and sparse tracking representations produced by global relevant queries in transformers is a better choice.

MOT20. Tab. 3.2 reports the results obtained in MOT20 testset. *In all settings*, similar to the case in MOT17, TransCenter *leads the competition by a large margin*

Table 3.2: Results on MOT20 testset: the table is structured following the same principle as Tab. 3.1. Methods with * are not associated to a publication. The best result within the same training conditions (background color) is underlined. The best result among published methods is in **bold**. Best seen in color.

	Public Detections										Private Detections									
Method	Data	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓	FPS ↑	Data	MOTA ↑	MOTP ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓	FPS ↑
*UnsupTrack [84]	**pt**	53.6	80.1	50.6	30.3	25.0	6,439	231,298	2,178	<17.5										
*TransTrack [131]											**ch**	64.5	80.0	59.2	49.1	13.6	28,566	151,377	3.565	7.2
TransCenter	**ch**	**72.8**	**81.0**	**57.6**	**65.5**	**12.1**	**28,026**	**110,312**	**2,621**	**8.4**	**ch**	**72.9**	**81.0**	**57.7**	**66.5**	**11.8**	**28,596**	**108,982**	**2,625**	**8.7**
CorrTracker [173]											**5d1**	65.2		69.1	66.4	8.9	79,429	**95,855**	5.183	8.5
GSDT_V2 [176]											**5d1**	67.1		67.5	53.1	13.2	31,507	135,395	3.230	0.9
GSDT [176]											**5d1**	67.1	79.1	67.5	53.1	13.2	31,913	135,409	3.131	0.9
SOTMOT [114]											**5d1**	68.6		**71.4**	**64.9**	**9.7**	57,064	101,154	4.209	8.5
FairMOT [203]											**5d1+CH**	61.8	78.6	**67.3**	**68.8**	**7.6**	103.440	**88,901**	5.243	**13.2**
*CSTrack [109]											**5d1+CH**	66.6	78.8	68.6	50.4	15.5	25,404	144,358	3.196	4.5
*RelationTrack [194]											**5d1+CH**	67.2	79.2	70.5	62.2	8.9	61,134	104,597	4.243	2.7
TransCenter	**5d1+CH**	**72.4**	**81.2**	**57.9**	**64.2**	**12.3**	**25,121**	**115,421**	**2,290**	**8.6**	**5d1+CH**	**72.5**	**81.1**	58.1	64.7	12.2	**25,722**	114,310	**2,332**	8.8
SORT [11]	**no**	42.7	78.5	45.1	16.7	26.2	27,521	264,694	4,470	**<27.7**										
Tracktor++ [7]	**no**	52.6	**79.9**	52.7	29.4	26.7	**6,930**	236,680	1,648	1.2										
ArTIST-T [147]	**no**	53.6		51.0	31.6	28.1	7,765	230,576	**1,531**	<1.2										
GNNMatch [130]	**no**	54.5	79.4	49.0	32.8	25.5	9,522	223,611	2,038	0.1										
TADAM [61]	**no**	56.6		51.6			39,407	18,2520	2,690											
MLT [201]											**no**	48.9	78.0	54.6	30.9	22.1	**45,660**	216,803	**2,187**	3.7
TransCenter	**no**	**67.7**	79.8	**58.9**	**65.6**	**11.3**	54,967	**108,376**	3,707	8.4	**no**	**67.7**	**79.8**	**58.7**	**66.3**	**11.1**	56,435	**107,163**	3.759	**8.4**

compared to all the other methods. Concretely, TransCenter outperforms current methods by +19.2%/+8.4% in MOTA with the public/private setting trained with **ch** and +11.1%/18.8% without extra data. From the results, we observe another remarkable achievement of TransCenter is the significant decrease of FN while keeping a relatively low FP number. This indicates that the dense representation of the detection queries can help effectively detect objects sufficiently and accurately. As for tracking, TransCenter maintains low IDS numbers in MOT20 running at around 8 fps in such crowded scenes, thanks to our careful choices of QLN and the TransCenter Decoder. Very importantly, to the best of our knowledge, our study is the first to report the results of *all settings* on MOT20, demonstrating the tracking capacity of TransCenter even in a densely crowded scenario. The outstanding results of TransCenter in MOT20 further show the effectiveness of our design.

KITII. Additionally, we show the results of TransCenter in the KITTI dataset. TransCenter outperforms significantly CenterTrack [209] in pedestrian tracking (+5.3% MOTA) while keeping a close performance in car tracking. However, the KITTI dataset is constructed in an autonomous driving scenario with only up to 15 cars and 30 pedestrians per image (some of the sequences even contain no pedestrians). The fast-moving camera motion and the sparse object locations cannot fully show the capacity of TransCenter to detect and track crowded objects.

Table 3.3: KITTI testset results in MOTA, MOTP, FP, FN, IDS and FPS. Best results are underlined.

	Method	MOTA ↑	MOTP ↑	FP ↓	FN ↓	IDS ↓	FPS ↑
Car	MASS [180]	84.6	85.4	4,145	786	353	100.0
	IMMDP [184]	82.8	82.8	5,300	422	211	5.3
	AB3D [179]	83.5	85.2	4,492	1,060	126	214.7
	SMAT [60]	83.6	85.9	5,254	175	198	10.0
	TrackMPNN [136]	87.3	84.5	2,577	1,298	481	20.0
	CenterTrack [209]	88.8	85.0	2,703	886	254	22.2
	TransCenter	87.3	83.8	3,189	847	340	18.5
Person	AB3D [179]	38.9	64.6	11,744	2,135	259	214.7
	TrackMPNN [136]	52.1	73.4	7,705	2,758	626	20.0
	CenterTrack [209]	53.8	73,7	8,061	2,201	425	22.2
	TransCenter	59.1	73.2	6,889	2,142	436	18.5

3.4.4 Efficiency-Accuracy Tradeoff Discussion

To have a direct idea to the better design of TransCenter, we discuss in detail the efficiency-accuracy tradeoff comparing TransCenter, TransCenter-Lite, and TransCenter-Dual to the transformer-based concurrent works – TransTrack [131] and TrackFormer [123] as well as the center-based MOT methods – CenterTrack [209] and FairMOT [203]. The comparisons take into account the number of model parameters, the model memory footprint during inference, the inference speed (frame per second or FPS), and the MOTA performance as shown in Tab. 3.4, completed with Tab. 3.1 and Tab. 3.2. Moreover, we compare the center heatmap/query responses of the aforementioned methods in Sec. 3.5.1, showing that TransCenter produces sufficient and accurate outputs.

Compared to Transformer-Based MOT. With the scientific purpose and respect to our concurrent works, we compare TransCenter with them both in accuracy and inference speed. Unfortunately, Trackformer [123] by far only shows results on MOT17 and TransTrack [131] does not show results in all settings. One important remark is that TransCenter systematically outperforms TransTrack

Table 3.4: Comparison among CenterTrack [209], FairMOT [203], Trackformer [123], TransTrack [131] and our proposed models in number of model parameters (#params), Inference Memory (IM), Frame Per Second (FPS), MOTA. The results are evaluated on both MOT17 and MOT20 testsets in private detection setting. All the models are pretrained on CrowdHuman [152] except FairMOT [203] trained also on **5d1+CH** datasets. The default input image size for CenterTrack [209] is 544 × 960, 608 × 1088 for FairMOT [203] and TransCenter-Lite, 640 × 1088 for TransCenter and TransCenter-Dual. TrackFormer [123] and TransTrack [131] use varying input sizes with short size of 800.

	Setting	#params (M) ↓	IM (MB) ↓	FPS ↑	MOTA ↑
MOT-17	CenterTrack [209]	20.0	892	17.5	67.8
	FairMOT [203]	20.3	892	25.9	73.7
	TrackFormer [123]	40.0	976	6.8	65.0
	TransTrack [131]	47.1	1,002	10.0	74.5
	TransCenter-Dual	39.7	954	5.6	76.0
	TransCenter	35.1	938	11.8	76.2
	TransCenter-Lite	8.1	838	17.5	73.5
MOT-20	CenterTrack [209]	20.0	892	-	-
	FairMOT [203]	20.3	892	13.2	61.8
	TrackFormer [123]	40.0	976	-	-
	TransTrack [131]	47.1	1,002	7.2	64.5
	TransCenter-Dual	39.7	954	5.1	73.5
	TransCenter	35.1	938	8.7	72.9
	TransCenter-Lite	8.1	838	11.0	68.0

and TrackFormer in both accuracy (MOTA) and speed (FPS) with a smaller model size (# of model parameters) and less inference memory consumption *in all settings*. Precisely, using the same training data, TransCenter exhibits better performance compared TransTrack by +1.7% MOTA (by +11.2% v.s. TrackFormer) in MOT17 and significantly by +8.4% in MOT20. Moreover, we recall that, unlike our concurrent works, TransCenter leverages pixel-level dense and multi-scale detection queries to predict dense center-based heatmaps, mitigating the miss-tracking problem while keeping relatively good computational efficiency with sparse tracking queries, efficient QLN and TransCenter Decoder. TransCenter demonstrates thus a significantly better efficiency-accuracy tradeoff. Finally, for different MOT applications, we provide TransCenter-Dual, introduced to further boost the performance in crowded scenes, and TransCenter-Lite for efficiency-critical applications.

Compared to Center-Based MOT. With the long-term dependencies and the dense interactions of queries in transformers, unlike pure CNN-based center MOT methods, our queries interact globally with each other and make global decisions. TransCenter exhibits much better accuracy compared to previous center-based MOT methods, CenterTrack and FairMOT [203, 209]. Precisely as shown in Tab. 3.1 and Tab. 3.4, TransCenter *tracks more and better* compared to CenterTrack, with +8.4% (+10.4%) MOTA and -71,505 (-73,812) FN in MOT17 private (public) detections, *using the same dense center representations and trained with the same data*. CenterTrack does not show results in MOT20 while FairMOT, similar to CenterTrack, showed good result in MOT20. It alleviates miss detections by training with much more data, leading to much fewer FN but producing much noisier detections (FP). Surprisingly, with much less training data (**CH**), TransCenter still outperforms by a large margin [203] (+11.1% MOTA shown in Tab. 3.4) even in very crowded MOT20, with *cleaner detections (suppressing -74,844 FP) and better tracking associations (-2,618 IDS)* shown in Tab. 3.2. Indeed, the inference speed is slower compared to CNN-based methods, but the above comparisons have demonstrated that previous center-based MOT methods are not comparable in terms of accuracy to TransCenter, with an acceptable fps around 11 fps for MOT17 and 8 fps for MOT20. Moreover, to adapt to applications with more strict inference constraints, TransCenter-Lite is introduced, keeping a better performance while having competitive inference speed compared to center-based MOT methods.

To conclude, TransCenter expresses both better accuracy and efficiency compared to transformer-based methods [123, 131]; much higher accuracy numbers and competitive efficiency compared to [203, 209], showing better efficiency-accuracy balance.

3.4.5 Ablation Study

In this section, we first experimentally demonstrate the importance of our proposed image-related dense queries with naive DETR to MOT approaches. Then, we justify the effectiveness of our choices of QLN and TransCenter Decoder (see illustrations in Fig. 3.3 and Fig. 3.4, respectively), considering the computational efficiency and accuracy. Both results are shown in Tab. 3.5. Furthermore, in

Table 3.5: Ablation Study of different model structure implementations evaluated on MOT17 and MOT20: different variants of the QLN and TransCenter-decoders are discussed in Fig. 3.3 and Fig. 3.4, respectively. The module ablated in each line is marked in grey. For the queries, "D-S" means dense detection and sparse tracking queries, with "D" refers to "dense" and "S" refers to "sparse", similarly for "S-S" and "D-D". Line 10, 12, 14, 16 are the proposed TransCenter, line 9 TransCenter-Lite and line 11 TransCenter-Dual, indicated with $*$.

	Settings				MOT17						MOT20					
Line	Encoder	Decoder	Queries	QLN	MOTA ↑	IDF1 ↑	FP ↓	FN ↓	IDS ↓	FPS ↑	MOTA ↑	IDF1 ↑	FP ↓	FN ↓	IDS ↓	FPS ↑
1	DETR	Dual	S-S[5]	QLN_E	49.4	49.6	4,909	8,202	602	2.5	42.5	26.8	19,243	153,429	4,375	1.5
2	DETR	Dual	D-D[6]	QLN_E	56.6	50.1	3,510	7,577	678	1.7	67.3	32.1	45,547	46,962	8,115	0.9
3	DETR	Dual	D-D	QLN_{DQ}	69.2	71.9	1,202	6,951	203	1.9	79.4[7]	66.9	7,697	53,987	1,680	1.2
4	DETR	Dual	D-D	QLN_{M_t}	66.8	71.2	1,074	7,757	167	1.9	78.3	64.7	5517	59,447	1,832	1.2
5	DETR	Dual	D-D	QLN_{D-}	65.8	70.3	1,122	7,987	176	1.9	78.4	64.2	5,340	59,288	1,798	1.1
6	DETR	Dual	D-D	QLN_{DQ}	69.2	71.9	1,202	6,951	203	1.9	79.4	66.9	7,697	53,987	1,680	1.2
7	DETR	Single	D-D	QLN_{DQ}	68.1	72.0	580	7,922	141	2.1	79.8	66.9	6,955	53,445	1,657	1.2
8	PVT	Single	D-D	QLN_{DQ}	72.3	71.4	1,116	6,156	249	8.1	83.6	75.1	14,358	34,782	1,348	6.1
9*	PVT-Lite	Single	D-S	QLN_{S-}	69.8	71.6	2,008	5,923	252	19.4	82.9	75.5	14,857	36,510	1,181	12.4
10*	PVT	Single-TQSA	D-S	QLN_{S-}	73.8	74.1	1,302	5,540	258	12.4	83.6	75.7	15,459	34,054	1,085	8.9
11*	PVT	Dual-TQSA	D-S	QLN_{S-}	74.6	76.5	892	5,879	128	5.6	84.6	78.0	13,415	33,202	921	4.8
12*	PVT	Single-TQSA	D-S	QLN_{S-}	73.8	74.1	1,302	5,540	258	12.4	83.6	75.7	15,459	34,054	1,085	8.9
13	PVT	Single	D-S	QLN_{S-}	69.4	72.5	1,756	6,335	197	13.3	83.0	74.7	14,584	36,441	1,321	9.8
14*	PVT	Single-TQSA	D-S	QLN_{S-}	73.8	74.1	1,302	5,540	258	12.4	83.6	75.7	15,459	34,054	1,085	8.9
15	PVT	Single	D-D	QLN_{D-}	71.1	71.7	1,274	6,274	278	9.0	82.5	75.0	12,686	40,003	1,216	6.3
16*	PVT	Single-TQSA	D-S	QLN_{S-}	73.8	74.1	1,302	5,540	258	12.4	83.6	75.7	15,459	34,054	1,085	8.9

Tab. 3.6, we ablate the impacts of removing the external Re-ID network and the NMS (Non-Maximum Suppression) during inference. Finally, we show an additional ablation of the number of decoder layers. For the ablation, we divide the training sets into a train-validation split, we take the first 50% of frames as training data and test on the last 25%. The rest 25% of frames in the middle of the sequences are thrown to prevent over-fitting. All the models are pre-trained on CrowdHuman [152] and tested under the private detection setting.

Dense Representations Are Beneficial. We implemented a naive DETR MOT tracker with its original 100 sparse queries (from learnable embeddings initialized from noise) with the DETR-Encoder and Dual TransCenter Decoder (Line 1 in Tab. 3.5). To compare, the same tracker but having dense representations (43,520,

[5] 100 noise-initialized learnable queries.

[6] 43,520 noise-initialized learnable queries.

[7] As other MOT methods, we observe that clipping the box size within the image size for tracking results in MOT20 improves slightly the MOTA performance. To have a fair comparison, all the results in MOT20 are updated with this technique.

i.e. $H/4 \times W/4$) is shown in Line 2. From the results shown in Line 1-2 of Tab. 3.5, we see that limited number of queries (100, by default in [214]) is problematic because it is insufficient for detecting and tracking objects, especially in very crowded scenes MOT20 (+106,467 FN, -24.8% MOTA, compared to Line 2). This indicates that having dense representations is beneficial, especially for handling crowded scenes in MOT (some visualization examples are shown in Sec. 3.5.4).

Naive Dense v.s. Image-Related Dense Representations. One naive way to alleviate the insufficient queries is to greatly increase the number of queries like in Line 2 of Tab. 3.5: we drastically increase the number of queries from 100 to 43,520 (i.e. $H/4 \times W/4$), same as the dense output of TransCenter. Meanwhile, we compare this naive dense queries implementation to a similar one in Line 3 but with image-related dense queries like in TransCenter. Concretely, we obtain the dense queries from QLN_{DQ} (as described in Fig. 3.3(c)) with the memories output from DETR-Encoder. We note that the image-related implementation has indeed 14,450 queries (i.e. the sum of the 1/8, 1/16, 1/32, and 1/64 of the image size), and the up-scale and merge operation (see Fig.3.5) forms a dense output having the same number of pixels of 43,520. Therefore, we ensure that the supervisions for the losses are equivalent[8] for both Line 2 and 3.

The main difference between Line 2 and 3 is the queries: the proposed multi-scale dense detection queries are related to the input image where one query represents one pixel. The benefit of image-related pixel-level queries are well-discussed in Sec. 3.3. From the experimental aspect in Tab. 3.5, for the noise-initialized queries in Line 2, despite the manual one-to-one matching during training, increasing the queries in a naive and drastical way tends to predict noisier detections causing much higher FP (compared to Line 3, +2,308 and +37,850 in MOT17 and MOT20, respectively) and thus much under-performed tracking results (-12.6% and -12.1% MOTA for MOT17 and MOT20, respectively). Surely, although more sophisticated implementations using sparse noise-initialized queries and Hungarian matching loss like in [131] and [123] exhibit improved results, TransCenter shows both better accuracy and efficiency, as discussed in Sec. 3.4.4.

[8]The Gaussian supervision in TransCenter for negative examples has values very close to 0, thus similar to the classification loss in Line 2.

QLN Inputs. A QLN generates tracking queries **TQ** and memories **TM**; detection queries **DQ** and memories **DM**. Based on the nature of tracking, we argue that **TQ** and **TM** should be obtained from information at different time steps since tracking means associating object positions in the adjacent frames. This creates two variants of QLN, namely QLN$_{M_t}$ with **TQ** from $\mathbf{M_t}$ and **TM** are from $\mathbf{M_{t-1}}$ (Line 4) and inversely, QLN$_{D-}$, where **TQ** are from $\mathbf{M_{t-1}}$ and **TM** are from $\mathbf{M_t}$ (Line 5). From their results, we do not observe a significant difference in terms of performance, indicating both implementations are feasible. Further, we experimentally compare QLN$_{M_t}$ (Line 4) and QLN$_{DQ}$ (Line 6), where the only difference is the number of FFN for outputting **TQ**. With an extra FFN for **TQ**, the performance is slightly improved (+2.4% for MOT17 and +1.1% for MOT20).

Efficient TransCenter Decoder. As we discuss in Sec. 3.3.2, the design of TransCenter Decoder can have important impact on the computational efficiency and accuracy with different variants (Line 11-16). Precisely, comparing Line 11 and 12, we observe indeed that, with dual-decoder handling cross-attention for both detection and tracking, the performance is superior (+0.8% MOTA for MOT17 and +1.0% for MOT20) but the inference is slowed down by around 50%. Balancing the efficiency and accuracy, we argue that TransCenter (Line 12) is a better choice. Moreover, removing the TQSA module (Line 13, 14), we obtain slight inference speed up (+0.9 fps for MOT17 and MOT20) but at the cost of accuracy (-4.4% MOTA in MOT17 and -0.6% in MOT20). Finally, we also study the effect of sparse and dense tracking (Line 15-16), surprisingly, we find that using sparse tracking can help better associate object positions between frames (-20 IDS in MOT17 and -131 IDS in MOT20) and in a more efficient way (+3.4 fps in MOT17 and +2.6 in MOT20).

Efficient PVT-Encoder. Passing from DETR-Encoder (Line 7) to PVT-Encoder (Line 8) helps get rid of the ResNet-50 feature extractor, which speeds up the inference from 2.1 fps to 8.1 in MOT17, and 1.2 to 6.1 fps in MOT20. Moreover, the PVT-Encoder exhibits better results which may be due to the lighter structure that eases the training (+4.2% MOTA in MOT17 and +3.8% in MOT20). Similarly, with PVT-Lite-Encoder, we can speed up +7 fps for MOT17 and +3.5 fps for MOT20, comparing Line 9 and 10 while keeping a competitive performance.

Table 3.6: Ablation study of external inference overheads: the results are expressed in MOTA, MOTP, FP, FN, IDS as well as FPS. They are evaluated on both MOT17 and MOT20 validation sets using TransCenter, TransCenter-Lite and TransCenter-Dual, respectively.

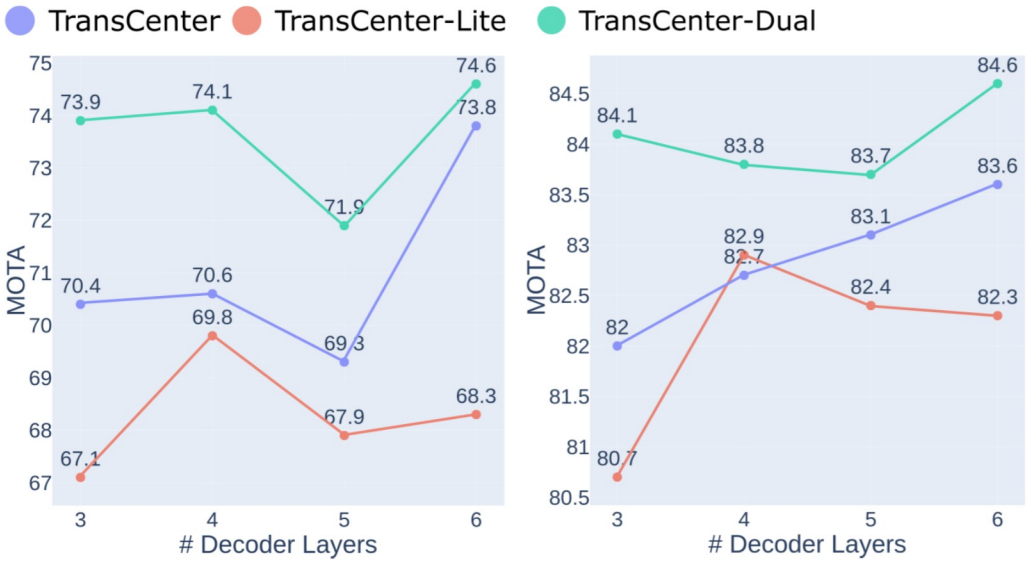
	MOT17						MOT20					
Setting	MOTA ↑	IDF1 ↑	FP ↓	FN ↓	IDS ↓	FPS ↑	MOTA ↑	IDF1 ↑	FP ↓	FN ↓	IDS ↓	FPS ↑
+ ex-ReID	73.7	73.5	1,047	5,800	286	4.5	83.6	75.5	15,468	34,057	1,064	2.9
+ NMS	73.8	74.1	1,300	5,548	261	10.8	83.6	75.8	15,458	34,053	1,084	5.1
TransCenter	73.8	74.1	1,302	5,540	258	12.4	83.6	75.7	15,459	34,054	1,085	8.9
+ ex-ReID	69.8	72.0	2,008	5,922	248	5.2	82.9	74.9	14,894	36,498	1,147	3.1
+ NMS	69.8	71.6	2,009	5,923	253	15.9	82.9	75.4	14,872	36,507	1,181	6.2
TransCenter-Lite	69.8	71.6	2,008	5,923	252	19.4	82.9	75.5	14,857	36,510	1,181	12.4
+ ex-ReID	74.6	75.8	840	5,882	162	3.0	84.5	77.7	13,478	33,209	913	2.2
+ NMS	74.6	76.5	891	5,879	127	5.1	84.6	77.9	13,413	33,203	921	3.4
TransCenter-Dual	74.6	76.5	892	5,879	128	5.6	84.6	78.0	13,415	33,202	921	4.8

External Inference Overheads. MOT methods like [7] use an external Re-ID network to extract identity features so that we can recover the objects which are temporally suspended by the tracker through appearance similarities. The Re-ID network (paired with a light-weight optical flow estimation network LiteFlowNet [77] pre-trained on KITTI [58]) is often a ResNet-50 [7], pre-trained on object identities in MOT17 trainset. From Tab. 3.6, we observe that this external Re-ID does help reduce IDS, especially in crowded scenes but it slows down significantly the inference. To speed up, we replace the external Re-ID features extractor simply by sampled features from memories $\mathbf{M}_t$, with a feature sampler like in QLN_{S-} (Fig. 3.3(a)) using positions from detections or tracks at t, which is almost costless in terms of calculation and achieves comparable results to external Re-ID.

One of the benefits of using dense representations is the discard of the one-to-one assignment process during training. Intuitively, no NMS is needed during inference. However, recall from Eq. 3.2 that the heatmaps are Gaussian distributions centered at the object centers, a 3x3 max-pooling operation is somehow needed to select the maximum response from each distribution (i.e. object center). In practice, a NMS only is performed as in [7] within tracked objects. Similarly, the NMS operation between tracks has little impact on the accuracy but with important overheads, as shown in Tab. 3.6. For this reason, NMS is also discarded from TransCenter, TransCenter-Lite, and TransCenter-Dual.

In summary, TransCenter can efficiently perform MOT with sparse tracking queries and dense detection queries operating on the proposed QLN and TransCenter

Decoder structures, leveraging the PVT-Encoder. The accuracy is further enhanced with the finer multi-scale features from the PVT-Encoder, the sparsity of the tracking queries as well as the chosen designs of the QLN and TransCenter Decoder. Therefore, TransCenter shows a better efficiency-accuracy performance compared to naive approaches and existing works.



(a) MOT17 (b) MOT20

Figure 3.6: Ablation results in MOTA using different numbers of TransCenter Decoder layers. The results are evaluated on both MOT17 and MOT20 validation sets with TransCenter, TransCenterlite and TransCenter-Dual.

Number of Decoder Layers. We ablate in Fig. 3.6 the number of decoder layers in TransCenter, TransCenter-Lite, and TransCenter-Dual. Precisely, we search the best number of decoder layers within the range of $[3, 6]$ For TransCenter, TransCenter-Dual, we find that having six layers of decoder gets the best results in terms of MOTA both in MOT17 and MOT20 while with four layers in TransCenter-Lite.

3.5 Qualitative Results and Visualizations

In this section, we qualitatively visualize the center heatmap responses (Sec. 3.5.1), the detection queries (Sec. 3.5.2) and the tracking queries (Sec. 3.5.3) of TransCen-

ter. Some tracking results (Sec. 3.5.4) in very crowded scenes of MOT20 are also provided.

3.5.1 Center Heatmap Response

We qualitatively visualize and compare our center heatmap response in Fig. 3.7(d) with other two center heatmap-based MOT methods [203, 209] in Fig. 3.7(c) and Fig. 3.7(b) as well as the box centers predicted from sparse queries from one concurrent transformer-based MOT method [131] in Fig. 3.7(a). Two *concurrent*

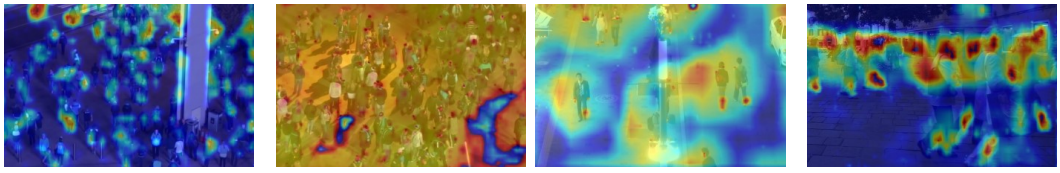
(a) TransTrack [131] (b) CenterTrack [209] (c) FairMOT [203] (d) TransCenter

Figure 3.7: Detection outputs of state-of-the-art MOT methods: (a) shows the bounding-box centers from the queries in TransTrack [131]; (b), (c) are center heatmaps of CenterTrack [209], FairMOT [203], and (d) is from TransCenter.

transformer-based MOT methods [123, 131] both use sparse queries, leading to miss detections (pink arrow), that are heavily overlapped, possibly leading to false detections (green arrow). Previous center-based MOT methods [203, 209] suffer from the same problems because the centers are estimated locally. TransCenter is designed to mitigate these two adverse effects by using dense (pixel-level) image-related detection queries producing a dense representation to enable heatmap-based inference and exploiting the attention mechanisms to introduce co-dependency among center predictions.

3.5.2 Detection-Query Visualization

We visualize in Fig. 3.8 the detection queries from TransCenter using the gradient-weighted class activation mapping, as described in [150]. The intuition is that we can consider the center heatmap prediction as a 2D classification task, similar to image segmentation. The center heatmap response is class-specific where



(a) (b) (c) (d)

Figure 3.8: Visualization of the detection queries **TQ** for TransCenter. The visualization is obtained using the gradient-weighted class activation mapping [150], as detailed in Sec. 3.5.2. (a) and (b) are from MOT20 while (c) and (d) are from MOT17. Zones with red/orange color represent higher response values and with blue color for lower values.

the class is binary with "person" or "background". Precisely, we calculate the gradient from the center heatmap w.r.t. the detection queries. The gradient is averaged over the image height and width. It is served as weights for the dense detection queries (**DQ**), producing weighted activations after multiplication. Since our dense detection queries are multi-scale, we re-scale the activations to the input image size and take the average of pixel values from all scales. The activation is plotted above the input image to show the link between their activated values and the object positions. For the visualization purpose, negative values are removed (by a ReLU activation function) and a normalization process by the maximum activation value is performed. From the visualizations, we can see that the detection queries tend to focus on areas where pedestrians are gathered. This is intuitive since the dense detection queries are used to find objects in a given scene.

(a) (b) (c) (d)

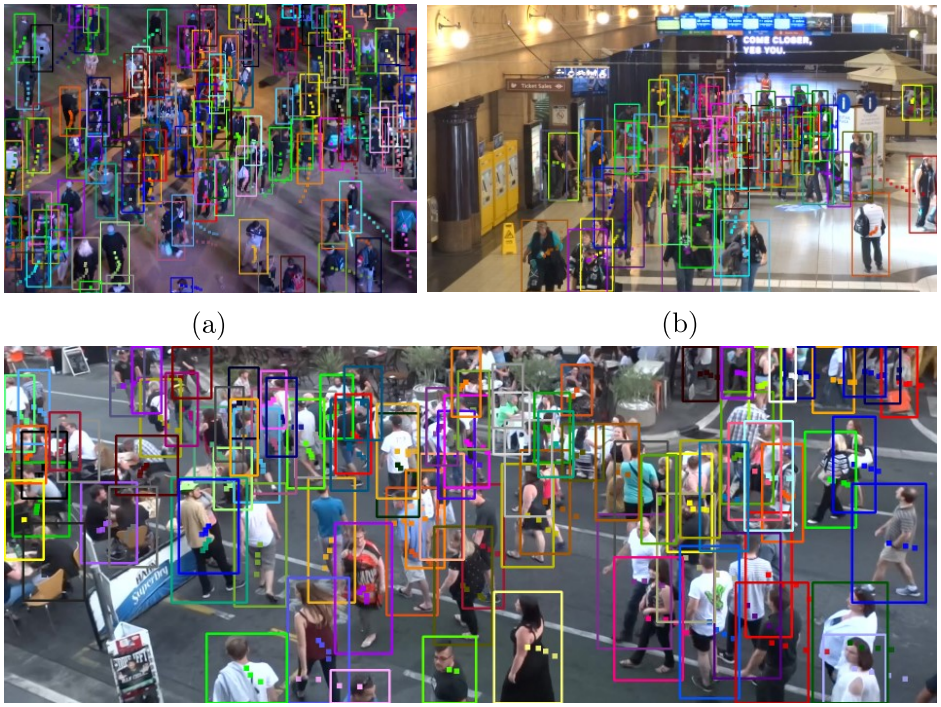
Figure 3.9: Visualization of tracking queries for TransCenter. We visualize the reference points in red dots, and their displacements (circles) and importance (circle radius) with offsets and weights calculated from the tracking queries **TQ**, as detailed in Sec. 3.5.3. (a) and (b) are from MOT20 while (c) and (d) are from MOT17.

3.5.3 Tracking-Query Visualization

Differently, the tracking queries **TQ** in TransCenter are sparse and thus cannot be plotted directly to an image like the detection queries. Therefore, we visualize them differently: Recall that **TQ** and **TM** interact in TDCA (see Sec. 3.2 in the main paper) for the tracking inside TransCenter Decoder, where we sample object features from the input **TM**. The sampling locations are obtained by displacing the input reference points (normalized object center positions at the previous frame) with sampling offsets (several offsets per reference point) learned from **TQ**. Moreover, the sampled features are weighted by the attention weights learned from **TQ**. Based on this mechanism, we visualize the tracking queries by looking at the sampling locations (displaced reference points) in the image t. Concretely, we show in Fig. 3.9 the reference points (track positions at $t-K$, K is set to 5 for better visualization) as red dots. The sampling locations are shown in circles with different colors referring to different object identities. Their radius is proportional to their attention weight (i.e. a bigger circle means a higher attention weight) and we filter out sampling locations with weights lower than a threshold (e.g. 0.2) for better visualizations. From Fig. 3.9, interestingly, we can see that the sampling locations are the surrounding of the corresponding objects and the weights and the attention weights get smaller when the locations get further from the object. This indicates qualitatively that the tracking queries from the previous time step are searching objects at t in their neighborhood.

3.5.4 Qualitative Visualizations in Crowded Scenes

We report in Fig. 3.10 some qualitative results on the crowded MOT20 sequences, to demonstrate the detection and tracking abilities of TransCenter in the context of crowded scenes. We show the predicted center trajectories and the corresponding object sizes. Fig. 3.10(a) is extracted from MOT20-04, Fig. 3.10(b) from MOT20-07 and Fig. 3.10(c) from MOT20-06. We observe that TransCenter manages to keep high recall, even in the context of drastic mutual occlusions, and reliably associate detections across time. To summarize, TransCenter exhibits outstanding results on both MOT17 and MOT20 datasets for both public and private detections, and for both with or without extra training data, which indicates that multiple-object



(a) (b)

(c)

Figure 3.10: Tracking trajectories visualization of very crowded scenes in MOT20 sequences under the private detection setting.

center point tracking using transformers equipped with dense image-related queries is a promising research direction.

3.6 Conclusion

In this work, we introduce TransCenter, a powerful and efficient transformer-based architecture with dense image-related representations for multiple-object tracking. TransCenter proposes the use of dense pixel-level multi-scale detection queries and sparse tracking queries. They are produced with carefully-designed QLN and interacting in TransCenter Decoder with image features. TransCenter is able to output dense representations for the objects' center, size, and sparse temporal displacement. Under the same training conditions, TransCenter outperforms its concurrent works in MOT17 and by a large margin in very crowded scenes like

MOT20, and even exhibits better performance than some methods trained with much more data. More importantly, TransCenter maintains a reasonably high efficiency while exhibiting a state-of-the-art performance, thanks to its careful designs proven by a thorough ablation.

3.7 Acknowledgments

Xavier Alameda-Pineda acknowledge funding from the ANR ML3RI project (ANR-19-CE33-0008-01) and the H2020 SPRING project (under GA #871245).

Chapter 4

DAUMOT: Training MOT with Unsupervised Domain Adaptation

(a) Miss detection (b) Over detection (c) Identity change

Figure 4.1: Sample qualitative results from different sequences of MOT20 comparing direct transfer (i.e. no adaptation) to DAUMOT. (a) shows the persons missed by direct transfer whereas (b) showcases the over detection, and (c) show person identity changes. The images illustrate the existence of large domain shifts between source (MOT17) and target (MOT20) when the model is not trained with DAUMOT. Green arrows point to the errors.

4.1 Introduction

In the recent past, there has been significant scientific progress on multiple object tracking (MOT) with various strategies being developed, most of them based on deep convolutional neural networks [7, 189, 203, 209]. To this end, the community has benefited from the existence of publicly available datasets such as MOT17 [124] and MOT20 [38], which contain thousands of annotated tracks.

While these available annotations facilitate researchers to validate their proposed methods, they are generally insufficient for real-world deployment. The main reason for this is that a tracking method trained on a *source* dataset and evaluated on a (different) *target* dataset tends to perform worse than the same method trained directly on the target dataset. Thus, it is usually necessary to collect and annotate a tracking dataset specific to each deployment scenario, a task that is often too costly and unrealistic.

As extensively shown in our experiments, the unavailability of annotations hinders supervised techniques to obtain competitive performance, motivating the need for unsupervised MOT training methods. We argue that this is due to a *distribution shift* between the source and target datasets, and we investigate *unsupervised domain adaptation (DA)* for MOT. As an example, Fig. 4.1 demonstrates the interest of the proposed DAUMOT training framework on some target sequences.

Recent MOT models rely on jointly solving the detection and tracking tasks [7, 189, 203, 209], and exploit person re-identification (re-ID) models to limit track fragmentation [188, 209] or to associate detections through time [7, 203]. Although to the best of our knowledge no unsupervised DA method has been reported in the MOT literature, there exists some art exploring DA for person detection and re-ID.

Previous works in the field of domain adaptive object detection [23, 28, 126, 146, 187, 213] have highlighted the drop in detection performance in the case of a significant domain shift. As an example, [28] addressed the adversarial DA problem for object detection. Such strategies have also been explored to tackle the unsupervised person re-ID (URID) task [46, 51, 56, 113, 200]. For instance, clustering and finetuning [51, 56] first clusters the features into pseudo-ID labels that are then used to fine-tune the feature extraction network.

Although DA has proven useful for both person detection and re-ID, the respective methods operate mainly on single images and are unable to exploit temporal information. Therefore, the immediate question is how to leverage that information in DA for MOT, and whether or not this is advantageous w.r.t. simply extending existing DA techniques to MOT.

We investigate unsupervised DA in MOT by jointly addressing the identity and detection domain shift problems as well as their disentanglement.

Our contributions can be summarized as follows:

- Empirical evidence is reported on the severity of the domain shift problem in MOT by measuring the tracking performance on a target set of a model trained on a source set (direct transfer).
- We introduce DAUMOT, a novel unsupervised DA MOT training strategy to alleviate domain shift, together with two baselines, namely TA and TADA.
- Different from DA for object detection, two adversarial losses are designed for MOT. First, adversarial sequence alignment to tackle the *inter-sequence* domain shift, employing multi-class discriminators at the detection and image levels. Second, identity-detection disentanglement is proposed to limit the coupling between the detection and re-ID branches.
- Experimental results demonstrate the effectiveness of DAUMOT with two different trackers on two unsupervised DA settings and using standard MOT datasets [38, 124], namely MOT17 → MOT20 and MOT20 → MOT17.

4.2 Related Works

Domain Adaptive Object Detection and Segmentation. While object detection has been extensively investigated, see for instance ([59, 137, 139]), adapting such methods under unsupervised learning constraints remains a scientific challenge. The use of adversarial DA ([55]) for object detection was firstly investigated in ([28]). They applied it to an FRCNN ([139]) detector and performed distribution

alignment at both image and detection levels. Their work motivated the appearance of a family of DA Faster-RCNN ([23, 126, 146, 187, 213]). Differentiated alignment at local and global scale ([146]) exploits global scene information to guide adaptation, Hierarchical Calibration Networks balance feature transferability and discriminability ([23]), additional image-level categorical classifiers improves categorical consistency ([187]) when performing alignment, while clustering methods at the RPN level ([213]) adjust region-level alignment. Other works take advantage of Cycle-GAN's ([212]) ability to transfer image style to create and finetune the detector on a synthetic dataset ([74, 78]), or cast the task as a robust learning problem ([87]). Moreover, [215] uses the unsupervised domain adaptation for semantic segmentation.

Unsupervised person re-ID. This approach takes advantage of recent achievements in supervised person re-ID models without requiring the costly labeling process on the target dataset. Most studies start from a pre-trained model on the source dataset and are based on the clustering and finetuning framework [46, 51, 56, 113, 200]. They alternate between a clustering step that generates noisy pseudo-labels and a finetuning step that adapts the network to the target dataset using the generated pseudo-labels. A lot of effort has been put into improving the quality of the pseudo-labels [46, 51, 56, 113]. Other approaches, inspired by unsupervised DA, use both source and target datasets during adaptation [21, 39, 135, 156, 207, 208]. They aim to match the source and target distributions while leveraging source ground-truth ID labels to keep the discriminative ability of features. For instance, SPGAN [39] proposes to learn a mapping from source detections to the target detections, and the mapped source detections are used to train the re-ID model. Alternatively, some exploit standard adversarial unsupervised DA to match the source and target distributions [55, 135]. However, negative transfer [37] induced by the strict ID separation between the source and target datasets limits the ability of such strategies. A different approach exploits spatial-temporal consistency to reconstruct tracks and use them to finetune the re-ID model [103, 119].

Unsupervised Object Tracking. UDT [172] proposes a simple unsupervised training strategy for single object trackers. The target object is annotated in at least one target template frame, and a siamese network searches the object in

another unlabeled frame (forward). The predicted object position is then used to inversely search the object in the labeled frame (backward). From this, a consistency loss can be calculated to train the model. USOT [204] builds upon this idea by generating target labels using optical flow and dynamic programming, while AMMC [75] leverages a novel data augmentation method by mimicking motion change. However, extensions of these works to MOT are not straightforward due to the variable number of objects that appear, overlap, and disappear at any single video frame.

tracking by animation [69] has been proposed to tackle the unsupervised MOT problem. The method is evaluated under very simple experimental settings using synthetic data (MNIST-MOT –floating hand-writing digits [98]– and Sprites-MOT –synthetic floating sprites–, both with uniform background). Only a few qualitative results in a simple multi-person tracking scenario with few pedestrians [142] are shown. This work has been extended [76] to noisy background conditions by adding more complex synthetic patterns. However, none of these methods are evaluated on standard MOT datasets [38, 124] and the training remains on synthetic floating objects with a simple background. SimpleReID [83] uses tracks produced by an existing tracker that has been trained on standard MOT datasets [38, 124] in a supervised manner. A re-ID network trained with contrastive losses temporally associates incomplete tracks belonging to the same person to a trajectory. Since SimpleReID uses target annotations, it cannot be considered unsupervised.

While there have been some attempts to address unsupervised MOT, they are not designed for realistic scenarios. We get inspiration from recent research in unsupervised DA for person detection and person re-ID to propose DAUMOT, an unsupervised MOT framework whose goal is to overcome the practical limitations of previous attempts. We evaluated the impact of DAUMOT on two recent MOT methods, namely FairMOT [203] and TransCenter [188], and on two unsupervised tracking settings using standard MOT datasets [38, 124].

4.3 Unsupervised DA MOT Training

Direct application of existing adaptation strategies for detection and person re-ID to the tracking problem does not yield optimal results, as shown in our experimental section (Sec. 4.4). We believe that this is due to three factors: (i) Methods conceived for DA in object detection do not exploit the sequential nature of the tracking problem. (ii) There can be domain shifts between tracking sequences from the same dataset (domain). A similar problem can be found in real-world applications, where a mobile platform might have to face different visual conditions once deployed. (iii) The detection and re-ID branches in tracking methods are usually jointly trained from shared features. However, when ground-truth labels are not available, straightforward training might add spurious correlations between the two branches (e.g. certain regions are more prone to have certain identities). Consequently, an unsupervised disentanglement mechanism between the two branches should be enforced.

We propose to address the unsupervised MOT training problem with DAUMOT, alternating between *tracking* and *adaptation* to exploit temporal information. Moreover, DAUMOT uses adversarial strategies to (i) align source and target datasets distributions, both at the detection and image levels, and (ii) to disentangle identity-detection dependencies. In the following, we first introduce some basic notations, and then present the main features of DAUMOT, namely tracking and adaptation, adversarial sequence alignment, and identity-detection disentanglement.

4.3.1 Notations

We first introduce the notations for a generic tracker (following the tracking-by-detection paradigm) and for the DA problem. In this work, we focus on online (i.e. causal) trackers for simplicity. The extension to offline/non-causal trackers is straightforward.

Let $\mathcal{S}$ and $\mathcal{T}$ be a labeled source and unlabeled target datasets respectively. They are constituted by $M^{\mathcal{S}}$ (resp. $M^{\mathcal{T}}$) video sequences with images $\boldsymbol{I}^{\mathcal{S}}_{i,t}$ (resp. $\boldsymbol{I}^{\mathcal{T}}_{i,t}$), where i and t are the sequence and time indices, respectively (for simplicity, we might omit the subscript t when it is clear from the context). Each sequence

of the source dataset $\mathcal{S}$ is annotated with the bounding boxes and ID labels of $N^{\mathcal{S}}$ tracks, denoted by $b^{\mathcal{S}}_{i,n}$, where n is the identity index of object n. The target dataset does not contain any annotated bounding boxes, and we will denote by $\tilde{b}^{\mathcal{T}}_{i,\tilde{n}}$ the bounding boxes of the $\tilde{N}^{\mathcal{T}}$ tracks inferred by the tracker.

Since one of the objectives of this work is to develop a widely-usable unsupervised training method that is agnostic to MOT methodologies available today, we consider in the following a generic tracker with a backbone ϕ followed by two branches (parameterized networks): a detection branch ψ_{DET} producing track candidates and an identity branch ψ_{ID} associating detections corresponding to the same object through time (see Fig. 4.2).

The proposed DAUMOT starts from models pre-trained on the source dataset $\mathcal{S}$ using their original training strategies. We denote the loss of the detection and identity branches as $\mathcal{L}^{\mathcal{S}}_{\text{DET}}$ and $\mathcal{L}^{\mathcal{S}}_{\text{ID}}$, and define the pre-training loss as:

$$\mathcal{L}^{\mathcal{S}}_{\text{PRE}}(\phi, \psi_{\text{DET}}, \psi_{\text{ID}}) = \mathcal{L}^{\mathcal{S}}_{\text{DET}}(\phi, \psi_{\text{DET}}) + \mathcal{L}^{\mathcal{S}}_{\text{ID}}(\phi, \psi_{\text{ID}}). \tag{4.1}$$

To illustrate with an example, in the case of FairMOT [203], ϕ is the DLA-34 variant [210], with ψ_{DET} and ψ_{ID} corresponding to two convolutional layers having the ReLU activation. $\mathcal{L}^{\mathcal{S}}_{\text{DET}}$ stands for the 2D focal loss of dense object center heatmaps whereas $\mathcal{L}^{\mathcal{S}}_{\text{ID}}$ is the categorical cross-entropy loss of object identities.

4.3.2 Tracking and Adaptation

Inspired by person re-ID's clustering and finetuning framework [46, 51, 56], we propose the following tracking and adaptation (TA) framework. Clustering and finetuning embeds the detections using the current re-ID model and clusters them to generate ID pseudo-labels, which are then used to finetune the re-ID model. This is carried out independently for each detection regardless of its spatial and temporal occurrence.

TA alternates between two different steps. The tracking step considers the appearance similarity by exploiting the identity embedding, and more importantly, explores the spatial (multiple object positions) and temporal (multiple frames) information to produce pseudo-tracks. This improves the quality of the pseudo-ID

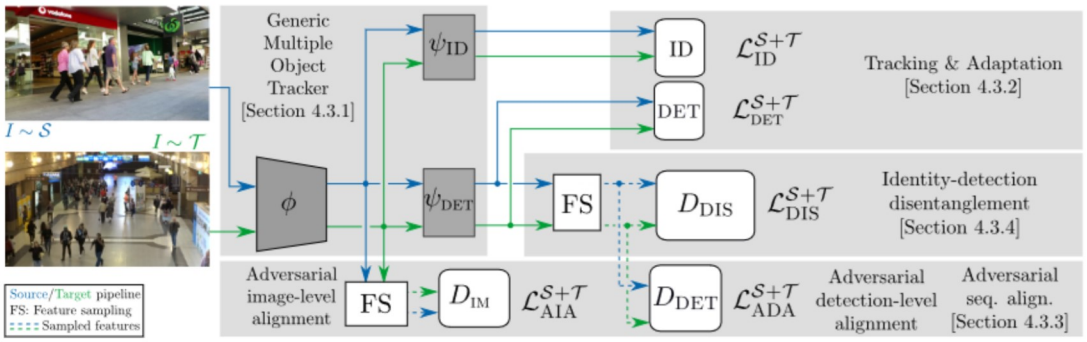


Figure 4.2: Overview of DAUMOT: alternating between *tracking* and *adaptation*. First, *tracking* uses a generic MOT method on the target sequences to obtain bounding-box and identity pseudo-labels. Second, the *adaptation* step updates the tracker using labels and pseudo-labels respectively for the source and target datasets, using standard ID and detection losses. In addition, we propose two adversarial strategies, namely *adversarial sequence alignment* and *identity-detection disentanglement*, to train the generic MOT method to be invariant to inter-sequence domain shifts and disentangle the identity ψ_{ID} and detection ψ_{DET} branches. The adversarial sequence alignment is implemented at the image and detection levels with two multi-class discriminators, D_{IM} and D_{DET}. The adversarial disentanglement is implemented with an ID multi-class discriminator D_{DIS}. All discriminators operate on features sampled from the respective feature maps.

labels because it forces them to be temporally, spatially, and visually consistent. Namely, the tracking step generates the pseudo-tracks on the whole unlabeled target dataset, $\tilde{b}_{i,\tilde{n}}^{\mathcal{T}}$, using the source pre-trained tracker and its original inference strategy.

The adaptation step exploits source ground-truth tracks $b_{i,n}^{\mathcal{S}}$ and the target pseudo-tracks $\tilde{b}_{i,\tilde{n}}^{\mathcal{T}}$ to perform domain adaptation to the identity and detection branches. Regarding the ID branch, the original ID classifier is extended to output of size $N^{\mathcal{S}} + \tilde{N}^{\mathcal{T}}$ and the identity loss is defined as the combination of the source $\mathcal{S}$ and target $\mathcal{T}$ ID losses:

$$\mathcal{L}_{\text{ID}}^{\mathcal{S}+\mathcal{T}}(\phi, \psi_{\text{ID}}) = \mathcal{L}_{\text{ID}}^{\mathcal{S}}(\phi, \psi_{\text{ID}}) + \mathcal{L}_{\text{ID}}^{\mathcal{T}}(\phi, \psi_{\text{ID}}) \tag{4.2}$$

Similarly, we finetune the detection branch with the following combination of the

source and target detection losses:

$$\mathcal{L}_{\text{DET}}^{\mathcal{S}+\mathcal{T}}(\phi, \psi_{\text{DET}}) = \mathcal{L}_{\text{DET}}^{\mathcal{S}}(\phi, \psi_{\text{DET}}) + \mathcal{L}_{\text{DET}}^{\mathcal{T}}(\phi, \psi_{\text{DET}}) \tag{4.3}$$

The ID classifier and final detection layers are implicit in the losses above, although they are displayed in Fig. 4.2.

Even though we consider TA as an extension of the clustering and finetuning framework to MOT, we note that it is in itself a novel approach to the MOT task that has not been covered in the literature so far. TA will be compared to DAUMOT to show the effectiveness of our adversarial components. Since there is no similar work for unsupervised MOT, we contribute with another baseline that extends TA with a domain adversarial binary (source/target) discriminator, similar to [28] for person detection, and name it TADA (tracking and adaptation with domain adaptation). We argue that this second baseline is not sufficient for the MOT problem for two main reasons. First, source/target discrimination does not account for inter-sequence distribution shifts that are often found in MOT. Second, since MOT methods deal with both detection and re-ID, we experimentally prove that it is beneficial to disentangle these two sub-tasks, as shown in [109] under the standard supervised paradigm. We further confirm this intuition in the *unsupervised domain adaptation* paradigm without modifying the original architecture, as explained below, thus proposing DAUMOT that can be used with a generic MOT tracker.

4.3.3 Adversarial Sequence Alignment

In MOT, both in standard datasets and in practical deployment, we can observe significant inter-sequence domain shifts. In fact, each sequence has a distinguishable and temporally consistent background, illumination conditions, and camera viewpoint. We propose to exploit these domain shifts with a sequence-based adversarial strategy that we name *adversarial sequence alignment*. The interest of an adversarial framework is to interpret the backbone ϕ and detection branch ψ_{DET} as *feature generators*: extracting image and detection features respectively. Indeed, if we adversarially train ϕ and ψ_{DET} to extract features that are invariant to domain shifts between training sequences (source and target), we will improve

their generalization performance to other sequences, as shown in [107] and our experiments.

The adversarial discriminator is a multi-class classifier where the classes are the source and target sequence indices, and thus aims to recognize the sequence. We apply this strategy at two different levels, as in [28]. First, with features sampled over the entire input image (image-level alignment), used after the backbone ϕ. Second, with features sampled at the bounding-box centers (detection-level alignment), used after the detection branch ψ_{DET}. Since ground-truth bounding boxes are not available for the target dataset, we sample detection features from the inferred bounding boxes. Sampling features from image-level feature maps allows us to reduce the computational overhead needed for adaptation.

Adversarial Image-level Alignment. It aims at endowing the shared embedding (output of the backbone ϕ) with invariance w.r.t. the inter-sequence domain shift. To do so, we randomly extract features from the feature map $\phi(\mathbf{I})$ and train a multi-class discriminator D_{IM} that classifies the sequence index of the input feature and adversarially updates ϕ. The discriminator is trained with a standard multi-class cross-entropy loss:

$$\mathcal{L}_{\text{AIA}}^{\mathcal{S}+\mathcal{T}} = -\mathbb{E}_{\boldsymbol{i}\sim\mathcal{S}\cup\mathcal{T},\boldsymbol{I}\sim\boldsymbol{i},f\sim\phi(\boldsymbol{I})}\left\{\log\left[D_{\text{IM}}(f)\right]_{\boldsymbol{i}}\right\}, \tag{4.4}$$

where the sequence $\boldsymbol{i}$ is sampled from the source and target datasets, the image $\boldsymbol{I}$ is sampled from the sequence, the feature f from the feature map $\phi(\boldsymbol{I})$, and $[D_{\text{IM}}(f)]_{\boldsymbol{i}}$ denotes the discriminator's output corresponding to the $\boldsymbol{i}$-th class.

Adversarial detection-level Alignment. It uses a similar strategy to the image-level adversarial alignment but considers the backbone together with the detection branch as the generator. It therefore samples features from the detection feature map $\psi_{\text{DET}}(\phi(\boldsymbol{I}))$. In addition, since the detection branch must focus on providing ID invariant detection features, we sample the feature maps within the object bounding boxes. While the ground-truth bounding boxes $b_{i,n}$ are available to sample the source dataset, this information is nonexistent for the target dataset. We overcome this problem by using the inferred bounding boxes $\tilde{b}_{i,\tilde{n}}$ as mentioned earlier. Therefore, we train ψ_{DET} and ϕ with a sequence multi-class discriminator

D_{DET}. The latter is trained with a standard multi-class cross-entropy loss:

$$\mathcal{L}_{\text{ADA}}^{\mathcal{S}+\mathcal{T}} = -\mathbb{E}_{\boldsymbol{i},\boldsymbol{I},f\sim\psi_{\text{DET}}(\phi(\boldsymbol{I}))}\left\{\log\left[D_{\text{DET}}(f)\right]_{\boldsymbol{i}}\right\}, \tag{4.5}$$

where the sequence $\boldsymbol{i}$ and the image $\boldsymbol{I}$ are sampled using the same strategy as in image-level adversarial alignment. The feature f is sampled from the feature map of the detection branch and within one of the ground-truth (inferred) bounding-boxes for the source (target) dataset.

Adversarial Sequence Alignment. Denoted as $\mathcal{L}_{\text{ASA}}$, it is a combination of the two aforementioned alignments by adding up $\mathcal{L}_{\text{AIA}}$ and $\mathcal{L}_{\text{ADA}}$. The adversarial game consists of training the image-level and detection-level discriminators to recognize the sequence from the features computed by the generator, while the generator aims to extract domain invariant features by maximizing $\mathcal{L}_{\text{ASA}}$.

4.3.4 Identity-detection Disentanglement

So far, we tackled the domain adaptation for the detection and identity branches separately by performing adversarial sequence alignment using losses Eq. 4.4 and Eq. 4.5, and adapting ψ_{ID} and ψ_{DET} with the losses Eq. 4.2 and Eq. 4.3, respectively. We further consider here the influence of ID information on the detection branch when performing distribution alignment. Since the backbone is shared to perform both tasks, it is reasonable to assume that the detection and ID information are not properly disentangled in the detection branch. This problem is not evident in the standard supervised paradigm thanks to the strong detection/ID supervision provided by ground-truth labels. In addition, because ID information is domain-specific, improving identity-detection disentanglement should improve the generalization ability of the detection branch to unseen identities.

We propose to disentangle ID information associated with bounding boxes in an adversarial setting, using a similar strategy than for the detection-level adversarial alignment. Features are sampled using the bounding box positions and an ID discriminator is trained to recognize the ID associated with the bounding box (i.e. the number of classes of the discriminator is equal to the number of identities in $\mathcal{S}$ and $\mathcal{T}$). While in the source dataset all this information is available, we use

inferred identities and bounding boxes for the target. In short, the disentanglement discriminator D_{DIS} aims to recognize the ID from $\psi_{\text{DET}}(\phi(\boldsymbol{I}))$, while the generator aims to provide ID-invariant features. The discriminator is trained with a multi-class cross-entropy loss:

$$\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}} = -\mathbb{E}_{\boldsymbol{I},n,f\sim\psi_{\text{DET}}(\phi(\boldsymbol{I}))}\left\{\log\left[D_{\text{DIS}}(f)\right]_{\boldsymbol{n}}\right\}, \tag{4.6}$$

where the image $\boldsymbol{I}$ is sampled from the source and target datasets, the identity (or pseudo-identity) n is sampled from the ones present in image $\boldsymbol{I}$, and the features f are sampled from the bounding box in image $\boldsymbol{I}$ corresponding to ID n. As in the previous case, the generator (ϕ, ψ_{DET}) is trained to maximize $\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$. To sum up, as illustrated in Fig. 4.2, DAUMOT leverages a TA strategy. It also removes ID dependency from detections and closes the domain gap between the labeled source domain sequences and unlabeled target domain sequences using adversarial training with the following min-max optimization process:

$$\begin{aligned}&\min_{\phi,\psi_{\text{ID}},\psi_{\text{DET}}}\max_{D_{\text{IM}},D_{\text{DET}},D_{\text{DIS}}}\mathcal{L}_{\text{ID}}^{\mathcal{S}+\mathcal{T}}(\phi,\psi_{\text{ID}})+\mathcal{L}_{\text{DET}}^{\mathcal{S}+\mathcal{T}}(\phi,\psi_{\text{DET}})\\&-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}(\phi,\psi_{\text{DET}},D_{\text{IM}},D_{\text{DET}})-\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}(\phi,\psi_{\text{DET}},D_{\text{DIS}}),\end{aligned} \tag{4.7}$$

where the various losses are weighted empirically.

4.4 Implementation Details and Results

We first detail in this section our own baselines (since there is no similar unsupervised MOT in the literature to our best knowledge), and the used MOT methods to which we implement the proposed DAUMOT (Sec. 4.4.1). Then, we explain the experimental protocol with three settings (two for state-of-the-art comparison in the testset and one for the ablation) in Sec. 4.4.2. Finally, the testset and ablation results are discussed in Sec. 4.4.3.

4.4.1 Implementation and Baselines

To begin with, we describe the various architectures that are ablated, the implementation details of the DAUMOT architecture, and finally the state-of-the-art MOT trackers that are used in our experiments.

Baselines and Ablation. We compare DAUMOT with three different baselines: direct transfer (DT) measures the performance without any adaptation (lower bound baseline), standard supervision training the tracker with target labels (upper bound baseline) and TADA (see Section 4.3.2). We also ablate the two adversarial losses of DAUMOT as well as the impact of the pseudo-track generation process on the MOT performance.

DAUMOT Implementation. The two discriminators used for adversarial sequence alignment are classifiers with as many classes as sequences in the source and target sets ($M^{\mathcal{S}} + M^{\mathcal{T}}$). They consist of three fully-connected layers with ReLU activation and normalization layers. The discriminator used for identity-detection disentanglement follows the same principle with the right number of classes. A gradient reversal layer [55] is used to reverse the back-propagated gradient and adversarially update the tracker.

MOT Methods. We use two state-of-the-art methods to demonstrate the interest of DAUMOT, namely FairMOT [203] and TransCenter [188].

FairMOT [203] leverages an encoder-decoder backbone ϕ based on a DLA-34 variant [210] to produce a shared feature heatmap processed by a re-ID and a detection branch. The original FairMOT implementation is used to conduct experiments along with the original training strategy. The model is pre-trained on COCO [167] and then on MIX data[1] along with the considered source dataset. For DAUMOT-FairMOT, the detection-level features are extracted from the heatmap features located right before the final heatmap output of the detection branch. The image-level features are extracted right after the feature extractor ϕ. Tracking pseudo-labels are updated three times for 17→20 and once for 20→17. The weight for $\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}$ is set to 1.0 for 17→20 and 0.5 for 20→ 17, and the weight for $\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$ is set to 0.25 and 0.1, respectively. We sample $L = 100$ features to train D_{IM}.

[1] Caltech Pedestrian [41, 42], CityPersons [199], CUHK-SYS [186], PRW [206], and ETH [45]. The MOT17 sequences are removed.

TransCenter [188] uses ResNet-50 [67] and Deformable Transformers [214] as its backbone ϕ. The model is pre-trained on COCO [167], on Crowdhuman [152] and then on the considered source dataset. For DAUMOT-TransCenter, we extract image-level features from the output of the detection decoder (see details in [188]). The detection-level features are pulled before the last convolutional layer of the center-heatmap output. Since TransCenter does not have ψ_{ID}, the weight for $\mathcal{L}_{\text{ID}}^{\mathcal{S}+\mathcal{T}}$ is 0. We set 0.1 for the weight of the adversarial losses and 1.0 for the rest in (4.7). The number of image-level features is as in FairMOT ($L = 100$) and pseudo-labels are computed just once.

4.4.2 Evaluation Protocol

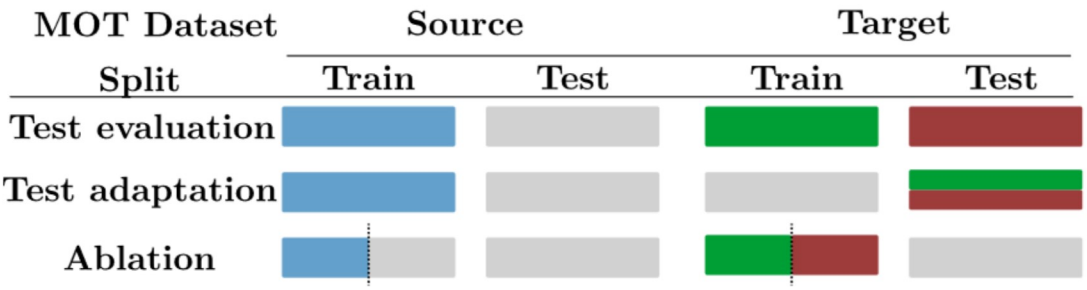


Figure 4.3: Overview of the unsupervised MOT splits. Standard MOT datasets are split into train and test where labels are available and unavailable, respectively. The colors encode which part of which subset is used for training, adaptation or evaluation.

Datasets and Metrics. We benchmark DAUMOT using two standard MOT datasets: MOT17 [124] and MOT20 [38]. We also define two adaptation settings: from MOT17 to MOT20 (17→20) and vice versa (20→17). The ablation study is run on the training set, while the final results are computed on the test set, similarly to [203, 209]. The official server is used to evaluate the performance of the MOT test set. A detailed diagram showing the training (source), adaptation (target) and evaluation (target) subsets is shown in Fig. 4.3.

To assess the MOT performance with the domain adaptation, we report tracking recall (Rcll) and precision (Prcn), ID switches (IDS) and F1 (IDF1), and finally

the standard MOT metric [9] MOTA. Since we are adapting the detection branch, we perform experiments in the private detection setting.

4.4.3 Results and Discussion

Table 4.1: MOT17 testset evaluation: we list here online methods using private detections. Best unsupervised method shown in **bold**.

	Method	Rcll ↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
MOT17 (Supervised)	CorrTracker [173]	0.824	0.940	3,369	0.736	0.765
	FUFET [151]	0.825	0.934	3,237	0.680	0.762
	MLT [201]	0.805	0.942	1,719	0.755	0.753
	CSTrack [109]	0.797	0.950	3,567	0.726	0.749
	PermaTrack [166]	0.796	0.939	3,699	0.689	0.738
	FairMOT [203]	0.792	0.942	3,303	0.723	0.737
	GSDT_V2 [176]	0.786	0.944	3,891	0.665	0.732
	TransCenter [188] [2]	0.781	0.950	4,614	0.622	0.732
	SOTMOT [114]	0.789	0.918	5,184	0.719	0.710
	TraDeS [81]	0.734	0.952	3,555	0.639	0.691
	QDTrack [129]	0.740	0.940	3,378	0.663	0.687
	CenterTrack [209]	0.716	0.956	3,039	0.647	0.678
	ChainedTracker [132]	0.716	0.948	5,529	0.574	0.666
	GSDT [176]	0.744	0.906	3,318	0.687	0.662
	TransTrack [131]	0.710	0.943	5,355	0.569	0.658
	MTP [22]	0.577	0.974	1,188	0.604	0.559
20→17 (Unsup.)	DT-FairMOT	0.758	0.883	3,087	0.692	0.652
	TADA-FairMOT	**0.759**	0.930	2,808	0.714	0.698
	DAUMOT-FairMOT	0.752	**0.939**	**2,694**	**0.727**	**0.698**
	DT-TransCenter	0.686	0.765	5,421	0.539	0.465
	TADA-TransCenter	**0.665**	**0.909**	1,845	0.605	**0.596**
	DAUMOT-TransCenter	0.649	0.903	**1,416**	**0.626**	0.577

Testset Evaluation Results. We report in Tables 4.1 and 4.2 test set evaluation tracking results on MOT17 and MOT20 respectively, for both DAUMOT-FairMOT and DAUMOT-TransCenter. Results for the baselines DT, TADA, and standard

[2] In this work, we use the version of TransCenter available online in August 2021 before its efficient design.

Table 4.2: MOT20 testset evaluation: we list here online methods using private detections. Best unsupervised method shown in **bold**.

	Method	Rcll ↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
MOT20 (Supervised)	SOTMOT [114]	0.805	0.879	4,209	0.714	0.686
	GSDT [176]	0.738	0.923	3,131	0.675	0.671
	GSDT_V2 [176]	0.738	0.924	3,230	0.675	0.671
	CSTrack [109]	0.721	0.936	3,196	0.686	0.666
	CorrTracker [173]	0.815	0.841	5,183	0.691	0.652
	TransCenter [188]	0.717	0.890	4,653	0.504	0.619
	FairMOT [203]	0.828	0.806	5,243	0.673	0.618
	MLT [201]	0.581	0.868	2,187	0.546	0.489
20→17 (Unsup.)	DT-FairMOT	0.483	0.911	4,659	0.421	0.427
	TADA-FairMOT	0.656	0.842	**3,192**	**0.601**	0.521
	DAUMOT-FairMOT	**0.642**	**0.871**	3,216	0.585	**0.541**
17→20	DT-TransCenter	0.720	0.864	6,356	0.441	0.594
	TADA-TransCenter	**0.747**	0.869	4,567	0.494	0.625
	DAUMOT-TransCenter	0.745	**0.888**	**3,836**	**0.522**	**0.644**

supervision are also reported, along with state-of-the-art supervised methods. We first note that for both trackers and datasets, DT performance is significantly lower than their supervised counterparts FairMOT and TransCenter, demonstrating the interest in investigating MOT DA: for MOT20 (MOT17), DT loses 19.1% (8.5%) MOTA in DT-FairMOT and 2.5% (26.7%) in DT-TransCenter. DAUMOT exhibits systematic tracking improvement for both trackers FairMOT/TransCenter in both settings, suggesting that our approach can effectively mitigate the domain shift problem. For MOT20 (MOT17), DT is improved by +11.4%/+5.0% (+4.6%/+11.2%) in MOTA and +16.4%/+8.1% (+3.5%/+8.7%) in IDF1.

The first –quite unexpected– result we extract from the table is that the proposed unsupervised strategy, DAUMOT, outperforms several (6 methods in 20→17 and 3 in 17→20) supervised state-of-the-art MOT methods. Moreover, when comparing DAUMOT with its supervised counterparts, a systematic and significant reduction of the identity switches (IDS) can be observed. We believe this is due to the identity dependency removal from detections and the reduction of domain shifts. DAUMOT-TransCenter outperforms supervised TransCenter (reported by [188])

by 2.5% in MOTA for MOT20. One of the reasons [3] is that the MOT20 testset has sequences with large domain shift (MOT20-06 and MOT20-08) w.r.t. MOT20 training sequences. By reducing domain shifts between source and target sequences, DAUMOT can learn more generalizable features. On the other hand, supervised training can easily overfit to the train sequences, leading to worse performance on the testset.

Moreover, compared to TADA, DAUMOT performs better in MOT20 by 2.0%/1.9% in MOTA using FairMOT/TransCenter. For MOT17, TADA shows competitive results compared to DAUMOT-TransCenter in MOTA, but it has higher IDS (+429) and lower IDF1 (-2.1%). This shows that disentangling identity from detections can help to reduce IDS and to have more ID-consistent trajectories. The same phenomenon is observed in FairMOT. DAUMOT achieves better IDF1 (+1.3%) and produces fewer IDS (-114) than TADA.

Testset Adaptation Results. Additionally, to better understand the impact of our method, we present experiments where we directly perform testset adaptation, that is when we consider the testset dataset as the target domain directly, as shown in the diagram of Fig. 4.3. While this is not standard practice, because supervised methods need labels, and using testset labels would bias the results, DAUMOT does not require the labels of the target set. Adaptation is therefore performed using only the images (without annotations). After adaptation, we evaluate the results in the official MOT testset server. We first note that DAUMOT systematically outperforms Direct Transfer performance, as expected. Notably, +4.3% (+13.3%) in MOTA for MOT17 (MOT20) while using FairMOT as the base tracker, and +5.2% (+1.8%) in MOTA for TransCenter. We also note that adapting to the target testset yields similar or lower performance compared to adapting to the target training set. DAUMOT-FairMOT changes by -0.3% (+1.9%) for MOT17 (MOT20), while DAUMOT-TransCenter degrades by -6.0% (-3.2%) in MOTA. This suggests that domain shift between train and testset is relatively small (adapting to testset instead of trainset not improving performance), which further motivates our approach to investigate cross-dataset tracking performance instead of same-dataset evaluation. It also indicates that pseudo-track errors might drive

[3] Also, we observe that clipping the bounding boxes within the images in MOT20 can bring the performance of TransCenter to 67.2% in MOTA.

Table 4.3: MOT17 testset adaptation: we list here online methods using private detections. Best unsupervised method shown in **bold**.

	Method	Rcll ↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
MOT17 (Supervised)	CorrTracker [173]	0.824	0.940	3,369	0.736	0.765
	FUFET [151]	0.825	0.934	3,237	0.680	0.762
	MLT [201]	0.805	0.942	1,719	0.755	0.753
	CSTrack [109]	0.797	0.950	3,567	0.726	0.749
	PermaTrack [166]	0.796	0.939	3,699	0.689	0.738
	FairMOT [203]	0.792	0.942	3,303	0.723	0.737
	GSDT_V2 [176]	0.786	0.944	3,891	0.665	0.732
	TransCenter [188]	0.781	0.950	4,614	0.622	0.732
	SOTMOT [114]	0.789	0.918	5,184	0.719	0.710
	TraDeS [81]	0.734	0.952	3,555	0.639	0.691
	QDTrack [129]	0.740	0.940	3,378	0.663	0.687
	CenterTrack [209]	0.716	0.956	3,039	0.647	0.678
	ChainedTracker [132]	0.716	0.948	5,529	0.574	0.666
	GSDT [176]	0.744	0.906	3,318	0.687	0.662
	TransTrack [131]	0.710	0.943	5,355	0.569	0.658
	MTP [22]	0.577	0.974	1,188	0.604	0.559
20→17	DT-FairMOT	0.758	0.883	3,087	0.692	0.652
	DAUMOT-FairMOT	**0.772**	**0.914**	**2,184**	**0.724**	**0.695**
	DT-TransCenter	**0.686**	0.765	5,421	0.539	0.465
	DAUMOT-TransCenter	0.670	**0.819**	**2,607**	**0.579**	**0.517**

tracking performance down, and is probably the limiting factor of our approach in terms of tracking performance.

Model/Loss Ablation Study. The full ablation study results with different network components/baselines are detailed in Table 4.5. Overall, DAUMOT outperforms unsupervised baselines in MOTA in almost all cases. Compared to DT, the gain of DAUMOT confirms the impact of domain shifts between source and target sequences. In addition, DAUMOT exhibits better performance than the baseline TADA, gaining +0.9%/+4.2% (+0.2%/+2.0%) in MOTA in MOT20 (MOT17) using FairMOT/TransCenter, confirming the interest of adversarial sequence alignment and identity-detection disentanglement.

We have also evaluated the contribution of each component in DAUMOT.

Table 4.4: MOT20 testset adaptation: we list here online methods using private detections. Best unsupervised method shown in **bold**.

	Method	Rcll ↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
MOT20 (Supervised)	SOTMOT [114]	0.805	0.879	4,209	0.714	0.686
	GSDT [176]	0.738	0.923	3,131	0.675	0.671
	GSDT_V2 [176]	0.738	0.924	3,230	0.675	0.671
	CSTrack [109]	0.721	0.936	3,196	0.686	0.666
	CorrTracker [173]	0.815	0.841	5,183	0.691	0.652
	TransCenter [188]	0.717	0.890	4,653	0.504	0.619
	FairMOT [203]	0.828	0.806	5,243	0.673	0.618
	MLT [201]	0.581	0.868	2,187	0.546	0.489
17→20	DT-FairMOT	0.483	**0.911**	4,659	0.421	0.427
	DAUMOT-FairMOT	**0.652**	0.882	**2,532**	**0.617**	**0.560**
	DT-TransCenter	0.720	0.864	6,356	0.441	0.594
	DAUMOT-TransCenter	**0.736**	**0.865**	**5,022**	**0.476**	**0.612**

Comparing full DAUMOT to TA (i.e. DAUMOT $-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}} - \mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$), we observe a significant drop in performance when discarding the adversarial alignments and disentanglement in almost all cases. Except for FairMOT in 20→17, the interest of using the proposed adversarial strategies is confirmed in terms of MOTA, IDS, IDF1, Rcll, and Prcn almost systematically. Remarkably, we observe how DAUMOT bridges approximately half the way from DT to full supervision, providing concluding evidence that the proposed adversarial sequence alignment and identity-detection disentanglement are an effective tool for unsupervised MOT.

Table 4.5: Model/Loss ablation study: split MOT train sets are used. Best results are in **bold**. Supervision is reported for completeness.

Setting		Variant	Rcll↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
17→20	FairMOT	DT	0.612	**0.913**	6,952	0.513	0.544
		TADA	0.717	0.893	3,032	0.672	0.626
		TA	0.703	0.895	2,909	0.663	0.617
		DAUMOT $-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}$	0.710	0.903	3,125	0.666	0.629
		DAUMOT $-\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$	0.709	0.902	**2,848**	**0.674**	0.628
		DAUMOT	**0.739**	0.882	3,673	0.667	**0.635**
		Supervised	0.754	0.918	2,346	0.725	0.683
	TransCenter	DT	0.647	0.939	5,808	0.417	0.595
		TADA	0.701	0.943	4,645	0.485	0.651
		TA	0.703	0.942	4,455	0.489	0.653
		DAUMOT $-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}$	0.729	0.943	3,688	0.534	0.680
		DAUMOT $-\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$	0.700	**0.947**	4,030	0.503	0.654
		DAUMOT	**0.751**	0.934	**3,548**	**0.551**	**0.693**
		Supervised	0.889	0.944	2,613	0.686	0.832
20→17	FairMOT	DT	0.601	0.907	188	0.646	0.536
		TADA	0.671	0.926	256	0.697	0.613
		TA	0.672	**0.942**	**218**	**0.699**	**0.626**
		DAUMOT $-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}$	**0.672**	0.934	242	0.694	0.620
		DAUMOT $-\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$	0.661	0.927	245	0.697	0.605
		DAUMOT	0.664	0.937	231	0.695	0.615
		Supervised	0.743	0.951	324	0.721	0.699
	TransCenter	DT	0.464	0.741	258	0.438	0.297
		TADA	0.467	0.830	138	0.511	0.368
		TA	0.475	0.778	163	0.487	0.337
		DAUMOT $-\mathcal{L}_{\text{ASA}}^{\mathcal{S}+\mathcal{T}}$	0.478	0.829	**130**	0.506	0.377
		DAUMOT $-\mathcal{L}_{\text{DIS}}^{\mathcal{S}+\mathcal{T}}$	**0.486**	0.833	163	**0.514**	0.386
		DAUMOT	0.458	**0.871**	132	0.509	**0.388**
		Supervised	0.702	0.974	223	0.681	0.679

Pseudo-tracks Generation Analysis. In supplement to the DAUMOT component ablation, we further study the impact of the frequency of pseudo-tracks generation in Table 4.6. Interestingly, we note that the MOT performance does not behave similarly in both settings: in MOT17→MOT20, we observe that higher update frequency leads to higher performance. This suggests that gradually improving the pseudo-track helps the model to converge to a better solution. On the contrary, in MOT20→MOT17, it is harmful when pseudo-track generation gets more frequent. We explain this by the fact that MOT20 has a track distribution

Table 4.6: Pseudo-track update frequency ablation: we change the number of iterations of the *tracking* and *adaptation*, i.e. the number of time η the pseudo-tracks are regenerated during adaptation. The overall number of epochs remains fixed to 20 for all runs. Best results are in **bold**.

	η value	Tracker	Rcll ↑	Prcn ↑	IDS ↓	IDF1 ↑	MOTA ↑
17→20	$\eta = 1$	FairMOT	0.700	0.881	4,316	0.625	0.600
	$\eta = 2$		0.715	**0.885**	**3,602**	0.658	0.617
	$\eta = 3$		**0.739**	0.882	3,673	**0.667**	**0.635**
	$\eta = 1$	TransCenter	0.751	0.934	3,548	0.551	0.693
	$\eta = 2$		0.747	**0.936**	3,445	0.550	0.690
	$\eta = 3$		**0.781**	0.916	**3,249**	**0.585**	**0.704**
20→17	$\eta = 1$	FairMOT	**0.664**	**0.937**	231	**0.695**	**0.615**
	$\eta = 2$		0.633	0.891	**195**	0.659	0.552
	$\eta = 3$		0.620	0.891	263	0.624	0.540
	$\eta = 1$	TransCenter	0.458	**0.871**	**132**	**0.509**	**0.388**
	$\eta = 2$		**0.496**	0.774	168	0.507	0.349
	$\eta = 3$		0.489	0.735	189	0.478	0.310

significantly different compared to MOT17. Per sequence track density for MOT20 goes from 70 to 205 with an average of 170.9, while for MOT17, track density goes from 9.6 to 69.8 with an average of 31.8. Also, aligning the distribution of the detection branch encourages the tracker to produce as many detections on MOT17 as in MOT20, and thus produce FPs. This is reflected by the diminishing precision, which is detrimental to pseudo-track generation.

This is not the case in MOT17 → MOT20. It is because the matching track density on MOT20 will only lead to FNs. This does not affect significantly our instance-level alignment strategy and thus the adaptation performance. We argue that MOT17→MOT20 is a *more realistic scenario* since a real-life source dataset should have lower-density tracks (annotation being less work-intensive, and most existing MOT datasets have low-density tracks) compared to the target dataset.

4.5 Conclusion

In this work, we investigate and propose the first unsupervised MOT using domain adaptation: DAUMOT. Our method alternates between tracking and adaptation, and uses adversarial adaptation strategies to mitigate inter-sequence domain shifts and identity-detection coupling. We quantitatively demonstrate the effectiveness of our approach with two state-of-the-art trackers on two standard MOT datasets and in two different settings. An extensive ablation study assesses the impact of each adaptation module. Indeed, we observe a performance gap between classic supervised and unsupervised domain adaptation training, which is reasonable and fosters further investigations in the under-discovered unsupervised domain adaptation MOT field.

As for limitations, MOT17 and MOT20 focus on people tracking and do not handle multi-class tracking scenarios, like in KITTI [58]. Moreover, our experiments are performed with source and target datasets that have similar sizes, although their track density varies significantly. In real-world applications, it is likely that due to its unsupervised nature, the target dataset would be much larger than the source.

4.6 Acknowledgments

This work was a collaboration with Guillaume Delorme, who had the original idea of using unsupervised domain adaptation techniques for training MOT. The first version of this work was explained in his thesis. We equally contribute to this current version where TransCenter is introduced and the experiments are redesigned.

Chapter 5

Conclusion and Future Work

5.1 DeepMOT-Beyond CLEAR-MOT

In Chap. 2, we propose DeepMOT that closes the gap between the training and evaluation by approximating MOTA and MOTP. A differentiable proxy of the Hungarian algorithm is built to solve the one-to-one assignment problem in a differentiable way and it conveys the gradient from the approximated MOTA and MOTP to the deep MOT network. DeepMOT can effectively train the detection and re-identification branches in a unified framework.

Recently, the leading position in MOT metrics taken by the CLEAR-MOT [9] is challenged by some promising MOT metrics [118, 168]. The main drawback of CLEAR-MOT is that it focuses more on the detections (FN, FP) errors while the association error (IDS) has a limited impact on the MOTA metric. Precisely, in modern mot methods, they usually have FN in the order of 100K and FP of 10K, while they only have around 1K of IDS. To alleviate the imbalanced MOTA, DeepMOT weights each component of MOTA but the search for good weights remains empirical. To solve this issue, built on MOTA, HOTA [118] suggests evaluating the detection and association performance with two separate metrics: the detection accuracy–DetA and the association accuracy–AssA. the association accuracy is balanced by replacing IDS with TPAs (True Positive Associations), FNAs (False Negative Associations) and FPAs (False Positive Associations). Combining balanced DetA and AssA, as a possible direction of research, extending DeepMOT

to differentiable HOTA might help the training more balanced in object detection and association.

Moreover, object (re-)appearing and disappearing (birth and death processes) commonly happen in MOT. To model them and penalize their errors in an end-to-end (differentiable) way remains unsolved in deepMOT and could be a future research topic in MOT.

5.2 Multimodal TransCenter

In Chap. 3, we study a more sophisticated way of embedding transformers into MOT with TransCenter, which obtains significant improvements compared to state-of-the-art mot methods while keeping a good efficiency-accuracy balance.

The higher efficiency allows deploying TransCenter in real-world robotic applications. In general, a robot can not only see (through cameras) but also hear (through microphone arrays). We believe that having audio information of the objects/speakers is beneficial for tracking. It is true especially in visual occlusions where the audio information can help locate the occluded objects. For this reason, a multimodal TransCenter will be useful for real-world robotic applications.

5.3 Open-World Tracking with DAUMOT

In Chap. 4, DAUMOT is among the first to tackle the unsupervised domain adaptation MOT problem. While we observe significant performance improvements in the target domain using DAUMOT, we consider for mow the domain gap within one single class (i.e. pedestrian). To transfer (with annotations) a MOT method of one object class to another is a brand-new MOT problem formulated recently as open-world tracking in [116]. It tackles the problem of enabling machines to track objects that they have never seen and the objects could be an object of a different category (e.g. human to car). As a building brick, TAO [36] tracking dataset provides overall 800 categories which can approximately cover common objects in the open world. However, the methodology for open-world tracking remains unclear, and extending DAUMOT to any class might be one solution.

Appendix A

List of Works

A.1 List of Publications

1. Y. Xu, A. Osep, Y. Ban, R. Horaud, L. Leal-Taixé, X. Alameda-Pineda. "How To Train Your Deep Multi-Object Tracker". IEEE CVPR, 2020.
2. G. Delorme, Y. Xu, S. Lathuilière, R. Horaud, X. Alameda-Pineda."CANU-ReID: A Conditional Adversarial Network for Unsupervised person ReIDentification". IAPR ICPR, 2021.
3. H. Xuan, Y. Xu, S. Chen, Z. Wu, J. Yang, Y. Yan, X. Alameda-Pineda."Active Contrastive Set Mining for Robust Audio-Visual Instance Discrimination". IJCAI, 2022.

A.2 List of Submissions

1. Y. Xu*, Y. Ban*, G. Delorme, C. Gan, D. Rus, X. Alameda-Pineda. "TransCenter: Transformers with Dense Queries for Multiple-Object Tracking". Under revision for IEEE TPAMI, 2022.
2. G. Delorme*, Y. Xu*, L. Gomez Camara, R. Horaud, E. Ricci, X. Alameda-Pineda."DAUMOT: Domain Adaptation for Unsupervised Multiple Object Tracking". Under submission to Elsevier CVIU, 2022.

List of Figures

List of Tables

Bibliography

[1] M. Anton, R. Stefan, and S. Konrad. Continuous energy minimization for multitarget tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(1):58–72, 2014.

[2] S. Ba, X. Alameda-Pineda, A. Xompero, and R. Horaud. An on-line variational bayesian model for multi-person tracking from cluttered scenes. *Computer Vision and Image Understanding*, 153:64–76, 2016.

[3] N. L. Baisa. Online multi-object visual tracking using a gm-phd filter with deep appearance learning. In *Proceedings of the IEEE international conference on information fusion (FUSION)*, pages 1–8. IEEE, 2019.

[4] Y. Ban, X. Alameda-Pineda, L. Girin, and R. Horaud. Variational bayesian inference for audio-visual tracking of multiple speakers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019.

[5] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud. Tracking multiple persons based on a variational bayesian model. In *Proceedings of the European Conference on Computer Vision (ECCV) workshops*, pages 52–67. Springer, 2016.

[6] L. Bastian, S. Konrad, C. Nico, and V. G. Luc. Coupled object detection and tracking from static cameras and moving vehicles. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 30(10):1683–1698, 2008.

[7] P. Bergmann, T. Meinhardt, and L. Leal-Taixé. Tracking without bells

and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 941–951. IEEE, 2019.

[8] J. Bergstra, D. Yamins, and D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning (ICML)*, pages 115–123. PMLR, 2013.

[9] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008.

[10] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 850–865. Springer, 2016.

[11] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft. Simple online and realtime tracking. In *Proceedings of the IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016.

[12] E. Bochinski, T. Senst, and T. Sikora. Extending iou based multi-object tracking by visual information. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6, 2018.

[13] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2544–2550. IEEE, 2010.

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

[15] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In

2009 IEEE 12th International Conference on Computer Vision, pages 1515–1522. IEEE, 2009.

[16] W. Brendel, M. Amer, and S. Todorovic. Multi object tracking as maximum weight independent set. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1273–1280. IEEE, 2011.

[17] A. A. Butt and R. T. Collins. Multi-target tracking by lagrangian relaxation to min-cost network flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1846–1853. IEEE, 2013.

[18] S. Caelles, A. Montes, K.-K. Maninis, Y. Chen, L. Van Gool, F. Perazzi, and J. Pont-Tuset. The 2018 davis challenge on video object segmentation. *arXiv:1803.00557*, 2018.

[19] S. Caelles, J. Pont-Tuset, F. Perazzi, A. Montes, K.-K. Maninis, and L. Van Gool. The 2019 davis challenge on vos: Unsupervised multi-object segmentation. *arXiv:1905.00737*, 2019.

[20] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229. Springer, 2020.

[21] X. Chang, Y. Yang, T. Xiang, and T. M. Hospedales. Disjoint label space transfer learning with common factorised space. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3288–3295, 2019.

[22] K. Chanho, L. Fuxin, A. Mazen, and M. R. James. Discriminative appearance modeling with multi-track pooling for real-time multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9553–9562. IEEE, 2021.

[23] C. Chen, Z. Zheng, X. Ding, Y. Huang, and Q. Dou. Harmonizing transferability and discriminability for adapting object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8869–8878. IEEE, 2020.

[24] L. Chen, H. Ai, R. Chen, and Z. Zhuang. Aggregate tracklet appearance features for multi-object tracking. *IEEE Signal Processing Letters*, 26(11):1613–1617, 2019.

[25] L. Chen, H. Ai, C. Shang, Z. Zhuang, and B. Bai. Online multi-object tracking with convolutional neural networks. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 645–649. IEEE, 2017.

[26] L. Chen, H. Ai, Z. Zhuang, and C. Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2018.

[27] L. Chen, H. Ai, Z. Zhuang, and C. Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2018.

[28] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3339–3348. IEEE, 2018.

[29] H. Cho, Y.-W. Seo, B. V. Kumar, and R. R. Rajkumar. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1836–1843. IEEE, 2014.

[30] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[31] W. Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 3029–3037. IEEE, 2015.

[32] P. Chu and H. Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 6172–6181. IEEE, 2019.

[33] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu, and N. Yu. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4836–4845. IEEE, 2017.

[34] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773. IEEE, 2017.

[35] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893. IEEE, 2005.

[36] A. Dave, T. Khurana, P. Tokmakov, C. Schmid, and D. Ramanan. Tao: A large-scale benchmark for tracking any object. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 436–454. Springer, 2020.

[37] G. Delorme, Y. Xu, S. Lathuilière, R. Horaud, and X. Alameda-Pineda. Canu-reid: A conditional adversarial network for unsupervised person re-identification. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*, pages 4428–4435. IEEE, 2021.

[38] P. Dendorfer, H. Rezatofighi, A. Milan, J. Shi, D. Cremers, I. Reid, S. Roth, K. Schindler, and L. Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes, 2020.

[39] W. Deng, L. Zheng, G. Kang, Y. Yang, Q. Ye, and J. Jiao. Image-image domain adaptation with preserved self-similarity and domain-dissimilarity for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 994–1003. IEEE, 2018.

[40] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 36(8):1532–1545, 2014.

[41] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 304–311. IEEE, 2009.

[42] P. Dollar, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 34(4):743–761, 2011.

[43] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[44] A. Ellis and J. Ferryman. Pets2010 and pets2009 evaluation of results using individual ground truthed single views. In *Proceedings of the IEEE international conference on advanced video and signal based surveillance (AVSS)*, pages 135–142. IEEE, 2010.

[45] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[46] H. Fan, L. Zheng, C. Yan, and Y. Yang. Unsupervised person re-identification: Clustering and fine-tuning. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 14(4):1–18, 2018.

[47] K. Fang, Y. Xiang, X. Li, and S. Savarese. Recurrent autoregressive networks for online multi-object tracking. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475. IEEE, 2018.

[48] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[49] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(9), 2010.

[50] S. Florian, K. Dmitry, and P. James. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823. IEEE, 2015.

[51] Y. Fu, Y. Wei, G. Wang, Y. Zhou, H. Shi, and T. S. Huang. Self-similarity grouping: A simple unsupervised cross domain adaptation approach for person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6112–6121. IEEE, 2019.

[52] C. Gan, D. Huang, P. Chen, J. B. Tenenbaum, and A. Torralba. Foley music: Learning to generate music from videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 758–775. Springer, 2020.

[53] C. Gan, D. Huang, H. Zhao, J. B. Tenenbaum, and A. Torralba. Music gesture for visual sound separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10478–10487. IEEE, 2020.

[54] C. Gan, H. Zhao, P. Chen, D. Cox, and A. Torralba. Self-supervised moving vehicle tracking with stereo sound. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 7053–7062, 2019.

[55] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

[56] Y. Ge, D. Chen, and H. Li. Mutual mean-teaching: Pseudo label refinery for unsupervised domain adaptation on person re-identification. In *International Conference on Learning Representations (ICLR)*, 2019.

[57] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[58] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361. IEEE, 2012.

[59] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1440–1448. IEEE, 2015.

[60] N. F. Gonzalez, A. Ospina, and P. Calvez. Smat: Smart multiple affinity metrics for multiple object tracking. In *International Conference on Image Analysis and Recognition (ICIAR)*, pages 48–62, 2020.

[61] S. Guo, J. Wang, X. Wang, and D. Tao. Online multiple object tracking with cross-task synergy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8136–8145. IEEE, 2021.

[62] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1735–1742. IEEE, 2006.

[63] P. Hamed, R. Deva, and C. Charles. Globally-optimal greedy algorithms for tracking a variable number of objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1208. IEEE, 2011.

[64] S. Han, P. Huang, H. Wang, E. Yu, D. Liu, X. Pan, and J. Zhao. Mat: Motion-aware multi-object tracking. *arXiv preprint arXiv:2009.04794*, 2020.

[65] J. He, Z. Huang, N. Wang, and Z. Zhang. Learnable graph matching: Incorporating graph partitioning with deep feature learning for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5299–5309. IEEE, 2021.

[66] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2961–2969. IEEE, 2017.

[67] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. IEEE, 2016.

[68] S. He, H. Luo, P. Wang, F. Wang, H. Li, and W. Jiang. Transreid: Transformer-based object re-identification. *arXiv preprint arXiv:2102.04378*, 2021.

[69] Z. He, J. Li, D. Liu, H. He, and D. Barber. Tracking by animation: Unsupervised learning of multi-object attentive trackers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1318–1327. IEEE, 2019.

[70] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 749–765. Springer, 2016.

[71] R. Henschel, L. Leal-Taixe, D. Cremers, and B. Rosenhahn. Improvements to frank-wolfe optimization for multi-detector multi-object tracking. *arXiv preprint arXiv:1705.08314*, 2017.

[72] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[73] A. Hornakova, R. Henschel, B. Rosenhahn, and P. Swoboda. Lifted disjoint paths with application in multiple object tracking. In *International Conference on Machine Learning (ICML)*, pages 4364–4375. PMLR, 2020.

[74] H.-K. Hsu, C.-H. Yao, Y.-H. Tsai, W.-C. Hung, H.-Y. Tseng, M. Singh, and M.-H. Yang. Progressive domain adaptation for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) workshops*, pages 749–757. IEEE, 2020.

[75] L. Hu, S. Huang, S. Wang, W. Liu, and J. Ning. Do we really need frame-by-frame annotation datasets for object tracking? In *Proceedings of the 29th ACM International Conference on Multimedia (ACMMM)*, pages 4949–4957. ACM, 2021.

[76] C.-H. Huck Yang, M. Chhabra, Y.-C. Liu, Q. Kong, T. Yoshinaga, and T. Murakami. Robust unsupervised multi-object tracking in noisy environments. *2021 IEEE International Conference on Image Processing (ICIP)*, 2021.

[77] T.-W. Hui, X. Tang, and C. C. Loy. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8981–8989. IEEE, 2018.

[78] N. Inoue, R. Furuta, T. Yamasaki, and K. Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5001–5009. IEEE, 2018.

[79] H. Izadinia, I. Saleemi, W. Li, and M. Shah. 2 t: Multiple people multiple parts tracker. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 100–114. Springer, 2012.

[80] S. Jeany, B. Mooyeol, C. Minsu, and H. Bohyung. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5620–5629. IEEE, 2017.

[81] W. Jialian, C. Jiale, S. Liangchen, W. Yu, Y. Ming, and Y. Junsong. Track to detect and segment: An online multi-object tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12352–12361. IEEE, 2021.

[82] Y. Jiang, S. Chang, and Z. Wang. Transgan: Two transformers can make one strong gan. *arXiv preprint arXiv:2102.07074*, 2021.

[83] S. Karthik, A. Prabhu, and V. Gandhi. Simple unsupervised multi-object tracking, 2020.

[84] S. Karthik, A. Prabhu, and V. Gandhi. Simple unsupervised multi-object tracking. *arXiv preprint arXiv:2006.02609*, 2020.

[85] M. Keuper, S. Tang, B. Andres, T. Brox, and B. Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 42(1):140–153, 2018.

[86] M. Keuper, S. Tang, Y. Zhongjie, B. Andres, T. Brox, and B. Schiele. A multi-cut formulation for joint segmentation and tracking of multiple objects. *arXiv preprint arXiv:1607.06317*, 2016.

[87] M. Khodabandeh, A. Vahdat, M. Ranjbar, and W. G. Macready. A robust learning approach to domain adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 480–490. IEEE, 2019.

[88] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 4696–4704. IEEE, 2015.

[89] C. Kim, F. Li, and J. M. Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 200–215. Springer, 2018.

[90] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *arXiv preprint arXiv:1412.6980*, 2014.

[91] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin. A novel performance evaluation methodology for single-target trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38(11):2137–2155, Nov 2016.

[92] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems (NeurIPS)*, 25:1097–1105, 2012.

[93] H. W. Kuhn and B. Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

[94] C.-H. Kuo, C. Huang, and R. Nevatia. Multi-target tracking by on-line learned discriminative appearance models. In *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition (CVPR), pages 685–692. IEEE, 2010.

[95] H. Law and J. Deng. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 734–750. Springer, 2018.

[96] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 33–40. IEEE, 2016.

[97] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler. MOTChallenge 2015: Towards a benchmark for multi-target tracking. *arXiv:1504.01942 [cs]*, Apr. 2015. arXiv: 1504.01942.

[98] Y. LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[99] B. Lee, E. Erdenee, S. Jin, M. Y. Nam, Y. G. Jung, and P. K. Rhee. Multi-class multi-object tracking using changing point detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 68–83. Springer, 2016.

[100] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4282–4291. IEEE, 2019.

[101] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8971–8980. IEEE, 2018.

[102] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8971–8980, 2018.

[103] M. Li, X. Zhu, and S. Gong. Unsupervised person re-identification by deep learning tracklet association. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 737–753. Springer, 2018.

[104] W. Li, J. Mu, and G. Liu. Multiple object tracking with motion and appearance cues. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 0–0, 2019.

[105] W. Li, R. Zhao, T. Xiao, and X. Wang. Deepreid: Deep filter pairing neural network for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 152–159. IEEE, 2014.

[106] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2953–2960. IEEE, 2009.

[107] Y. Li, X. Tian, M. Gong, Y. Liu, T. Liu, K. Zhang, and D. Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 624–639. Springer, 2018.

[108] Z. Li, Y. Li, and N. Ramakant. Global data association for multi-object tracking using network flows. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008.

[109] C. Liang, Z. Zhang, Y. Lu, X. Zhou, B. Li, X. Ye, and J. Zou. Rethinking the competition between detection and reid in multi-object tracking. *arXiv preprint arXiv:2010.12138*, 2020.

[110] M. Lin, C. Li, X. Bu, M. Sun, C. Lin, J. Yan, W. Ouyang, and Z. Deng. Detr for crowd pedestrian detection. *arXiv preprint arXiv:2012.06785*, 2020.

[111] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 2980–2988, 2017.

[112] X. Lin, L. Girin, and X. Alameda-Pineda. Unsupervised multiple-object tracking with a dynamical variational autoencoder. In *arXiv:2202.09315*, 2022.

[113] Y. Lin, X. Dong, L. Zheng, Y. Yan, and Y. Yang. A bottom-up clustering approach to unsupervised person re-identification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, pages 8738–8745, 2019.

[114] Z. Linyu, T. Ming, C. Yingying, Z. Guibo, W. Jinqiao, and L. Hanqing. Improving multiple object tracking with single object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2453–2462. IEEE, 2021.

[115] Q. Liu, Q. Chu, B. Liu, and N. Yu. Gsm: Graph similarity model for multi-object tracking. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 530–536, 2020.

[116] Y. Liu, I. E. Zulfikar, J. Luiten, A. Dave, A. Ošep, D. Ramanan, B. Leibe, and L. Leal-Taixé. Opening up open-world tracking. *arXiv preprint arXiv:2104.11221*, 2021.

[117] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[118] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe. Hota: A higher order metric for evaluating multi-object tracking. *International journal of computer vision*, 129(2):548–578, 2021.

[119] J. Lv, W. Chen, Q. Li, and C. Yang. Unsupervised cross-dataset person re-identification by transfer learning of spatial-temporal patterns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7948–7956. IEEE, 2018.

[120] C. Ma, C. Yang, F. Yang, Y. Zhuang, Z. Zhang, H. Jia, and X. Xie. Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for

multiple object tracking. In *Proceedings of IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2018.

[121] L. Ma, S. Tang, M. J. Black, and L. Van Gool. Customized multi-person tracker. In *Asian conference on computer vision (ACCV)*, pages 612–628. Springer, 2018.

[122] T. Meinhardt, A. Kirillov, L. Leal-Taixé, and C. Feichtenhofer. Trackformer: Multi-object tracking with transformers. *CoRR*, abs/2101.02702, 2021.

[123] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer. Trackformer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021.

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

[125] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler. Online multi-target tracking using recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[126] D.-K. Nguyen, W.-L. Tseng, and H.-H. Shuai. Domain-adaptive object detection via uncertainty-aware distribution alignment. In *Proceedings of the 28th ACM International Conference on Multimedia (ACMMM)*, pages 2499–2507, 2020.

[127] S. Oh, S. Russell, and S. Sastry. Markov chain monte carlo data association for multi-target tracking. *IEEE Transactions Automatic Control*, 54(3):481–497, 2009.

[128] A. Ošep, W. Mehner, M. Mathias, and B. Leibe. Combined image- and world-space tracking in traffic scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1988–1995. IEEE, 2017.

[129] J. Pang, Q. Linlu, L. Xia, C. Haofeng, Q. Li, D. Trevor, and Y. Fisher. Quasi-dense similarity learning for multiple object tracking. In *Proceedings*

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 164–173. IEEE, 2021.

[130] I. Papakis, A. Sarkar, and A. Karpatne. Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*, 2020.

[131] S. Peize, J. Yi, Z. Rufeng, X. Enze, C. Jinkun, H. Xinting, K. Tao, Y. Zehuan, W. Changhu, and L. Ping. Transtrack: Multiple-object tracking with transformer. *CoRR*, abs/2012.15460, 2020.

[132] J. Peng, C. Wang, F. Wan, Y. Wu, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, and Y. Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 145–161. Springer, 2020.

[133] J. Pont-Tuset, F. Perazzi, S. Caelles, P. Arbeláez, A. Sorkine-Hornung, and L. Van Gool. The 2017 davis challenge on video object segmentation. *arXiv:1704.00675*, 2017.

[134] C. Qi, O. Wanli, L. Hongsheng, W. Xiaogang, L. Bin, and Y. Nenghai. Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 4836–4845. IEEE, 2017.

[135] L. Qi, L. Wang, J. Huo, L. Zhou, Y. Shi, and Y. Gao. A novel unsupervised camera-aware domain adaptation framework for person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8080–8089. IEEE, 2019.

[136] A. Rangesh, P. Maheshwari, M. Gebre, S. Mhatre, V. Ramezani, and M. M. Trivedi. Trackmpnn: A message passing graph neural architecture for multi-object tracking. *arXiv:2010.12138*, 2021.

[137] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE/CVF Confer-*

ence on Computer Vision and Pattern Recognition (CVPR), pages 779–788. IEEE, 2016.

[138] D. B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions Automatic Control*, 24(6):843–854, 1979.

[139] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems (NeurIPS)*, 28:91–99, 2015.

[140] S. H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid. Joint probabilistic data association revisited. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3047–3055. IEEE, 2015.

[141] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 17–35. Springer, 2016.

[142] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016.

[143] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention (MICCAI)*, pages 234–241. Springer, 2015.

[144] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[145] A. Sadeghian, A. Alahi, and S. Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 300–311. IEEE, 2017.

[146] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Strong-weak distribution alignment for adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6956–6965. IEEE, 2019.

[147] F. Saleh, S. Aliakbarian, H. Rezatofighi, M. Salzmann, and S. Gould. Probabilistic tracklet scoring and inpainting for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14329–14339. IEEE, 2021.

[148] M. Santiago, G. Michael, D. Dengxin, and V. G. Luc. Pathtrack: Fast trajectory annotation with path supervision. In *IEEE International Conference on Computer Vision (ICCV)*, pages 290–299. IEEE, 2017.

[149] S. Schulter, P. Vernaza, W. Choi, and M. Chandraker. Deep network flow for multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6951–6960. IEEE, 2017.

[150] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Journal of Computer Vision (IJCV)*, pages 1573–1405. Springer, 2019.

[151] C. Shan, C. Wei, B. Deng, J. Huang, X.-S. Hua, X. Cheng, and K. Liang. Tracklets predicting based adaptive graph tracking. *arXiv preprint arXiv:2010.09015*, 2020.

[152] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[153] H. Sheng, Y. Zhang, J. Chen, Z. Xiong, and J. Zhang. Heterogeneous association graph fusion for target association in multiple object tracking. *IEEE Transactions on Circuits and Systems for Video Technology (TCSVT)*, 2018.

[154] B. Shuai, A. Berneshawi, X. Li, D. Modolo, and J. Tighe. Siammot: Siamese multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12372–12382, 2021.

[155] J. Son, M. Baek, M. Cho, and B. Han. Multi-object tracking with quadruplet convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5620–5629. IEEE, 2017.

[156] L. Song, C. Wang, L. Zhang, B. Du, Q. Zhang, C. Huang, and X. Wang. Unsupervised domain adaptive re-identification: Theory and practice. *Pattern Recognition*, 102:107173, 2020.

[157] D. Sugimura, K. M. Kitani, T. Okabe, Y. Sato, and A. Sugimoto. Using individuality to track individuals: Clustering individual trajectories in crowds using local appearance and frequency trait. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1467–1474. IEEE, 2009.

[158] R. Sundararaman, C. De Almeida Braga, E. Marchand, and J. Pettre. Tracking pedestrian heads in dense crowd. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3865–3875, 2021.

[159] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Subgraph decomposition for multi-target tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5033–5041. IEEE, 2015.

[160] S. Tang, B. Andres, M. Andriluka, and B. Schiele. Multi-person tracking by multicut and deep matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 100–111. Springer, 2016.

[161] S. Tang, M. Andriluka, B. Andres, and B. Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3539–3548. IEEE, 2017.

[162] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in neural information processing systems (NeurIPS)*, volume 16, page 25, 2004.

[163] P. Thomas, L. Pandikow, A. Kim, M. Stanley, and J. Grieve. Open synthetic dataset for improving cyclist detection. In *https://paralleldomain.com/open-datasets/bicycle-detection*, 2021.

[164] Z. Tian, C. Shen, H. Chen, and T. He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 9627–9636, 2019.

[165] T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 2012.

[166] P. Tokmakov, J. Li, W. Burgard, and A. Gaidon. Learning to track with object permanence. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 10860–10869. IEEE, 2021.

[167] L. Tsung-Yi, M. Michael, B. Serge, H. James, P. Pietro, R. Deva, D. Piotr, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.

[168] J. Valmadre, A. Bewley, J. Huang, C. Sun, C. Sminchisescu, and C. Schmid. Local metrics for multi-object tracking. *arXiv preprint arXiv:2104.02631*, 2021.

[169] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[170] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe. Mots: Multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7942–7951, 2019.

[171] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe. MOTS: Multi-object tracking and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7942–7951. IEEE, 2019.

[172] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li. Unsupervised deep tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1308–1317. IEEE, 2019.

[173] Q. Wang, Y. Zheng, P. Pan, and Y. Xu. Multiple object tracking with correlation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3876–3886. IEEE, 2021.

[174] S. Wang and C. C. Fowlkes. Learning optimal parameters for multi-target tracking with contextual interactions. *International Journal of Computer Vision (IJCV)*, 122(3):484–501, 2017.

[175] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao. Pvtv2: Improved baselines with pyramid vision transformer. *arXiv preprint arXiv:2106.13797*, 2021.

[176] Y. Wang, K. Kitani, and X. Weng. Joint object detection and multi-object tracking with graph neural networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 13708–13715. IEEE, 2021.

[177] M. Weber, J. Xie, M. Collins, Y. Zhu, P. Voigtlaender, H. Adam, B. Green, A. Geiger, B. Leibe, D. Cremers, et al. Step: Segmenting and tracking every pixel. *arXiv preprint arXiv:2102.11859*, 2021.

[178] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu. Ua-detrac: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding (CVIU)*, 193:102907, 2020.

[179] X. Weng, J. Wang, D. Held, and K. Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10359–10366. IEEE, 2020.

[180] X. Weng, Y. Wang, Y. Man, and K. M. Kitani. Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6499–6508. IEEE, 2020.

[181] X. Weng, Y. Yuan, and K. Kitani. Joint 3d tracking and forecasting with graph neural network and diversity sampling. *arXiv preprint arXiv:2003.07847*, 2020.

[182] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, 2013.

[183] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 37(9):1834–1848, 2015.

[184] Y. Xiang, A. Alahi, and S. Savarese. Learning to track: Online multi-object tracking by decision making. In *IEEE International Conference on Computer Vision (ICCV)*, pages 4705–4713. IEEE, 2015.

[185] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1903–1911. IEEE, 2015.

[186] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. Joint detection and identification feature learning for person search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3415–3424, 2017.

[187] C.-D. Xu, X.-R. Zhao, X. Jin, and X.-S. Wei. Exploring categorical regularization for domain adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11724–11733. IEEE, 2020.

[188] Y. Xu, Y. Ban, G. Delorme, C. Gan, D. Rus, and X. Alameda-Pineda. Transcenter: Transformers with dense queries for multiple-object tracking. *arXiv preprint arXiv:2103.15145*, 2021.

[189] Y. Xu, A. Osep, Y. Ban, R. Horaud, L. Leal-Taixé, and X. Alameda-Pineda. How to train your deep multi-object tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6787–6796. IEEE, 2020.

[190] F. Yang, W. Choi, and Y. Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR*, pages 2129–2137. IEEE, 2016.

[191] F. Yang, H. Yang, J. Fu, H. Lu, and B. Guo. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5791–5800. IEEE, 2020.

[192] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5147–5156. IEEE, 2016.

[193] L. Yang, Y. Fan, and N. Xu. Video instance segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 5188–5197, 2019.

[194] E. Yu, Z. Li, S. Han, and H. Wang. Relationtrack: Relation-aware multiple object tracking with decoupled representation. *arXiv:2105.04322*, 2021.

[195] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pages 2636–2645. IEEE, 2020.

[196] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan. Poi: Multiple object tracking with high performance detection and appearance feature. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 36–42. Springer, 2016.

[197] H. Yu, Y. Zhou, J. Simmons, C. P. Przybyla, Y. Lin, X. Fan, Y. Mi, and S. Wang. Groupwise tracking of crowded similar-appearance targets from low-continuity image sequences. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 952–960. IEEE, 2016.

[198] F. Zeng, B. Dong, T. Wang, X. Zhang, and Y. Wei. End-to-end multiple-object tracking with transformer. In *arXiv:2105.03247*, 2021.

[199] S. Zhang, R. Benenson, and B. Schiele. Citypersons: A diverse dataset for pedestrian detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3221. IEEE, 2017.

[200] X. Zhang, J. Cao, C. Shen, and M. You. Self-training with progressive augmentation for unsupervised cross-domain person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8222–8231. IEEE, 2019.

[201] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Ke, and Z. Xiong. Multiplex labeling graph for near-online tracking in crowded scenes. *IEEE Internet of Things Journal*, 7(9):7892–7902, 2020.

[202] Y. Zhang, P. Sun, Y. Jiang, D. Yu, Z. Yuan, P. Luo, W. Liu, and X. Wang. Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*, 2021.

[203] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *International Journal of Computer Vision (IJCV)*, 2021.

[204] J. Zheng, C. Ma, H. Peng, and X. Yang. Learning to track objects from unlabeled videos. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 13546–13555. IEEE, 2021.

[205] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*, pages 1116–1124. IEEE, 2015.

[206] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian. Person re-identification in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1367–1376. IEEE, 2017.

[207] Z. Zhong, L. Zheng, S. Li, and Y. Yang. Generalizing a person retrieval model hetero- and homogeneously. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–188. Springer, 2018.

[208] Z. Zhong, L. Zheng, Z. Luo, S. Li, and Y. Yang. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 598–607. IEEE, 2019.

[209] X. Zhou, V. Koltun, and P. Krähenbühl. Tracking objects as points. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 474–490. Springer, 2020.

[210] X. Zhou, D. Wang, and P. Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[211] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang, and M.-H. Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 366–382. IEEE, 2018.

[212] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of*

the IEEE/CVF international conference on computer vision (ICCV), pages 2223–2232. IEEE, 2017.

[213] X. Zhu, J. Pang, C. Yang, J. Shi, and D. Lin. Adapting object detectors via selective cross-domain alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 687–696. IEEE, 2019.

[214] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*, 2020.

[215] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 289–305. Springer, 2018.