Learning of Global Objective for Network Flow in Multi-Object Tracking

Shuai Li[1] Yu Kong[1] Hamid Rezatofighi[2]
[1]Rochester Institute of Technology [2]Monash University
{sl6009, Yu.Kong}@rit.edu hamid.rezatofighi@monash.edu

Abstract

This paper concerns the problem of multi-object tracking based on the min-cost flow (MCF) formulation, which is conventionally studied as an instance of linear program. Given its computationally tractable inference, the success of MCF tracking largely relies on the learned cost function of underlying linear program. Most previous studies focus on learning the cost function by only taking into account two frames during training, therefore the learned cost function is sub-optimal for MCF where a multi-frame data association must be considered during inference. In order to address this problem, in this paper we propose a novel differentiable framework that ties training and inference together during learning by solving a bi-level optimization problem, where the lower-level solves a linear program and the upper-level contains a loss function that incorporates global tracking result. By back-propagating the loss through differentiable layers via gradient descent, the globally parameterized cost function is explicitly learned and regularized. With this approach, we are able to learn a better objective for global MCF tracking. As a result, we achieve competitive performances compared to the current state-of-the-art methods on the popular multi-object tracking benchmarks such as MOT16, MOT17 and MOT20.

1. Introduction

While being a classical problem, multi-object tracking (MOT) [37, 52] has been yet one of the most active research areas in computer vision, as being a fundamental basic-level perception task for many real-world problems, *e.g.*, in visual surveillance, and autonomous driving [20]. Thanks to the great progress in object detection [18, 42] techniques, "tracking-by-detection" paradigm has dominated the tracking community recently. Given an input video, a set of detection hypotheses is first generated for each frame and the goal of tracking is to associate these detection responses across time, locally or globally, to form all the trajectories. Among various previous works, minimum-cost network-flow [4, 41, 60] based methods have gained increasingly attention due to its fast inference property. In this work, we specifically focus on the network-flow based tracking.

The min-cost network flow formulation for multi-object tracking problem is, indeed, an instance of constrained integer linear program (ILP) with uni-modular constraint matrices [60]. Therefore, the solution to such an ILP problem can be optimally obtained by solving its relaxed version, *i.e.* a constrained linear program (LP), which has an identical optimal integer solution to its ILP counterpart [2]. Given its computationally tractable inference, the success of a network-flow based multi-object tracking approach largely depends on designing a proper cost function. Many previous works have focused on learning a robust objective function, *e.g.* the matching cost, between detections in two frames utilizing a neural network trained using a, *e.g.* binary cross-entropy [53], triplet [11] or contrastive [34] loss. The major drawback of these approaches is that they only consider a limited temporal context during training, *e.g.*, two or three frames. As such, the learned objective function can be sub-optimal as it ignores long-term temporal contexts and associations. Several recent works adopt graph neural networks [8, 14, 39] in order to learn a better feature representation for a spatio-temporal graph formulating multi-object tracking problem. However, their training objectives are still constrained locally, *e.g.*, a binary cross-entropy as a local edge loss is employed during the training stage and thus, knowledge about the global tracking result is not yet properly incorporated.

Recently, there have been few attempts to learn a proper objective function for an LP problem representing a global data association in MOT [19, 50]. Seminal work of [50] adds a log-barrier term into the objective function and adopts a change of basis technique to deal with equality constraints in the linear program, as a result, heuristics are involved in choosing the optimal temperature parameters during the interior method's optimization process, and the tracking results are inferior compared to many top-ranked methods in tracking benchmarks [20, 37]. The work of [19] performs 3D tracking on video and LiDAR data, followed by a re-projection step to perform 2D tracking, which is computationally demanding. In contrast to their works, we

propose a general framework which adopts Bi-level optimization technique embraced with implicit function theorem [22] to perform end-to-end learning of a global cost function for LP based tracking. At the lower-level of our optimization, our framework solves a linear program and the upper-level contains a general loss function that regularizes the tracking solution. By approximating the original linear program as a continuous quadratic program during the forward pass, it is possible to differentiate through the optimal KKT conditions of the relaxed convex quadratic problem. In this way, the cost for data association can be trained end-to-end by back-propagating the gradient of the loss through differentiable layers. In addition, we integrate a stronger observation model [5] compared to [50], together with the learned optimal cost function for data association. Our framework achieves competitive results compared to current state-of-the-art approaches on MOT16, MOT17 and MOT20 benchmarks.

In summary, our main contributions are as follows:

- We adopt the classical min-cost network flow formulation to address multi-object tracking problem and propose a novel bi-level optimization technique which is able to learn a global cost for tracking directly from multi-frame data association results.
- In order to address the non-differentiable problem of the constrained linear program, we propose to approximate the original integer linear program as a *continuous* quadratic program and to back-propagate quadratic program solution's gradients w.r.t. the model's parameters.
- The proposed tracking method achieves results comparable to current state-of-the-art trackers on the popular MOT16, MOT17 and MOT20 benchmark, demonstrating its effectiveness.

2. Related Works

Multi-Object Tracking Multi-object tracking remains an active field in computer vision for many years. The way of solving multi-object tracking can be roughly divided into two mainstream approaches namely online and offline methods. Online methods, make the decision by the observations up to the current frame. Popular approach of [55] employs Hungarian Algorithm [33] to associate observations to tracked objects first then use Kalman filter to update the object's states in an recursive manner. JPDAF [44] extends Global Nearest Neighbor matching principle by allowing all observations to track associations within certain gating areas, making the solutions more robust at the cost of heavy computation. MCMC based data association methods [9, 29] provide a probabilistic formulation of data association and therefore incorporate arbitrary priors. There also exists tracking methods which make use of deep neural networks. Seminal work of Milan *et al.* [38] uses LSTM to address state estimation and data association jointly. Later work such as Fang's *et al.* [17] tracked each object through RNN in real-time by coupling the internal and external memory cells. Although online methods can be used in time-critical situations, they make non-reversible decisions, due to the greedy fashion in the data association step.

Offline methods [26, 51] for MOT usually constructs a graph whose nodes are the detection hypothesis and edges are the potential links between detection hypothesis, by optimizing a well designed objective function with physically plausible constraints, the final tracking solution can be found. Among them, network-flow [4, 41, 60] based approaches have become popular due to its fast inference and global-optimal solution property, while more robust solutions can be achieved by employing higher-order terms [12, 26, 46, 54] at the cost of heavy computation.

Graph Neural Networks for MOT Multi-object tracking is essentially a graph optimization problem, and there are several works that attempt to solve tracking by adopting Graph Neural Networks [21, 32]. Early work of [27] combines CNN and LSTM to learn appearance and motion features together followed by GCN [32] for feature refinement. Li *et al*. [35] designs an appearance and motion graph network separately for feature learning using a modified message passing network, for online tracking. The recent work from Dai *et al*. [14] clusters and ranks tracklets through a graph convolution network and shows promising results. Braso and Leal-Taxie [8] leverages message passing networks for network flow based tracking but their work optimizes a binary cross-entropy loss during training and does not allow learning from data association directly, also a heuristic rounding step is applied to ensure disjoint path constraints. By contrast, our method directly perform back-propagation from data association and do not require heuristics at inference stage.

End-to-End Learning in MOT There exists several works that attempt to learn affinity measure for tracking in an end-to-end fashion. The framework in [57] utilizes a GRU to approximately differentiate Hungarian Algorithm and achieves descent performance. Burke and Ramamoorthy [10] adopts Sinkhorn Network within Kalman Filtering framework to learn association costs using an EM algorithm, but can only track fixed objects. Similar to our work, Papakis *et al*. [39] proposes a differentiable matching layer using Sinkhorn Network, while our work is a generalization of their work in multi-frame case. Peng *et al.* [40] perform joint detection and data association using a deep CNN. He *et al*. [23] presents an end-to-end learnable graph matching method but its quadratic formulation largely slows down the inference speed due to its exponential complexity. All of these works perform learning in an online manner. In contrast, our work performs end-to-end learning for global data association from multiple frames and is more robust than online methods during inference.

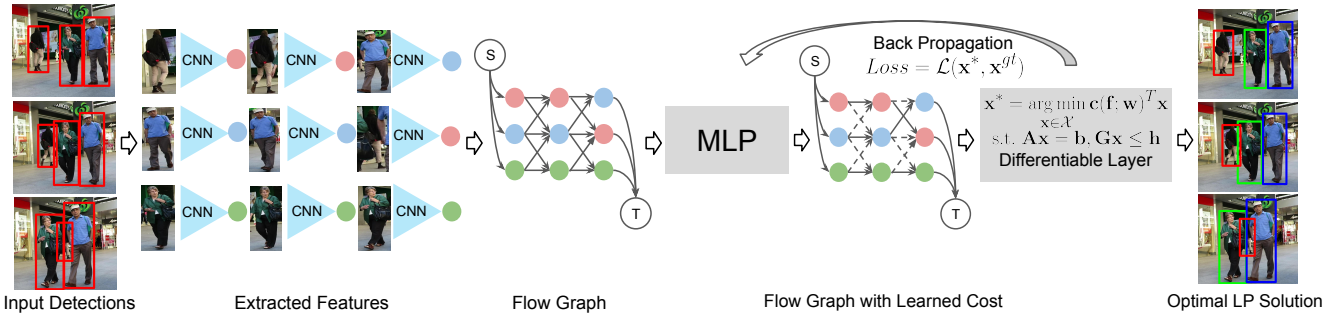

Figure 1. Illustration of the proposed tracking method. Given a sequence of frames and a set of detection hypotheses as input. Detection's appearance features extracted by a pretrained person Re-Identification network are combined with geometric cues to build a directed flow graph where detections represent nodes and edges connect detections across frames. A MLP is used to regress the linking probability between detections. During training the lower-level linear program generates a prediction $\mathbf{x}^*$, which is passed through a differentiable layer to produce a loss $\mathcal{L}$ from upper-level, the loss is back-propagated through previous layers in order to learn an optimal parameterized cost $\mathbf{c}(\mathbf{f};\mathbf{w})$. At inference time the model outputs data association by solving a linear program and achieves tracking. $\mathbf{A}, \mathbf{b}, \mathbf{G}, \mathbf{h}$ denote the Linear Program's constraints.

3. Approach

In this section, we briefly revisit the minimum cost network flow formulation for solving multi-object tracking problem, then present our proposed end-to-end learning strategy which performs back-propagation through optimal linear program solutions that learns a suitable cost function for tracking task.

3.1. Minimum-Cost Network Flow Problem

Given a set of detection hypothesis $\mathcal{D} = \{\mathbf{d}_i\}$, where $\mathbf{d}_i = (t_i, x_i, y_i, w_i, h_i, s_i)$ denotes a detection at frame t_i located at position x_i, y_i with bounding box size w_i, h_i and confidence score s_i respectively, the goal of tracking is to seek a set of K trajectories $\mathcal{T} = \{T_k\}$ according to bayes rule, which maximizes the posterior probability of data association given input detections: $P(\mathcal{T}|\mathcal{D}) = \frac{P(\mathcal{D}|\mathcal{T})P(\mathcal{T})}{P(\mathcal{D})}$, where $P(\mathcal{D})$ is a normalizing constant that does not influence the solution. Assuming the tracks are independent with each other, and the detections are conditionally independent given the tracks, we aim to optimize:

$$\begin{aligned}\mathcal{T}^* &= \arg\max_{\mathcal{T}} p(\mathcal{T}) \cdot \prod_i p(\mathbf{d}_i|\mathcal{T}) \\ &= \arg\max_{\mathcal{T}} \prod_k p(T_k) \cdot \prod_i p(\mathbf{d}_i|\mathcal{T})\end{aligned} \tag{1}$$

where $p(\mathbf{d}_i|\mathcal{T})$ is the likelihood of observing detection $\mathbf{d}_i$ within a track, a Bernoulli distribution is modeled with the output of pedestrian detection. $p(T_k)$ denotes the probability of choosing a sequence of detections for track T_k. A first-order Markovian assumption is placed for a specific track T_k, the probability can be factorized as:

$$p(T_k) = p_{en}(\mathbf{d}_1)\left(\prod_{i=1}^{l-1} p_{tran}(\mathbf{d}_{i+1}|\mathbf{d}_i)\right) p_{ex}(\mathbf{d}_l) \tag{2}$$

Specifically, $p_{en}(\mathbf{d}_1), p_{ex}(\mathbf{d}_l)$ denotes the trajectory having length l, the flow enters at detection $\mathbf{d}_1$ and exits at detection $\mathbf{d}_l$, $p_{tran}(\mathbf{d}_{i+1}|\mathbf{d}_i)$ models the temporal transition prior that detection $\mathbf{d}_{i+1}$ follows $\mathbf{d}_i$ within a certain trajectory. We formulate the above tracking problem as a minimum cost network-flow problem by taking the negative logarithm of Eq 1, and incorporating the disjoint path constraint as well as flow conservation constraint that flow coming to a node is equal to the flow coming out of a node [60]. As such, the above problem can be converted to a constrained integer linear program:

$$\begin{aligned}\mathbf{x}^* &= \arg\min_{\mathbf{x}\in\mathcal{X}} \mathbf{c}(\mathbf{f};\mathbf{w})^T\mathbf{x} \\ &\text{s.t. } \mathbf{Ax} = \mathbf{b}, \mathbf{Gx} \leq \mathbf{h}\end{aligned} \tag{3}$$

where $\mathbf{x} \in \{0,1\}^n$ is a binary vector consisting of all the edges in the flow graph. $\mathbf{c}(\mathbf{f};\mathbf{w})$ is the parameterized cost function with $\mathbf{f}$ denotes all features in the graph and $\mathbf{w}$ the weights of the MLP. $\mathbf{A} \in \mathbb{R}^{2m\times n}, \mathbf{b} \in \mathbb{R}^{2m}$ and $\mathbf{G} \in \mathbb{R}^{2m\times n}, \mathbf{h} \in \mathbb{R}^{2m}$ denote the equality and inequality constraints of the linear program respectively, where m is the number of detections in the flow graph. Note that although we adopt an interior point method to solve the linear program, other optimization techniques, such as maximum-weighted clique, K-Shortest path algorithm can be adopted during inference, given the designed cost function.

3.2. End-to-End Learning of Cost Functions for Min-Cost Flow

Figure 1 illustrates our proposed training pipeline. At the lower level, we solve a linear program, where the cost $\mathbf{c} = [\mathbf{c}^{det}, \mathbf{c}^{en}, \mathbf{c}^{ex}, \mathbf{c}^{tran}]$. Specifically, for a detection $\mathbf{d}_i$, $\mathbf{c}_i^{det}$ is the detection cost, $\mathbf{c}_i^{en}, \mathbf{c}_i^{ex}$ are designed such that a track starts or ends at this detection. $\mathbf{c}^{tran}$ is a vector consisting of transition costs between two detections. The loss $\mathcal{L}$ at upper-level characterizes the difference between the solution produced by the LP during forward pass and the corresponding ground truth. In order to learn the parameterized cost function $\mathbf{c}(\mathbf{f}; \mathbf{w})$ in Eq. 3, we need to calculate the gradient of loss w.r.t. $\mathbf{w}$: $\frac{d\mathcal{L}}{d\mathbf{w}} = \frac{d\mathcal{L}}{d\mathbf{x}^*}\frac{d\mathbf{x}^*}{d\mathbf{c}}\frac{d\mathbf{c}}{d\mathbf{w}}$. Therefore, our formulation requires differentiating through the optimal linear program's solution during training. $\frac{d\mathcal{L}}{d\mathbf{x}}$ and $\frac{d\mathbf{c}}{d\mathbf{w}}$ are easy to calculate, while computing $\frac{d\mathbf{x}^*}{d\mathbf{c}}$ is difficult, which requires differentiating through an $\arg\min$ operator. Furthermore, the solutions of the linear program are inherently discrete and need to satisfy certain constraints which further complicates the problem.

Back-propagation through Linear Program Inspired by the work of Amos [1], we propose to back-propagate through linear program's solution at the optimal Karush–Kuhn–Tucker (KKT) condition, and explicitly adopt the implicit function theorem [3, 22] at this condition. Specifically, for our optimization problem in Eq. 3, its Lagrangian is given by:

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T(\mathbf{G}\mathbf{x} - \mathbf{h}) + \boldsymbol{\nu}^T(\mathbf{A}\mathbf{x} - \mathbf{b}) \quad (4)$$

where $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$ is the linear objective, $\boldsymbol{\lambda} \geq 0$ and $\boldsymbol{\nu}$ are the dual variables which correspond to inequality and equality constraints, respectively. Taking into account stationarity condition, complementary slackness and primal feasibility of the convex problem's KKT condition and apply implicit function theorem on top of these equations, we can get the following matrix equation [3]:

$$\begin{bmatrix} \nabla^2_{\mathbf{x}} f(\mathbf{x}) & \mathbf{G}^T & \mathbf{A}^T \\ diag(\boldsymbol{\lambda})\mathbf{G} & diag(\mathbf{G}\mathbf{x} - \mathbf{h}) & \mathbf{0} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \frac{d\mathbf{x}}{d\mathbf{c}} \\ \frac{d\boldsymbol{\lambda}}{d\mathbf{c}} \\ \frac{d\boldsymbol{\nu}}{d\mathbf{c}} \end{bmatrix} = \begin{bmatrix} -\frac{d\nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})}{d\mathbf{c}} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \quad (5)$$

Here, $diag(\cdot)$ operation converts a vector into a diagonal matrix. By solving Eq. 5, the desired Jacobian $\frac{d\mathbf{x}}{d\mathbf{c}}$ can be acquired. However, directly doing so is not feasible, since $\nabla^2_{\mathbf{x}} f(\mathbf{x})$ would become $\mathbf{0}$ due to the linear objective. As a result, the left-hand side of Eq. 5 would become singular, and $\frac{d\mathbf{x}}{d\mathbf{c}}$ would be trivials.

In order to tackle this problem, we propose to add a Tikhonov damping term γ into the original linear objective so that $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \gamma||\mathbf{x}||_2^2$, and further relax the $\mathbf{x}$ such that $\mathbf{x} \in [0, 1]$ to enable gradient-based optimization. Therefore, the original linear program in Eq. 3 essentially becomes a *continuous* quadratic program (QP):

$$\begin{aligned} \hat{\mathbf{x}} = &\arg\min_{\mathbf{x} \in \mathcal{X}} \frac{1}{2}\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x} \\ &\text{s.t. } \mathbf{A}\mathbf{x} = \mathbf{b}, \mathbf{G}\mathbf{x} \leq \mathbf{h} \end{aligned} \quad (6)$$

In particular, $\mathbf{Q} \in \mathbb{R}^{n \times n}$, and $\mathbf{Q} \succ \mathbf{0}$, so the quadratic objective is strictly convex. Since $\nabla^2_{\mathbf{x}} f(\mathbf{x}) = \mathbf{Q}$, the new linear system can be written as Eq. 7. By solving this KKT equation during the forward pass, we can get the desired Jacobian $\frac{d\hat{\mathbf{x}}}{d\mathbf{c}}$ and back-propagate the QP in order to perform gradient-based end-to-end training.

$$\begin{bmatrix} \mathbf{Q} & \mathbf{G}^T & \mathbf{A}^T \\ diag(\boldsymbol{\lambda})\mathbf{G} & diag(\mathbf{G}\mathbf{x} - \mathbf{h}) & \mathbf{0} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \frac{d\mathbf{x}}{d\mathbf{c}} \\ \frac{d\boldsymbol{\lambda}}{d\mathbf{c}} \\ \frac{d\boldsymbol{\nu}}{d\mathbf{c}} \end{bmatrix} = \begin{bmatrix} -\frac{d\nabla_{\mathbf{x}} L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu})}{d\mathbf{c}} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \quad (7)$$

As for the lower-level's loss, we employ L_2 loss, which directly measures the difference between the predicted data association $\hat{\mathbf{x}}$ and the ground truth assignment $\mathbf{x}^{gt}$ as $||\hat{\mathbf{x}} - \mathbf{x}^{gt}||_2^2$. Note that the other loss functions between two binary vector, *e.g.*, hamming loss, can also be used here. Our experiment shows that our framework is not very sensitive to the tuned value of γ as long as γ is small. Therefore, we set $\gamma = 0.1$. The full training algorithm is detailed in Algorithm 1.

Algorithm 1 Gradient descent for end-to-end learning of cost function for network flow.

Input: Training set $\mathcal{D}_{train} = \{(\mathbf{f}_i, \mathbf{x}_i^{gt})\}_{i=1}^N$ with N flow graphs, each paired with feature representation $\mathbf{f}_i$ and ground truth data association $\mathbf{x}_i^{gt}$

Output: MLP with learned optimal model parameters $\mathbf{w}^*$

Initialize learning rate α, MLP with $\mathbf{w}_0$
repeat (For each iteration t)
  Randomly sample M graphs from $\mathcal{D}_{train}$.
  for $i = 1$ to M **do**
    Forward $\mathbf{f}_i$ to MLP to obtain cost $\mathbf{c}_i$, and solve Eq. 6 to get $\hat{\mathbf{x}}_i$, calculate loss: $\mathcal{L}(\hat{\mathbf{x}}_i, \mathbf{x}_i^{gt})$
    Calculate gradient of $\mathcal{L}$: $\frac{d\mathcal{L}}{d\mathbf{w}_i} = \frac{d\mathcal{L}}{d\hat{\mathbf{x}}_i}\frac{d\hat{\mathbf{x}}_i}{d\mathbf{c}_i}\frac{d\mathbf{c}_i}{d\mathbf{w}_i}$.
  end for
  Set $\frac{d\mathcal{L}}{d\mathbf{w}_t} = \frac{1}{M}\sum_{i=1}^M \frac{d\mathcal{L}}{d\mathbf{w}_i}$, perform gradient descent step: $\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha\frac{d\mathcal{L}}{d\mathbf{w}_t}$
until Convergence

Return Learned MLP parameterized by $\mathbf{w}^*$

3.3. Network Flow Cost Function

Considering trained MLPs with parameters $\mathbf{w}^*$, it is possible to design the unary and binary potentials defined in the flow graph for inference.

Detection Cost. Given a detection $\mathbf{d}_i$, the unary cost $\mathbf{c}_i^{det}$ is defined as $-s_i$, where s_i is the detection confidence output by a class-specific classifier. This term favors high-confidence person detections to be selected in the tracking results.

Entry/Exit Cost. These costs are learned scalars such that during linear program's inference, a longer-track is more likely to be selected. We cross-validated on the training set and found that a scalar 1 works well in practice. Note that a high entry/exit cost trivially yields all-zero solution of the LP as it increases total cost/energy.

Transition Cost. Given a pair of detections $\mathbf{d}_i^t$ and $\mathbf{d}_j^{t+1}$ and their edge feature representation $\mathbf{f}_{ij}$, their matching probability is: $p_{tran}(\mathbf{d}_j|\mathbf{d}_i) = \text{MLP}(\mathbf{f}_{ij}; \mathbf{w}^*)$. The corresponding item in $\mathbf{c}^{tran}$ is set to $-\log p_{tran}(\mathbf{d}_j|\mathbf{d}_i)$, which means detection pairs with high matching probabilities should be connected.

4. Experiments

4.1. Datasets

We conduct the experiments on MOT16, MOT17 [37] and MOT20 [15] pedestrian tracking dataset. MOT16 and MOT17 contain the same videos, except that MOT16 applies DPM [18] detections as input for tracking, while MOT17 evaluates tracking performance under three different detection inputs, namely DPM [18], FRCNN [42] and SDP [58]. Further, MOT20 has been designed to challenge the tracking algorithm's ability to track crowded scenes [15].

Since MOT16 and MOT17 has almost similar ground truth annotation, we train our model on the MOT16 training set. We use MOT16-09, MOT16-13 sequence to form our validation set and the remaining 5 sequences as the training set. For fair comparison with other methods, the tracking performance is reported in MOT16, MOT17 and MOT20 test set using the provided public detections.

4.2. Implementation Details

Detections. As network-flow based methods are very sensitive to false positives. We first pre-process the raw input detections using [5] provided detections as suggested by [25], in order to obtain a set of high quality detections.

Features. Suppose we have a pair of detections $\mathbf{d}_i$ and $\mathbf{d}_j$ with $(t_i, x_i, y_i, w_i, h_i, s_i)$ and $(t_j, x_j, y_j, w_j, h_j, s_j)$ respectively, we follow the work of [8] to encode spatial-temporal constraint as the geometric feature: $(\frac{2(x_j - x_i)}{h_i + h_j}, \frac{2(y_j - y_i)}{h_i + h_j}, \log \frac{h_i}{h_j}, \log \frac{w_i}{w_j})$.

This constraint encodes important relative position information between two detections, the intuition is that under a small time gap, the pedestrians cannot move far away, and the size of the pedestrian should not change much due to markovian property.

In addition to the above mentioned spatial-temporal features, we also incorporate appearance features of person detections. For this, we adopt the pre-trained deep ReID [61] architecture, which is the state-of-the-art model in the person re-identification literature, and use the pre-trained model to extract accurate appearance feature for each detection. The normalized cosine distance between two detection's appearance features output by the ReID network ϕ is concatenated with the above mentioned feature together with the generalized intersection over union (GIoU) [43] metric, which is more robust to geometric deformation, as the final edge feature for regression:

$$\mathbf{f}_{ij} = (\frac{2(x_j - x_i)}{h_i + h_j}, \frac{2(y_j - y_i)}{h_i + h_j}, \log \frac{h_i}{h_j}, \log \frac{w_i}{w_j}, \phi(\mathbf{d}_i)^T \phi(\mathbf{d}_j), \text{GIoU}(\mathbf{d}_i, \mathbf{d}_j)). \tag{8}$$

Training. In order to generate training samples, we subdivide each training sequence equally into (overlapping) T frames, where we set T = 15. Note that we only consider connections of nodes in two adjacent frames to avoid heavy memory consumption and speed up calculation. We utilize the available ground-truth annotations which defines the ideal data association between objects across frames. In particular, for each item in $\mathbf{c}^{tran}$ we define it as $-\log p_{tran}(\mathbf{d}_j|\mathbf{d}_i)$, where $p_{tran}(\mathbf{d}_j|\mathbf{d}_i) \in \{0, 1\}$ is the matching probability for ground-truth bounding box $\mathbf{d}_i$ and $\mathbf{d}_j$. -1 is used for entries in $\mathbf{c}^{det}$ as each annotated box is a true positive. For each ground-truth box $\mathbf{d}_i$, its corresponding entry/exit cost is set as: $\mathbf{c}_i^{en} = \mathbf{c}_i^{ex} = 1$, so that a track can start and terminate at any detection.

We experimented with different network architectures for scoring the cost function, *e.g.* linear classifier, MLP, and found that a two-layer multi-layer perceptron works better in practice. Therefore, we adopt a two layer MLP with ReLU non-linearity which outputs a probability distribution between 0 and 1 as affinity measure, using Adam optimizer with initial learning rate of 10^{-3} with a weight decay of 10^{-4} for around 10 epochs. We select the model that performs best on the validation set for tracking at test time.

Inference. Tracking is performed over (overlapping) batches with length of 50-150 frames depending on the detection density in a specific video. Within each batch, a maximum of $\Delta = 5$ frames gap between detections is allowed to join detections, to handle false negatives and short term occlusions. We utilize Gurobi's solver to solve each ILP within each batch to obtain tracks. Tracks are finally stitched across adjacent batches to form the final tracks.

Long-Term Occlusion Handling. Due to long-term occlusions in reality, tracks are usually fragmented and identity switches occur. In response, we propose to handle these issues using a second round of network-flow tracker, except that the nodes now are tracklets (short tracks) instead of nodes in the first round, with appearance and motion constraints. Specifically, given tracklet T_i, we use [61] to

Figure 2. Effect of applying single-object tracker. Left: Target to track. Middle: Tracker fails to track target due to missing detection. Right: The SOT tracker is able to follow target for a few frames.

extract appearance features, *e.g.* $\mathbf{f}_i^{app} = \frac{1}{L}\sum_{d=1}^{L}\mathbf{f}_d^{app}$ denotes its final representation. Therefore for track T_i and T_j, $1 - \mathbf{f}_i^{appT}\mathbf{f}_j^{app}$ is their final matching cost. Regarding motion constraint, we estimate the average velocity for each pair of temporally non-overlapping tracklet, such that τ_{dist} is used to reject physically implausible connections. As such, tracklets are joined and long-term occlusions within a track could be recovered throughs bi-linear interpolation.

Post-Processing. It is possible that our tracker lost several objects during tracking, due to the drastic appearance change/illumination, etc. In order to address this problem, we add a single object tracker (SOT) to keep track of the lost objects. Specifically, for a track that disappears before the video reaches to the last frame, we make use of the last position of the tracker to initialize the single object tracker [36] and perform tracking. In order to kill the tracker in case the tracked object confuses with other objects, we compare the tracked object's appearance with the initialized object, whenever their similarity of color histogram fall below a threshold τ, the tracker is terminated to avoid tracking false positives or drift to background. From Figure 2, by adding a SOT, our method successfully tracks the lost object, and the SOT tracker ends when a complete occlusion occurs.

4.3. Evaluation

In order to validate the effectiveness of the proposed end-to-end learning method, we compare our learned MLP against a baseline method, which is trained with binary cross-entropy (BCE) objective using annotated data association, which does not allow back-propagation from association. L_1 and L_2 are the proposed end-to-end learning approaches which adopt L_1 and L_2 as the loss, respectively. All methods takes the same edge features as input for a fair comparison.

Evaluation of Tracking. Multiple Object Tracking Accuracy (MOTA) [6] and IDF1 [45] are the two widely adopted metrics for tracking performance evaluation. The two metrics are defined as: MOTA$= 1 - \frac{\sum_t \text{FP}_t+\text{FN}_t+\text{IDS}_t}{\sum_t \text{GT}_t}$ and IDF1$= \frac{2\text{IDTP}}{2\text{IDTP}+\text{IDFP}+\text{IDFN}}$. MOTA measures the tracking error in terms of number of false positive (FP), false negative (FN) and identity switches (IDS), it focus more on the detection's perspective, while IDF1 [45] reflects the tracker's ability to maintain identities over time and thus concentrates more on tracking side. In addition, Multiple Object Tracking Precision (MOTP), the percentage of Mostly Tracked (ML) objects and the percentage of Mostly Lost (ML) objects as well as the number of total fragmentations within a trajectory (Frag) constitutes the main evaluation metrics for tracking.

Effectiveness of the Learned Affinity Measures. Area Under Curve (AUC) is suitable to measure a binary classifier's performance, therefore a stronger MLP should obtain a higher AUC score. Mean Squared Error (MSE) is adopted to compare the QP's output $\mathbf{x}^*$, including the detection, entry/exit cost, against ground-truth annotation $\mathbf{x}^{gt}$. MSE Edge solely takes into account the predicted data association compared with ground truth association. From Table 1, it is clear that by back-propagating through multi-frame data association, our proposed approach, regardless of the losses used, outperforms baseline method trained using BCE objective in all metrics used. Overall, L_1 and L_2 loss achieve similar performance.

Loss	AUC↑	BCE↓	MSE↓	MSE Edge↓
BCE	0.996	0.064	0.026	0.017
L_1	**0.997**	0.047	0.013	0.008
L_2	**0.997**	**0.005**	**0.010**	**0.006**

Table 1. Evaluation of the affinity metrics and data association results using the proposed training strategy versus baseline methods on the MOT16 validation set.

Method	MOTA ↑	IDF1 ↑	IDS ↓	MT ↑	ML ↓
w/o 2^{nd}-round MCF	38.9	43.5	134	22	57
w 2^{nd}-round MCF	**42.9**	**55.1**	**73**	**28**	**51**

Table 2. Ablation study on MOT16 validation set, first-round MCF achieves decent performance, adding second-round MCF further boosts the performance due to long-term occlusion handling.

Method	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓
Baseline	49.33	58.91	100	163	822	24814	**157**
L_1	51.45	**60.15**	123	151	**810**	23715	186
L_2	**51.54**	**60.15**	**124**	**147**	831	**23641**	193

Table 3. Evaluation of tracking performance under different approaches on MOT17 validation set, End-to-end learned cost performs better than baseline trained using BCE loss.

Method	MOTA↑	REC↑	PREC↑	MT↑	IDS↓	FRAG↓
DNF [50](Linear)	28.25	38.01	80.09	9.67	342	1620
DNF [50](MLP)	31.10	37.53	85.88	8.51	289	1562
Proposed Method	**44.27**	**45.03**	**98.84**	**14.9**	**260**	**365**

Table 4. Evaluation of tracking performance under different training strategy on MOT16 training set, ↑ means high number is better, ↓ is opposite. Best performance under each metric is shown in bold-faced font.

Effectiveness of Second Stage Network Flow Data Association. In order to outline the contribution of the first-stage data association on the final tracking performance, we test the proposed method on MOT16 validation set. According to the results in Table 2, the first stage NF data association provides decent results (as a good tracklet initialization) for the second round of network flow tracker, where long-term occlusions must be handled. We can also see the performance improvements in second stage network flow tracking compared to solely having the first stage tracking. Interms of MOTA, the improvements are not as significant as IDF1/IDS, which properly reflects the contribution of the second stage to the long-term occlusions. Therefore the core method (*i.e*. the first stage) plays a crucial rule to the final performance.

Effectiveness of The Proposed Approach over Baseline in Tracking. In Table 3, we compare our method against a NF baseline trained using BCE objective, in terms of tracking metrics. Note that for all three methods, we use the same strategy of second-round MCF for long-term occlusion handling, thus the difference of final tracking performance merely stems from the first-stage's association results. Though our method is slightly inferior in IDS metric, our proposed approach surpass the baseline in MOTA, and ML metrics, which suggests back-propagation from data association is indeed necessary to reach a stronger tracking performance.

Comparison against [50]. We compare our proposed learning method against the approach proposed by Schulter *et al*. [50] in Table 4. We followed the same train/val split as their work and cross-validated on the MOT16 training set for fair comparison. For all tracking metrics, we outperform their end-to-end learning method. The higher recall and precision can be attributed to the fact that we have a stronger observation model, as well as our better learned affinity measure for data association. We improve over the baseline MOTA by 44% and significantly increase the MT/FRAG metric.

Method	Mode	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
EAMTT [49]	Online	38.8	42.4	7.9	49.1	8114	102452	965
RAN [17]	Online	45.9	48.8	13.2	41.9	6871	91713	648
AMIR [47]	Online	47.2	46.3	14.0	41.6	**2681**	92856	774
MOTDT [11]	Online	47.6	50.9	15.2	38.3	9253	85431	792
GraphNetwork [35]	Online	47.7	43.2	16.1	34.3	9518	83875	1907
UMA [59]	Online	50.5	52.8	17.8	**33.7**	7587	81924	685
Tracktor++ [5]	Online	54.4	52.5	19.0	36.9	3280	79149	682
LINF1 [16]	Offline	41.5	45.7	11.6	51.3	7896	99224	430
BiLSTM [31]	Offline	42.1	47.8	14.9	44.4	11637	93172	753
NOMT [12]	Offline	46.4	53.3	18.3	41.4	9753	87565	359
LMP [54]	Offline	48.8	51.3	18.2	40.1	6654	86245	481
MPNTrack [8]	Offline	**58.6**	**61.7**	**27.3**	34.0	4949	**70252**	**354**
LPT(Ours)	Offline	57.4	58.7	22.7	37.2	4201	73114	427

Table 5. Tracking results on MOT16 test set with DPM [18] detections as input. Bold and underlined numbers indicate the best and the second best performances.

Method	Mode	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
DMAN [62]	Online	48.2	55.7	19.3	38.3	26128	263608	2194
MOTDT [11]	Online	50.9	52.7	17.5	35.7	24069	250768	2474
STRN [56]	Online	50.9	56.5	20.1	37.0	27,532	246924	2593
FAMNet [13]	Online	52.0	48.7	19.1	33.4	14138	253616	3072
DeepMOT [57]	Online	53.7	53.8	19.4	36.6	11731	247447	1947
Tracktor++v2 [5]	Online	56.3	55.1	21.1	35.3	**8866**	235449	1987
GCNNMatch [39]	Online	57.3	56.3	24.2	**33.4**	14100	225042	1911
MHT_DAM [30]	Offline	50.7	47.2	20.8	36.9	22875	252889	2314
jCC [28]	Offline	51.2	54.5	20.9	37.0	25937	247822	1802
JBNOT [24]	Offline	52.6	50.8	19.7	35.8	31572	232659	3050
MPNTrack [8]	Offline	58.8	61.7	**28.8**	33.5	17413	213594	**1185**
Lif_T [25]	Offline	**60.5**	**65.6**	27.0	33.6	14966	**206619**	1189
LPT(Ours)	Offline	57.3	57.7	23.3	36.9	15187	224560	1424

Table 6. Tracking results on MOT17 test set using public detections [18, 42, 58] as input. The best and second best performances are shown in bold and underlined numbers respectively.

Method	Mode	MOTA ↑	IDF1 ↑	MT ↑	ML ↓	FP ↓	FN ↓	IDS ↓
SORT20 [7]	Online	42.7	45.1	16.7	26.2	27521	264694	4470
Tracktor++V2 [5]	Online	52.6	52.7	29.4	26.7	**6930**	236680	1648
ArTist [48]	Online	53.6	51.0	31.6	28.1	7765	230576	**1531**
GCNNMatch [39]	Offline	54.5	49.0	32.8	25.5	9522	223611	2038
ApLift [26]	Offline	**58.9**	**56.5**	**41.3**	**21.3**	17739	**192736**	2241
LPT(Ours)	Offline	57.9	53.5	39.0	22.8	9980	205949	1827

Table 7. Tracking results on MOT20 test sequences with public detections. Bold and underlined numbers indicate the best and the second best performances.

Comparison with Other State-of-the-arts. We compare our tracker's performance against other tracking algorithms on MOT16, MOT17 and MOT20 benchmark. In our final results, we selected the models trained using L_2 loss for benchmark evaluation. We do not apply SOT in our final implementation as the improvements are marginal.

Table 5, 6 and 7 compares our method, namely Linear Program Tracker (LPT), against existing methods. On MOT16 benchmark, our method has second-best performace in MOTA, IDF1, MT and FN. Note that the work of [8] use Message Passing Networks that learns a better feature representation for temporal connections, while our method does not utilize Graph Networks, and our result is still comparable with their approaches.

On MOT17 benchmark, the work of DeepMOT [57] trains a neural network using the MOT Metric in an end-to-end manner. However their method is mainly optimized for improving two-frame data association. In contrast with their method, our method works in an offline manner which incorporates longer temporal contextual information both during training and inference. The results show that our method achieves better MOTA and IDF1 compared to those in [57]. A notable strong baseline is [39], which leverages a Graph Convolution Network followed by a Sinkhorn Network to perform end-to-end training of data association, we achieve similar performance in terms of MOTA compared to their approach but have better IDF1 and IDS score, due to the

Figure 3. Qualitative results of our tracking algorithm on MOT17 test split. Our method is able to track persons through long-term occlusions and also performs well in crowded scenes, best viewed in color.

advantage of accurately learned costs as well as the multi-frame data association formulation. It is expected by further incorporating Graph Convolution Network (GCN) into feature learning, the performance of our method would be better. Figure 3 shows some qualitative results, our method is able to track objects through long-term occlusions and recover missing detections.

Compared with the current SOTA method LifT [25], our method achieves slightly worse MOTA and IDS metric. It should be noted that, the work of LifT considers a lifted connections between detections that span over 50 frames, making the formulation NP-hard and computationally heavy. By contrast, our method has polynomial time complexity. Although our method is slightly worse in-terms-of performance, we achieve faster inference speed: our tracker consumes 1-5 mins while their ILP solver requires 26.6 mins per sequence in average. We believe by working with a more powerful optimization technique such as multi-cut, multiple-hypothesis tracking during inference, our performance could be further leveraged.

Finally, we test our method on MOT20 [15], which is designed for tracking the crowds. It is worth to mention that various pruning heuristics are applied in state-of-the-art method [26], in order to make their NP-hard problem tractable, while our method does not require complex pre-processing steps to sparsify the graph. Overall, our method achieves second-best on MOTA and IDF1 metrics, which are slightly inferior than [25] but have better IDs metric than their results. Thanks to the multi-frame data association used during training/inference, our performance largely exceeds [5, 39, 48] in MOTA and MT metrics.

5. Conclusion

In summary, we have presented a general framework and a novel training method for learning the cost functions of the min-cost flow multi-object tracking problem. By solving a differentiable *continuous* quadratic program (QP), our approach is able to incorporate multi-frame data association results as well as tracking specific constraints in order to obtain a better global objective for tracking. Although we perform tracking using network flow linear program inference, other formulations that respect tracking constraints can be employed for end-to-end learning as well. One major limitation of our method is that, only data association is learned end-to-end, whilst the part of object detection is separated from training. Since the success of this tracker largely relies on the quality of input detections, in the future we plan to explore training object detection jointly with our network flow framework. We also aim to consider if the end-to-end learning of a higher order optimization objective can further improve the tracking performance.

References

[1] Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pages 136–145. PMLR, 2017. 4

[2] Anton Andriyenko and Konrad Schindler. Globally optimal multi-target tracking on a hexagonal lattice. In *European Conference on Computer Vision*, pages 466–479. Springer, 2010. 1

[3] Shane Barratt. On the differentiability of the solution to convex optimization problems. *arXiv preprint arXiv:1804.05098*, 2018. 4

[4] Jerome Berclaz, Francois Fleuret, Engin Turetken, and Pascal Fua. Multiple object tracking using k-shortest paths optimization. *IEEE transactions on pattern analysis and machine intelligence*, 33(9):1806–1819, 2011. 1, 2

[5] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 941–951, 2019. 2, 5, 7, 8

[6] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008. 6

[7] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 7

[8] Guillem Brasó and Laura Leal-Taixé. Learning a neural solver for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6247–6257, 2020. 1, 2, 5, 7

[9] Ernesto Brau, Jinyan Guan, Kyle Simek, Luca Del Pero, Colin Reimer Dawson, and Kobus Barnard. Bayesian 3d tracking from monocular video. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3368–3375, 2013. 2

[10] Michael Burke and Subramanian Ramamoorthy. Learning data association without data association: An em approach to neural assignment prediction. *arXiv preprint arXiv:2105.00369*, 2021. 2

[11] Long Chen, Haizhou Ai, Zijie Zhuang, and Chong 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. IEEE, 2018. 1, 7

[12] Wongun Choi. Near-online multi-target tracking with aggregated local flow descriptor. In *Proceedings of the IEEE international conference on computer vision*, pages 3029–3037, 2015. 2, 7

[13] Peng Chu and Haibin Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6172–6181, 2019. 7

[14] Peng Dai, Renliang Weng, Wongun Choi, Changshui Zhang, Zhangping He, and Wei Ding. Learning a proposal classifier for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2443–2452, 2021. 1, 2

[15] Patrick Dendorfer, Hamid Rezatofighi, Anton Milan, Javen Shi, Daniel Cremers, Ian Reid, Stefan Roth, Konrad Schindler, and Laura Leal-Taixé. Mot20: A benchmark for multi object tracking in crowded scenes. *arXiv preprint arXiv:2003.09003*, 2020. 5, 8

[16] Loïc Fagot-Bouquet, Romaric Audigier, Yoann Dhome, and Frédéric Lerasle. Improving multi-frame data association with sparse representations for robust near-online multi-object tracking. In *European Conference on Computer Vision*, pages 774–790. Springer, 2016. 7

[17] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018. 2, 7

[18] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009. 1, 5, 7

[19] Davi Frossard and Raquel Urtasun. End-to-end learning of multi-sensor 3d tracking by detection. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 635–642. IEEE, 2018. 1

[20] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 1

[21] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017. 2

[22] Stephen Gould, Richard Hartley, and Dylan John Campbell. Deep declarative networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2, 4

[23] Jiawei He, Zehao Huang, Naiyan Wang, and Zhaoxiang 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*, pages 5299–5309, 2021. 2

[24] Roberto Henschel, Yunzhe Zou, and Bodo Rosenhahn. Multiple people tracking using body and joint detections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 0–0, 2019. 7

[25] Andrea Hornakova, Roberto Henschel, Bodo Rosenhahn, and Paul Swoboda. Lifted disjoint paths with application in multiple object tracking. In *International Conference on Machine Learning*, pages 4364–4375. PMLR, 2020. 5, 7, 8

[26] Andrea Hornakova, Timo Kaiser, Paul Swoboda, Michal Rolinek, Bodo Rosenhahn, and Roberto Henschel. Making higher order mot scalable: An efficient approximate solver for lifted disjoint paths. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6330–6340, 2021. 2, 7, 8

[27] Xiaolong Jiang, Peizhao Li, Yanjing Li, and Xiantong Zhen. Graph neural based end-to-end data association frame-

work for online multiple-object tracking. *arXiv preprint arXiv:1907.05315*, 2019. 2

[28] Margret Keuper, Siyu Tang, Bjoern Andres, Thomas Brox, and Bernt Schiele. Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE transactions on pattern analysis and machine intelligence*, 42(1):140–153, 2018. 7

[29] Zia Khan, Tucker Balch, and Frank Dellaert. An mcmc-based particle filter for tracking multiple interacting targets. In *European Conference on Computer Vision*, pages 279–290. Springer, 2004. 2

[30] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. Multiple hypothesis tracking revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 4696–4704, 2015. 7

[31] Chanho Kim, Fuxin Li, and James M Rehg. Multi-object tracking with neural gating using bilinear lstm. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 200–215, 2018. 7

[32] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 2

[33] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 1955. 2

[34] Laura Leal-Taixé, Cristian Canton-Ferrer, and Konrad Schindler. Learning by tracking: Siamese cnn for robust target association. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 33–40, 2016. 1

[35] Jiahe Li, Xu Gao, and Tingting Jiang. Graph networks for multiple object tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 719–728, 2020. 2, 7

[36] Alan Lukezic, Tomas Vojir, Luka ˇCehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6309–6318, 2017. 6

[37] Anton Milan, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016. 1, 5

[38] Anton Milan, S Hamid Rezatofighi, Anthony Dick, Ian Reid, and Konrad Schindler. Online multi-target tracking using recurrent neural networks. In *Thirty-First AAAI conference on artificial intelligence*, 2017. 2

[39] Ioannis Papakis, Abhijit Sarkar, and Anuj Karpatne. Gcnnmatch: Graph convolutional neural networks for multi-object tracking via sinkhorn normalization. *arXiv preprint arXiv:2010.00067*, 2020. 1, 2, 7, 8

[40] Jinlong Peng, Changan Wang, Fangbin Wan, Yang Wu, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Chained-tracker: Chaining paired attentive regression results for end-to-end joint multiple-object detection and tracking. In *European conference on computer vision*, pages 145–161. Springer, 2020. 2

[41] Hamed Pirsiavash, Deva Ramanan, and Charless C Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR 2011*, pages 1201–1208. IEEE, 2011. 1, 2

[42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 1, 5, 7

[43] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 658–666, 2019. 5

[44] Seyed Hamid Rezatofighi, Anton Milan, Zhen Zhang, Qinfeng Shi, Anthony Dick, and Ian Reid. Joint probabilistic data association revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 3047–3055, 2015. 2

[45] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016. 6

[46] Amir Roshan Zamir, Afshin Dehghan, and Mubarak Shah. Gmcp-tracker: Global multi-object tracking using generalized minimum clique graphs. In *European conference on computer vision*, pages 343–356. Springer, 2012. 2

[47] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *Proceedings of the IEEE international conference on computer vision*, pages 300–311, 2017. 7

[48] Fatemeh Saleh, Sadegh Aliakbarian, Hamid Rezatofighi, Mathieu Salzmann, and Stephen Gould. Probabilistic tracklet scoring and inpainting for multiple object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14329–14339, 2021. 7, 8

[49] Ricardo Sanchez-Matilla, Fabio Poiesi, and Andrea Cavallaro. Online multi-target tracking with strong and weak detections. In *European Conference on Computer Vision*, pages 84–99. Springer, 2016. 7

[50] Samuel Schulter, Paul Vernaza, Wongun Choi, and Manmohan Chandraker. Deep network flow for multi-object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6951–6960, 2017. 1, 2, 6, 7

[51] Aleksandr V Segal and Ian Reid. Latent data association: Bayesian model selection for multi-target tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2904–2911, 2013. 2

[52] Zhihong Sun, Jun Chen, Liang Chao, Weijian Ruan, and Mithun Mukherjee. A survey of multiple pedestrian tracking based on tracking-by-detection framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(5):1819–1833, 2020. 1

[53] Siyu Tang, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Multi-person tracking by multicut and deep matching. In *European Conference on Computer Vision*, pages 100–111. Springer, 2016. 1

[54] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. Multiple people tracking by lifted multicut and person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3539–3548, 2017. 2, 7

[55] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 2

[56] Jiarui Xu, Yue Cao, Zheng Zhang, and Han Hu. Spatial-temporal relation networks for multi-object tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3988–3998, 2019. 7

[57] Yihong Xu, Aljosa Osep, Yutong Ban, Radu Horaud, Laura Leal-Taixé, and Xavier Alameda-Pineda. How to train your deep multi-object tracker. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6787–6796, 2020. 2, 7

[58] Fan Yang, Wongun Choi, and Yuanqing Lin. Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2129–2137, 2016. 5, 7

[59] Junbo Yin, Wenguan Wang, Qinghao Meng, Ruigang Yang, and Jianbing Shen. A unified object motion and affinity model for online multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6768–6777, 2020. 7

[60] Li Zhang, Yuan Li, and Ramakant Nevatia. Global data association for multi-object tracking using network flows. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008. 1, 2, 3

[61] Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-scale feature learning for person re-identification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3702–3712, 2019. 5

[62] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. Online multi-object tracking with dual matching attention networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 366–382, 2018. 7