Online Multiple-Pedestrian Tracking with Detection-Pair-Based Graph Convolutional Networks

Weijiang Feng, Long Lan (*Member, IEEE*), Michael Buro, and Zhigang Luo (*Member, IEEE)*

Abstract—**The typical Internet of Things application, unattended driving systems, will need the ability to recognize relevant traffic participants and detect dangerous situations ahead of time. An important component of these systems is one that is able to distinguish pedestrians and track their motion to make intelligent driving decisions. This paper develops a high accuracy multiple pedestrian tracking algorithm which is vital for intelligent transportation. Here, we use the off-the-shelf detectors and explore the benefits of modeling pedestrian interactions, such as the interaction of two pedestrians simultaneously matched to two pedestrians in another frame, for robust detection association. Explicitly studying interactions is non-trivial. Previous works often manually selected interacting detections (or "tracklets") to simplify the association process. In this paper, we propose a novel association method based on Deep Graph Convolutional Affinity Networks (DGCANs) and extend detection-level interactions to the association-level, which treats a potential association of a detection pair as a node in the graph, and explicitly modeling the interactions among potential associations. Specifically, with the novel node, two corresponding edges are readily designed to model the compatible and colliding interactions between related associations. Our proposed method, by redefining nodes and edges, enables us to blend sufficient interaction cues from appearance and motion, and learns a robust affinity measure in an end-to-end fashion. Using the Hungarian algorithm as an online tracker, our method archives state-of-the-art performance on benchmark datasets 2D MOT15, MOT16, and MOT17.**

Index Terms—**multiple pedestrian tracking, graph convolutional network, online tracking, intelligent transportation**

I. Introduction

INTERNET of Things (IoT) is a smart network that provides connections among nearly everything around us to enable a vast number of applications. With the advance of IoT, massive number of IoT devices (sensors, processors, actuators, etc.) are deployed to enable the monitoring and manipulation of things under minor or no human intervention. Intelligent urban surveillance systems for smart cities, and unattended driving systems for smart transportation are typical IoT applications that both involve multiple pedestrian tracking as the key technology. Here is an example of the usage of multiple pedestrian tracking for an unattended driving system: self-driving cars for smart transportation need to react to external factors in real-time. The moving pedestrians always bring considerable uncertainty to the autonomous vehicle system. If a self-driving car can track the trajectories of moving pedestrians on the road, it will perceive both normal and sudden movements of its surroundings, and take proper actions accordingly. Therefore, developing a high accuracy multiple pedestrian tracking algorithm for one of the typical IoT applications, i.e., smart transportation, is important. Multi-pedestrian tracking, a very important visual perception and analysis technology, will provide a more comprehensive and accurate scene understanding and certainly boost the smartness and advance of IoT.

Multiple-pedestrian tracking aims at estimating the locations of all pedestrians in a scene and maintaining their identities across consecutive frames to generate their trajectories. Recent advances in object detection [1], [2] make it possible to generate high-quality detection responses of pedestrians in the form of bounding boxes. Using these bounding boxes, the current predominant methods follow the paradigm of "tracking-by-detection", which studies how to associate bounding boxes in different frames to produce a complete trajectory for each individual. Through this kind of data association formulation, the multiple-pedestrian tracking community has made significant progress. However, real world applications face many difficulties including frequent object occlusions, interactions among objects, missing or inaccurate detections, and false positive detections. Therefore, the multi-pedestrian tracking task is still challenging.

Depending on the frame processing method, existing multiple-pedestrian tracking methods can be categorized into online and offline methods. Online methods [3]–[6] only utilize the current and previous frame, which makes them suitable for real-time tasks. However, due to their incremental nature, they are prone to identity switching, especially in environments in which pedestrians frequently interact with each other. On the other hand, offline methods [7]–[11] can analyze the

This work was partially supported by the National Natural Science Foundation of China (No. 61906210), National Grand R&D Plan (Grant No. 2020AAA0103501). *Corresponding author: Long Lan.*

W. Feng is with The PLA Information Engineering University, Zhengzhou 450001, China and College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China (email: fengweijiang14@nudt.edu.cn).

L. Lan is with Institute for Quantum Information & State Key Laboratory of High Performance Computing, College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China (email: long.lan@nudt.edu.cn).

Z. Luo is with College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China (email: zgluo@nudt.edu.cn).

M. Buro is with Department of Computing Science, University of Alberta, Edmonton, Alberta, T6G 2E8, Canada (email: mburo@ualberta.ca).

entire frame sequence, and therefore have the opportunity to perform better than online methods. However, the more global optimization task is often computational expensive and not suitable for time-critical applications. Therefore, one current research focus is on reducing identity switches in online tracking systems.

A typical online multiple-pedestrian tracking pipeline is shown in Fig. 1(a). Assume that there are two detections $d1, d2$ in frame t and two detections $d3, d4$ in frame $t+1$, where $d1, d4$ share the same identity, as to $d2$ and $d3$. The first step is to generate all pairs containing one detection d_t^i in frame t and one detection d_{t+1}^j in frame $t+1$. (In this paper, we call a pair of detections from two different frames a potential association. We use detection-detection pair and potential association interchangeably. If the two detections of a potential association share the same identity, the potential association is true, otherwise it is false.) In the second step, each detection-detection pair (d_t^i, d_{t+1}^j) is fed into a feature extraction model, such as a Siamese Convolutional Neural Network [12], to compute feature representations of the detected pedestrians. The third step is estimating the affinity $s(d_t^i, d_{t+1}^j)$ of every potential association based on their individual representations, e.g., by applying the cosine-similarity between the representation vectors. The resulting similarity matrix can then be fed into a bipartite matching algorithm, such as the Hungarian method [13], to determine which potential association is true. This online multiple-pedestrian tracking computation pipeline is easy to implement and has achieved competitive tracking performance [14].

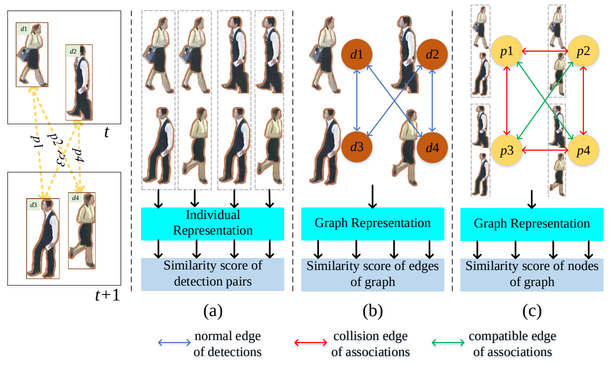

d1
d2
d3
d4
t
t+1
p1
p2
p3
p4
Individual Representation
Similarity score of detection pairs
Graph Representation
Similarity score of edges of graph
Graph Representation
Similarity score of nodes of graph
(a)
(b)
(c)
normal edge of detections
collision edge of associations
compatible edge of associations

Fig. 1. Illustration of three multiple-pedestrian tracking pipelines based on two adjacent image frames with provided detection responses (shown on the left). (a) The pipeline used by conventional methods in which the pairwise interactions between different detection-detection pairs are ignored. (b) Previous graph-based representation model that represents detections as nodes and encodes interactions between detections. (c) Our proposed method that represents potential associations as nodes. Interactions between different detection-detection pairs, including compatible interactions (green edges) and collision interactions (red edges), are incorporated in the graph representation to allow more robust and accurate affinity estimation.

Interaction information among detection-detection pairs has been shown to be beneficial in robust affinity estimation [15], [16]. An interaction between two potential associations may be *compatible*, meaning that both potential associations can be true associations, such as the interaction between associations $p2$ and $p3$, and the interaction between associations $p1$ and $p4$ in the left sub-figure of Fig 1; it may also be *mutually exclusive* (or *conflicting*), meaning that at most one of the two potential associations can be true, such as the interaction between associations $p1$ and $p2$, and the interaction between associations $p3$ and $p4$ in the left sub-figure of Fig 1. Associations $p2$ and $p3$ are similar in the sense that the two detections of each association share the same identity, i.e., detections d_t^1 and d_{t+1}^4 of association $p2$ share the same identity, detections d_t^2 and d_{t+1}^3 of association $p3$ share the same identity as well. Besides, associations $p1$ and $p4$ are similar in the sense that the two detections of each association have different identities, i.e., detections d_t^1 and d_{t+1}^3 of association $p1$ have different identities, detections d_t^2 and d_{t+1}^4 of association $p4$ have different identities as well. As a result, both the two associations connected by a compatible interaction should have high similarity scores (e.g., both associations $p2$ and $p3$ should have high similarity scores) or low similarity scores (e.g., both associations $p1$ and $p4$ should have low similarity scores). Thus, the similarity score of each of the two compatible associations can be inferred from each other, and we can learn better similarity scores for the two compatible associations by utilizing the interaction information between them. On the contrary, associations $p1$ and $p2$ are conflicting, i.e., at most one of these two associations can have a high similarity score, since at most one detections at frame $t+1$ can be associated to the detection d_t^1. The high similarity score of association $p1$ indicates that we should have a low similarity score for association $p2$, and vice versa. Thus we can estimate better similarity scores for conflicting associations by taking their mutually exclusive interaction information into consideration.

However, conventional methods only estimate the affinity of potential associations of detection-detection pairs individually, and ignore the pairwise interactions among detection-detection pairs. For instance, when we estimate the affinity between detection d_t^1 in frame t and detection d_{t+1}^4 in frame $t+1$, conventional methods only consider the single (d_t^1, d_{t+1}^4) pair. The interactions among detection-detection pairs, like the compatible interaction between (d_t^1, d_{t+1}^4) and (d_t^2, d_{t+1}^3), colliding interaction between (d_t^1, d_{t+1}^4) and (d_t^1, d_{t+1}^3), and the colliding interaction between (d_t^1, d_{t+1}^4) and (d_t^2, d_{t+1}^4), are ignored. In this case, some true associations (e.g., associations of two detections from the same pedestrian that have different pose, orientation, illumination, etc.) may have smaller similarity scores than some false associations (e.g., associations of two detections from different pedestrians that have similar appearance and are close to each other), making it difficult for the bipartite matching algorithm to find good associations.

Recently, researchers have tried to address the above problem using graph neural networks (GNNs), since GNNs provide a natural way of modeling the associations of separate detections. For example, Brasó *et al.* [9] relate MOT to the classical network flow framework. They treat each detection as a node in a graph, and each pair of detections as an edge of the graph. Binary flow variables for every edge in the graph are predicted by a novel graph message-passing network that is able to capture the graph structure of the MOT problem. However, as shown in Fig. 1(b), these methods regard detections as nodes and potential associations of detections as edges, making it difficult to explicitly explore the interactions between associations of pedestrians. The definition of nodes and edges in all aforementioned methods only allows them to consider the association between two detections implicitly through graph operations.

Inspired by the importance of interactions among detection-detection pairs and the limitation of previous graph representation models, in this paper we propose to utilize the graph convolutional network (GCN), a kind of GNN, to explicitly encode the interactions between potential associations of pedestrian-pedestrian pairs in a learning manner. GCNs process graph-structured data—generalizing CNNs. In GCNs, the representation of a node is correlated with those of its neighbors by the graph convolution operation. Laplacian smoothing graph convolution enforces the features of nodes to be similar to those of their neighbors, while Laplacian sharpening graph convolution pushes the features of nodes away from those of their neighbors [17]. As shown in Fig. 1(c), by considering each association p_i as a node in a graph and then separately computing compatible edges and colliding edges, both the compatible and the collision interactions among these associations are naturally encoded by graph convolution for the purpose of affinity estimation.

Different from previous GNN based models which take detections as nodes and potential associations as edges of a graph, our proposed method takes the potential associations as nodes and potential interactions among potential associations as edges, i.e., our method takes the 'edges' of previous GNN based models as 'nodes' of our graph, and explicitly encode the interaction information among detection-detection pairs into the edges of our graph, making our model better utilize the interaction information among detection-detection pairs, such as compatible and mutually exclusive interaction information. Besides, the edges of previous GNN based models are homogeneous, meaning that they only consider one kind of interaction between detections. However, the edges of our proposed graph representation model have two different types, and therefore we consider both compatible interaction and mutually exclusive interaction between associations through Laplacian smoothing operation and Laplacian sharpening operation, respectively.

Specifically, in this paper we design a novel Deep Graph Convolutional Affinity Network (DGCAN) for multiple-pedestrian tracking. A DGCAN jointly learns the pedestrian representation and the affinity in end-to-end fashion. It utilizes the Laplacian smoothing graph convolution with compatible edges to account for compatible interactions among associations of detections from two different frames and applies the Laplacian sharpening graph convolution to collision edges to account for mutually exclusive interactions. Both appearance features and motion information is encoded in the constructed graph to encourage discriminative representation of detections and robust affinity estimation of associations. Based on this efficient affinity computation network, we associate pedestrians in the current video frame to the pedestrians in multiple previous frames to generate reliable trajectories using the Hungarian method. The proposed online tracking method achieves state-of-the-art performance on the popular 2D MOT15, MOT16, and MOT17 challenge datasets.

The main contributions of this paper are as follows:

1) We identify a weakness of conventional tracking-by-detection methods that they fail to sufficiently model the interactions between detection-detection pairs, and propose a GCN-based scheme of encoding the compatible and mutually exclusive interactions between associations for affinity learning.
2) We formulate a joint learning process of the pedestrian representation and the affinity based on appearance and motion. The learning process is performed in an end-to-end manner that can alleviate the limitation of manual interaction definitions.
3) Extensive experimental results on benchmarks demonstrate the superior performance of the proposed tracker compared to state-of-the-art methods.

The rest of the paper is organized as follows: First, related work is discussed in more detail. Second, the proposed method and major contributions are explained. Third, implementation details, comparisons with state-of-the-art trackers, and ablation results are presented and discussed. Lastly, we conclude the paper by summarizing our contributions and describing future research directions.

II. Related Work

Our work is related to traditional methods that use interactions among pairs of detections and graph neural networks (GNNs), which we will review in turn.

Traditional methods that use interactions among pairs of detections: Yang *et al.* [18] regarded interactive tracklets as pairwise terms of a Conditional Random Field (CRF) model and redefined the data association cost. Shi *et al.* [15] combined individual temporal energy (the similarity score) of association hypotheses (i.e., detection-detection pairs) and spatial interaction energy between two association hypotheses into a unified optimization framework. Lan *et al.* [16] incorporated two kinds of pairwise interactions into a quadratic pseudo 0-1 optimization framework, in which the collision interactions distinguished the closely positioned similar-looking objects, and the overlapping interactions (heavily overlapping tracks that originate from the same objects) suppressed the overlapped trackers that were assumed to originate from the same object. Lan *et al.* [7] introduced two kinds of edges to explicitly model the interactions between tracklets: "close edges" imposing physical constraints between two temporally overlapping tracklets and "distant edges" accounting for higher-order motion and appearance consistency between two temporally isolated tracklets. Although these methods successfully utilize the interactions between pedestrian associations, we found that manually defined interactions limit the performance and using combination-optimizing methods, such as quadratic pseudo-Boolean optimization, are computationally expensive.

Graph Neural Networks for Multi-Pedestrian Tracking: Recently, GNNs have been introduced for multi-pedestrian tracking in order to incorporate object interactions. Ma *et al.* [19] take individual detections as nodes, and use a GNN to update node features. The updated node features are then used to compute an adjacency matrix based on their cosine similarity for data association. MPNTracker [9] exploits the classical network flow formulation, and model the tracking problem with an undirected graph where each node represents a unique detection. They cast the tracking problem into a binary classification problem over edges, and perform learning directly with a message passing network (MPN) to account for global interactions among detections. Li *et al.* [20] propose to use two separate GNNs, one for learning appearance features, and the other for learning motion features. In their graph networks, they also introduce a global variable to capture the global relationship among all nodes and edges. Liu *et al.* [21] utilized the idea of graph representation and leveraged the relations among objects to improve robustness of the similarity model. They regard each object as an anchor and build a directed local graph, taking both the feature of individual objects and the relations among objects into account, and design a graph-matching module for the proposed graph representation to alleviate the impact of unreliable relations among objects. GCNNMatch [22] models each detection as a node and feasible connections between detections from previous frame and new detections at the current frame as edges of a graph. Then it uses a graph convolutional neural network to update node features, and utilizes Sinkhorn normalization to enforce bipartite matching constraints. Different from all the aforementioned GNN based models which take detections as nodes and potential associations as edges, our proposed method takes the potential associations as nodes and potential interactions among potential associations as edges to explicitly encode the interaction information among detection-detection pairs.

III. Method

Multiple-pedestrian tracking using the tracking-by-detection paradigm is a challenging problem that involves feature extraction, affinity estimation, and adaptation to changes in the number of pedestrians. In this work, we jointly model pedestrian representation and affinity estimation across two different frames for online tracking. In conventional approaches, different detection-detection pairs are evaluated individually, i.e., the affinity estimation between a pair of detections does not influence other pairs. However, the interaction information between detection-detection pairs is valuable for refining the affinity estimation for each detection-detection pair. To implement this idea, we developed a Deep Graph Convolutional network (GCN)-based Affinity Network (DGCAN), which models the interaction of associations to improve representation learning and affinity estimation. As illustrated in Fig. 2, a DGCAN takes two images I_t, I_{t+n} that are n frames apart and their corresponding detection sets $BB_t = (d_t^1, d_t^2 \cdots d_t^p), BB_{t+n} = (d_{t+n}^1, d_{t+n}^2 \cdots d_{t+n}^q)$ as input, and creates a graph with each node representing a detection-detection pair $(d_t^i, d_{t+n}^j), i \in [1,p], j \in [1,q]$. The DGCAN then computes the similarity score $s(d_t^i, d_{t+n}^j)$ for each detection-detection pair by Laplacian smoothing graph convolution using compatible edges (green edges in Fig. 2, where we only show compatible edges that link to node (d_t^1, d_{t+n}^2), and Laplacian sharpening graph convolution using colliding edges (a subset marked red in Fig. 2). The proposed tracker uses the estimated affinities computed by the DGCAN to generate trajectories by associating pedestrians in the current frame to those in previous frames. While the proposed model is deployed to track pedestrians online, we do not strictly limit the considered frames to be consecutive during training. Instead, we allow them to be n time-stamps apart to increase the robustness of the proposed network. Here, n is a random number between 1 and N_V, where N_V is the maximum allowed distance between two input frames.

A. Graph Convolutional Network Operations

Basic notation. Consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes, a set of edges $\mathcal{E}$, an adjacency matrix $A \in \mathbb{R}^{N\times N}$, and a degree matrix $D_{ii} = \sum_j A_{ij}$. The off-diagonal elements of D are zero. The symmetric graph Laplacian L_{sys} and random walk graph Laplacian L_{rw} are defined by $L_{\text{sys}} = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}}$ and $L_{\text{rw}} = I_N - D^{-1}A$, respectively, where $I_N \in \mathbb{R}^{N\times N}$ denotes an identity matrix. Both L_{sys} and L_{rw} are positive semi-definite matrices.

Propagation rule of GCN. The layer-wise propagation rule of a typical multi-layer GCN is:

$$H^{(l+1)} = \sigma\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} H^{(l)} W^{(l)}\right). \tag{1}$$

Here, $\tilde{A} = A + I_N$ is the adjacency matrix of the undirected graph $\mathcal{G}$ with added self-loops, $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, $W^{(l)}$ is a layer-specific trainable weight matrix, $\sigma(\cdot)$ denotes an activation function like $\mathrm{ReLU}(\cdot) = \max(0,\cdot)$, $H^{(l)} \in \mathbb{R}^{N\times d}$ is the activation matrix in the l^{th} layer, and $H^{(0)}$ is the feature matrix of the input nodes.

Laplacian smoothing. Li *et al.* [23] showed that the graph convolution operation in GCNs is actually a special form of Laplacian smoothing which computes a new representation of a node as a weighted local average of the node's and its neighbors' values, i.e.:

$$\begin{aligned} H^{(l+1)} &= (1-\gamma)H^{(l)} + \gamma \tilde{D}^{-1}\tilde{A}H^{(l)} \\ &= H^{(l)} - \gamma(I_N - \tilde{D}^{-1}\tilde{A})H^{(l)} \\ &= H^{(l)} - \gamma \tilde{L}_{\text{rw}} H^{(l)}, \end{aligned} \tag{2}$$

where γ $(0 < \gamma \leq 1)$ is a regularization parameter that balances the importance between the node and its neighbors. If we set $\gamma = 1$ and replace $\tilde{L}_{\text{rw}}$ with $\tilde{L}_{\text{sys}}$, Eq. 2 becomes $H^{(l+1)} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}$ which is identical to the graph convolution in Eq. 1. Based on this interpretation, Li *et al.* [23] explain the reason why GCN works well for semi-supervised node classification: the smoothing operation of GCN enforces the features of nodes in the same cluster to be similar, thus greatly easing the classification task. Inspired by this insight, we use GCNs to ease the classification of compatible nodes (i.e., detection-detection pairs) by designing compatible edges that model compatible interactions.

Laplacian sharpening. If a deep GCN is constructed by many Laplacian smoothing layers, the output features may be excessively smoothed, resulting in indistinguishable nodes [23], which is undesirable. This over-smoothing issue will make the features of vertices indistinguishable. To alleviate the undesirable over-smoothing issue, we propose to incorporate the Laplacian sharpening based graph convolution [17]. Laplacian sharpening, which is the counterpart of Laplacian

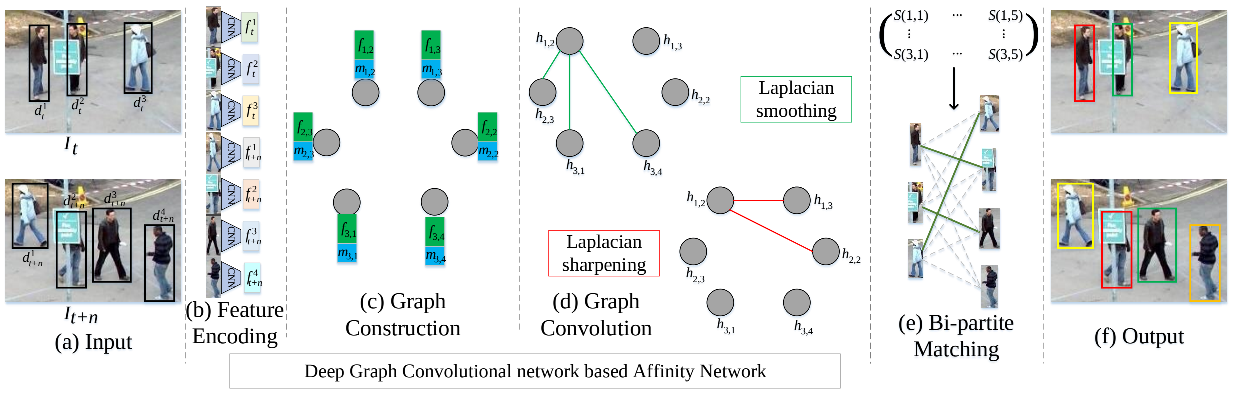


Fig. 2. The pipeline of the proposed tracker. The central part is a deep graph convolutional network-based affinity network (b-d) which explores the relationship information between detection-detection pairs. A pair of n-frame-apart images I_t, I_{t+n} are fed to the network along with the sets of detection responses in these images. All detections are processed by the same CNN for feature encoding. For graph construction, we take each detection in the first set of detections and its top K (here $K = 2$) most similar detections in the second set of detections as detection-detection pairs, and take these detection-detection pairs as nodes. Then, the Laplacian smoothing graph convolution with compatible edges and Laplacian sharpening graph convolution with colliding edges are responsible for affinity estimation. We augment the similarity matrix with an extra column to account for pedestrians leaving the scene. Finally, bipartite matching is performed based on the augmented matching probability matrix to obtain tracking results.

smoothing, encourages the output features of each node to be distant from the features of centroid of its neighbors. This property of Laplacian sharpening is exactly desired to deal with mutually exclusive nodes of our graph. Thus. We incorporate mutual exclusion constraints by applying Laplacian sharpening to colliding edges like so:

$$\begin{aligned} H^{(l+1)} &= (1+\gamma)H^{(l)} - \gamma D^{-1}AH^{(l)} \\ &= H^{(l)} + \gamma(I_N - D^{-1}A)H^{(l)} \\ &= H^{(l)} + \gamma L_{\text{rw}} H^{(l)}. \end{aligned} \tag{3}$$

If we set $\gamma = 1$ and replace L_{rw} with L_{sys}, Laplacian sharpening is expressed as

$$H^{(l+1)} = (2I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}})H^{(l)}. \tag{4}$$

Since the spectral radius of $2I_N - D^{-\frac{1}{2}}AD^{-\frac{1}{2}}$ is 3 [17], repeated application of this operator could result in numerical instability. Following [17], we employ a numerically stable form of Laplacian sharpening with spectral radius of 1, which is

$$H^{(l+1)} = \sigma\left(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \tag{5}$$

where $\hat{A} = 2I_N - A$ and $\hat{D} = 2I_N + D$.

B. Feature Encoding

We employ the powerful "feature extraction" sub-network of DAN [14] to extract a dim_1-dimensional appearance feature descriptor for each detection. The feature extractor passes a pair of video frames and detection centers through two streams of convolutional layers, and concatenates features at different levels of abstraction to generate the final feature description. In the following, we denote $F_t = (f_t^1, f_t^2 \cdots f_t^p) \in \mathbb{R}^{dim_1 \times p}$ and $F_{t+n} = (f_{t+n}^1, f_{t+n}^2 \cdots f_{t+n}^q) \in \mathbb{R}^{dim_1 \times q}$ as the collections of appearance features for two sets of detections, respectively.

C. Graph Construction

In this paper, we take advantage of GCNs to utilize the interaction information among detection-detection pairs from two different frames and the relative spatial position information between detections in the same frame. We define nodes and edges of the weighted graph as follows.

Node construction. Node set $\mathcal{V}$ consists of nodes $v_{i,j}$ representing detection-detection pair (d_t^i, d_{t+n}^j), with $d_t^i \in BB_t$ and $d_{t+n}^j \in BB_{t+n}$. In order to reduce the computation cost, we do not take every detection-detection pair $\left(d_t^i, d_{t+n}^j\right)$ as a potential association. Instead, for each detection $d_t^i \in BB_t$, we only select its top K $(K \ll q)$ most similar detections (here we use the cosine distance of normalized appearance features of two detections to estimate their similarity) in frame $t+n$ to construct potential associations. The number of nodes N is therefore $p \times K$. The appearance feature representation $f_{i,j}$ for node $v_{i,j}$ is defined as the absolute difference between the appearance features of d_t^i and d_{t+n}^j, i.e., $f_{i,j} = |f_t^i - f_{t+n}^j|$. We also take the motion information of the detection-detection pair (d_t^i, d_{t+n}^j) into account. Given the bounding box $[x_t^i, y_t^i, w_t^i, h_t^i]$ of detection d_t^i and $[x_{t+n}^j, y_{t+n}^j, w_{t+n}^j, h_{t+n}^j]$ of d_{t+n}^j, we project the 4-dimensional vector

$$[\frac{|x_t^i - x_{t+n}^j|}{w_{t+n}^j}, \frac{|y_t^i - y_{t+n}^j|}{h_{t+n}^j}, \frac{|w_t^i - w_{t+n}^j|}{w_{t+n}^j}, \frac{|h_t^i - h_{t+n}^j|}{h_{t+n}^j}]$$

into a dim_2-dimensional vector $m_{i,j}$ by a single-layer fully connected network. We concatenate batch normalized $f_{i,j}$ and $m_{i,j}$ to obtain the final feature representation $g_{i,j}$ of node $v_{i,j}$:

$$g_{i,j} = [\text{BN}(f_{i,j}) \mid\mid \text{BN}(m_{i,j})] \in \mathbb{R}^{dim_1 + dim_2}. \tag{6}$$

Edge construction. The adjacency matrix element $A_{(i_1,j_1),(i_2,j_2)}$ of the weighted graph that quantifies the interaction between v_{i_1,j_1} and v_{i_2,j_2} not only depends on the similarity between detections $d_t^{i_1}$ and $d_t^{i_2}$ in frame t, but also on the similarity between detections $d_{t+n}^{j_1}$ and $d_{t+n}^{j_2}$ in frame $t+n$. So we first learn the pairwise relationship $A_t(i_1, i_2) = \phi(f_t^{i_1}, f_t^{i_2}; \theta_t)$ between detections $d_t^{i_1}$ and $d_t^{i_2}$ with parameters θ_t, and the pairwise relationship $A_{t+n}(j_1, j_2) = \phi(f_{t+n}^{j_1}, f_{t+n}^{j_2}; \theta_{t+n})$ between detections $d_{t+n}^{j_1}$ and $d_{t+n}^{j_2}$ with parameters θ_{t+n}. The topological structure between detections in the same frame, which is crucial for affinity estimation, is incorporated in the pairwise relationship learning process.

By considering each detection set as a graph with each detection as its one node, we adopt Delaunay Triangulation, which is a widely adopted strategy to produce sparsely connected graph, to compute two initial adjacency matrices A_t', A_{t+n}' for encoding the relative spatial position information of the corresponding detection sets. Note that we do not consider self-loops, which means that all diagonal elements of A_t' and A_{t+n}' are zero. Then, we implement ϕ via a single-layer network:

$$\begin{aligned} A_t(i_1, i_2) &= \phi(f_t^{i_1}, f_t^{i_2}, A_t'; \theta_t) \\ &= \frac{A_t'(i_1,i_2)\exp\big(\sigma(\theta_t^T[f_t^{i_1}||f_t^{i_2}])\big)}{\sum_{k=1}^{p} A_t'(i_1,k)\exp\big(\sigma(\theta_t^T[f_t^{i_1}||f_t^{k}])\big)}, \end{aligned} \tag{7}$$

where $||$ denotes concatenation. A_{t+n} can be computed analogously.

Compatible edges. If $(i_1 \neq i_2)$ and $(j_1 \neq j_2)$, nodes v_{i_1,j_1} and v_{i_2,j_2} are considered compatible (*i.e.*, both nodes can be classified to 1, meaning that $d_t^{i_1}$ can be connected to $d_{t+n}^{j_1}$ and $d_t^{i_2}$ can be connected to $d_{t+n}^{j_2}$ simultaneously). Intuitively, if the detection response pair i_1 and i_2 in the n-th frame has similar relative position with detection pair j_1 and j_2 in the $(n+t)$-th frame, the nodes v_{i_1,j_1} and v_{i_2,j_2} have a larger possibility to both take 1. Thus we define the element $A_{comp}((i_1,j_1),(i_2,j_2))$ as the product of $A_t(i_1,i_2)$ and $A_{t+n}(j_1,j_2)$. The compatible edge weight is defined as follows:

$$A_{comp}((i_1,j_1),(i_2,j_2)) = \left\{ \begin{array}{c} A_t(i_1,i_2)\cdot A_{t+n}(j_1,j_2), \;\; if\ i_1 \neq i_2\ and\ j_1 \neq j_2; \\ 0, \;\; otherwise. \end{array} \right. \tag{8}$$

It can be written in the following matrix form:

$$A_{comp} = A_t \otimes A_{t+n}, \tag{9}$$

where $\otimes$ denotes the Kronecker product.

Colliding edges. If $(i_1 = i_2)$ or $(j_1 = j_2)$, nodes v_{i_1,j_1} and v_{i_2,j_2} are mutually exclusive (*i.e.* at most one node can be classified to 1, meaning that if $d_t^{i_1}$ is connected to $d_{t+n}^{j_1}$, $d_t^{i_2}$ cannot be connected to $d_{t+n}^{j_2}$, and vice versa). Our A_{coll} only works in the situations of $i_1 = i_2$ or $j_1 = j_2$. The intuition is that we heavily punish the situation that two distanced detections in one frame suddenly merge into an identical individual in another frame (large punishment value), and relax the case that two very close detections in one frame may come from the same pedestrian because of the detector failure and allow to merge (small punishment value). Thus, we define the collision edge weight as follows:

$$A_{coll}((i_1,j_1),(i_2,j_2)) = \left\{ \begin{array}{c} A_{t+n}(j_1,j_2), \;\; if\ i_1 = i_2; \\ A_t(i_1,i_2), \;\; if\ j_1 = j_2; \\ 0, \;\; otherwise. \end{array} \right. \tag{10}$$

It can be written in the following matrix form:

$$A_{coll} = I_p \otimes A_{t+n} + A_t \otimes I_q, \tag{11}$$

where I_p, I_q are identity matrices of size p and q, respectively.

D. Affinity Estimation via Graph Convolution

After graph construction, we obtain an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with one compatibility adjacency matrix A_{comp} and one collision adjacency matrix A_{coll}. The feature representation of each node $v_{i,j}$ is denoted by $g_{i,j}$, and we write $H^{(0)}$ as:

$$H^{(0)} = (g_{1,1}\cdots g_{1,q}, \cdots, g_{p,1}\cdots g_{p,q}) \in \mathbb{R}^{(dim_1+dim_2)\times(p\times q)}. \tag{12}$$

To consider compatible interactions, two Laplacian smoothing layers are stacked to process $H^{(0)}$ according to Eq. 1 with A_{comp}, resulting in $H^{(1)} \in \mathbb{R}^{dim_3\times(p\times q)}$ and $H^{(2)} \in \mathbb{R}^{dim_4\times(p\times q)}$, respectively. Then, to also consider mutually exclusive interactions, we use a Laplacian sharpening layer to process $H^{(2)}$ according to Eq. 5 with A_{coll}, producing $H^{(3)} \in \mathbb{R}^{1\times(p\times q)}$. $H^{(3)}$ is then reshaped to obtain similarity matrix $S_{init} \in \mathbb{R}^{p\times q}$.

Note that similarity matrix S_{init} does not account for pedestrians leaving the scene. To take this into account, we append an extra column to S_{init}, filled with value η, a DGCAN hyper-parameter, resulting in matrix $S_{aug} \in \mathbb{R}^{p\times(q+1)}$, i.e., we fix the score of a pedestrian leaving a scene to η. If the first q values in the i^{th} row are smaller than η, then the i^{th} tracked pedestrian is thought to have left the scene (this is achieved by applying the row-wise softmax operation described below).

E. Matching Prediction and Loss Function

In the above formulation, the i^{th} $(1 \leq i \leq p)$ row of the augmented matrix S_{aug} associates the i^{th} identity in frame t to $q+1$ identities in frame $t+n$, which includes the case in which the i^{th} identity has left the scene in frame $t+n$. To construct a matrix S that indicates the association probabilities between each pedestrian in frame t and all identities in frame $t+n$ we apply a row-wise soft-max operation to S_{aug}:

$$S(i,j) = \frac{\exp(S_{aug}(i,\ j))}{\sum_k \exp(S_{aug}(i,k))}. \tag{13}$$

In order to train DGCAN, we adopt the cross-entropy loss function. Let $L_{t,t+n} \in \{0,1\}^{p\times(q+1)}$ denote the ground truth bipartite matching solution between two detection sets in frames t and $t+n$. Similar to the augmented matrix S_{aug}, the ground truth matching matrix $L_{t,t+n}$ contains an extra column, indicating pedestrians leaving the scene between frame t and $t+n$. With this, the cross-entropy loss is defined as

$$\mathcal{L} = -\sum_{i,j} \begin{array}{l} L_{t,t+n}(i,j)\log(S(i,j)) \\ +(1-L_{t,t+n}(i,j))\log(1-S(i,j)) \end{array} \tag{14}$$

F. Confidence-Based Deep Track Association

At each time step t, we are able to efficiently compute the association probability matrix $S_{t-n,t}$ $(n \in [1, N_V])$ using DGCAN. In order to obtain a reliable track-to-detection association matrix $S_{track-det}^t$, we accumulate affinities between detections in the current frame and those in multiple previous frames. Specifically, based on multiple association probability matrices, we propose a confidence-based deep track association procedure to construct the track-to-detection association matrix $S^t_{track-det}$. The $(i,j)^{th}$ value of association matrix $S^t_{track-det}$ is a weighted sum of two association probabilities, one is between the j^{th} detection and the recent associated detection of the i^{th} track, and the other is between the j^{th} detection between the history buffer of the i^{th} track, which itself is another weighted sum of association probabilities between the j^{th} detection and associated detections in the history buffer of the i^{th} track.

The accumulated affinity $S^t_{track-det}(i,j)$ between the i^{th} identity in the current track set and the j^{th} identity in the current frame t is defined as follows:

$$\begin{aligned} S^t_{track-det}(i,j) = &\frac{c^i_{rcnt}}{\lambda_c} S_{t-n^i_{rcnt},t}(i_{rcnt}, j) \\ &+ \left(1 - \frac{c^i_{rcnt}}{\lambda_c}\right) \sum_{l=1}^{N(His^i)} \left(w^i_l \cdot S_{t-n^i_l,t}(i_l, j)\right), \end{aligned} \tag{15}$$

where n^i_{rcnt} and n^i_l denote the frame gap between the recent, l^{th} historical matched pedestrians in track i and the j^{th} new pedestrian in the current frame t, respectively. $S_{t-n^i_{rcnt},t}(i,j)$ denotes the association probability between the j^{th} new pedestrian at frame t and the recently matched pedestrian in track i, which is the ${i_{rcnt}}^{th}$ pedestrian at frame $t - n^i_{rcnt}$. $S_{t-n^i_l,t}(i_l, j)$ denotes the association probability between the j^{th} new pedestrian at frame t and the l^{th} historical matched pedestrian in track i, which is the ${i_l}^{th}$ pedestrian at frame $t - n^i_l$. The association probabilities are associated through matching confidence variables c^i_{rcnt} and w^i_l, where c^i_{rcnt} is the matching confidence of the most recent pedestrian and w^i_l indicates the normalized matching confidence of the l^{th} historical pedestrian. His^i is the history buffer that stores historically matched pedestrians for track i, and $N(His^i)$ denotes its length. We use parameter λ_c to control the effect of recent match confidence. w^i_l is defined as follows:

$$w^i_l = \frac{c^i_l}{\sum_{k=1}^{N(His^i)} c^i_k}, \tag{16}$$

where c^i_l is the matching confidence of the l^{th} historical pedestrian for track i. Through this deep track association, association probabilities between the j^{th} new pedestrian at frame t and historical pedestrians in previous frames are all considered. The matching confidence is calculated as,

$$c^i_{rcnt} = S^{t_{rcnt}}_{track-det}(i, j^*), \tag{17}$$

where j^* is the recently associated pedestrian index in the previous frame t_{rcnt} for track i. The triplet $[t_{rcnt}, j^*, c^i_{rcnt}]$ is added to the history buffer His^i for track i if the matching score c^i_{rcnt} is larger than τ_{his_score}. Each history buffer is a FIFO buffer with a maximum length of τ_{his}.

In summary, the association matrix is calculated in the form of weighted combination. The contribution of the recent pedestrian is proportional to its matching score c^i_{rcnt}. The saved historical pedestrians are associated through corresponding matching scores.

After obtaining $S^t_{track-det}$, the commonly used Hungarian algorithm is applied to find a cost-optimal bipartite matching, which enables online tracking especially in streaming video applications like autonomous vehicle driving.

IV. Experiments

This section presents the experiments to demonstrate the effectiveness of the proposed method.

A. Implementation Details

The proposed DGCAN is implemented using the PyTorch machine learning framework [24], and training is conducted on a server with one NVIDIA Tesla V100 GPU, one AMD EPYC 7601 CPU, and 256 GB of RAM. We take the "feature extractor network" of DAN [14] as our feature encoding network, where the output feature dimension dim_1 is 520. The maximum value of frame gaps for input image pairs N_V is set to 30, the hyper-parameter η stored in the appended column vector is 0.1, and the score threshold τ_{his_score} is set to 0.8. The initial parameters of the feature encoding network are the trained parameters of DAN [14], and the parameters of the motion network, the pairwise relationship learning network, the Laplacian smoothing layer, and the Laplacian sharpening layer are initialized using Xavier initialization [25]. We use the SGD optimizer to train DGCAN. The momentum and weight decay are set to 0.9 and 5e-4, respectively. We adopt an initial learning rate of 0.01 which is divided by a factor of 10 at epochs 20, 30, 40, and 45. The model is trained for 50 epochs. Since the number of detections (p, q), the number of nodes, and the dimension of adjacency matrix vary in different frames, we set the batch size in each iteration to 1. All hyper-parameters are chosen by small preliminary experiments, except for some hyper-parameters that may have great impact on performance. These hyper-parameters are chosen based on the ablation study results reported below.

B. Datasets and Evaluation Metrics

We conduct experiments on 3 popular multiple-pedestrian tracking challenge benchmarks including 2D MOT15 [26], MOT16, and MOT17 [27]. MOT17 includes a total of 7 training sequences and 7 test sequences each of which is provided with three sets of public detections, namely DPM [28], Faster R-CNN [1], and SDP [2]. The MOT16 benchmark contains the same sequences as MOT17 but only provides DPM detections. The 2D MOT15 benchmark provides ACF detections [29] for 11 training sequences and 11 test sequences.

For quantitative evaluation, we use the widely adopted CLEAR MOT Metrics [30] and trajectory-based metrics [31]: **MOTA** evaluates the accuracy in the presence of false positives (**FP**), false negatives (**FN**), and identity switches (**IDS**). **MOTP** measures the intersecting area of the tracking output and the ground truth. **IDF1** is the ratio of correctly identified detections over the average number of ground truth and computed detections, which indicates the average maximum consistent tracking rate. **MT** evaluates the mostly tracked trajectories, i.e., the ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span. **ML** evaluates the mostly lost trajectories, i.e., the ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective lifespan. **IDS** counts the total number of identity switches. **FRAG** counts the total number of times that a trajectory is interrupted during tracking. Among these metrics, **MOTA** and **IDF1** are two comprehensive and important metrics which are widely recognized in this field.

C. Comparison with the State-of-the-Art

We compare the proposed DGCAN with existing approaches that use the public detections provided by the benchmarks as ours. In this section, we evaluate our method in two settings, DGCAN, which utilizes the original detections, and DGCAN_C, which uses the bounding box regression of CenterTrack [32], all acting on the public detections provided by the MOT benchmark datasets.

MOT17 results. In Table I, we summarize the results of our method and the published state-of-the-art techniques on MOT17. The table contains both online and offline results. As can be seen, the proposed DGCAN method is able to outperform most of the existing offline methods and all online methods that use the original public detections in terms of MOTA and IDF1. The proposed DGCAN_C that use CenterTrack [32] to refine the public detections outperforms all the previous methods and sets a new state-of-the-art in terms of MOTA. Among the compared methods that use the original public detections, LSST [8] uses a single object tracker to predict bounding boxes to reduce false negatives. STRN_MOT17 [39] combines appearance, location, and topology cues in a unified spatial-temporal relation network. Our DGCAN uses the same appearance feature extractor as DAN [14], but a different affinity estimation module. Both STRN_MOT17 and DAN are similar to our DGCAN method, but in comparison, our method achieves better MOTA scores (plus 3.5 and 2.0, respectively).

Some methods uses a high-performance detector “Faster-RCNN” [1] to refine the provided public detections, such as Tracktor++ [41], GSM_Tracktor [21], GCNNMatch [22], and MPNTrack [9]. Tracktor++ [41] predicts detections in the next frame by taking the refined detections in the current frame as bounding box proposals. Some methods utilize “CenterTrack” [32] to refine the provided public detections, including CenterTrack [32], UnsupTrack [42], and our DGCAN_C. CenterTrack [32] applied a detection model to localize objects in the current frame and predict their associations with the previous frame. UnsupTrack [42] incorporates an unsupervised re-identification network with CenterTrack [32]. Among all the methods that use refined detections, our proposed DGCAN_C achieves 64.4 MOTA, the new state-of-the-art MOTA score. DGCAN_C outperforms the previous best graph neural network based method (GCNNMatch) by 7.4 MOTA score, demonstrating the effectiveness of taking associations as nodes of a graph and explicitly incorporating the interaction information among these associations. The proposed DGCAN_C outperforms the previous best online method (UnsupTrack) that also uses “CenterTrack” for refinement by 2.7 MOTA score, which further validates the effectiveness of the proposed method.

Note that our proposed method under-performed in certain metrics, e.g. FP, IDS, FRAG. Of all the evaluation metrics, FP and FN are opposite pair that there exists a relation of “as one falls, another rises”. Specifically, in a similar class of multiple-pedestrian tracking methods, if a method has a lower FN, it tends to have a higher FP, because the method tends to judge more detection responses as real targets, in which case even false targets may be misjudged as real targets, resulting in higher FP. When a tracker has a lower FN and a higher FP, the tracker needs to associate more fake pedestrians, and the tracker is more easily associated with these spurious pedestrians, resulting in higher IDS and FRAG. In intelligent transportation applications, we believe that in order to prevent car accidents due to missing targets such as pedestrians, the real target should not be missed even if the tracker generates more fake targets. That is, the tracker should tend to choose to have a lower FN and the resulting relatively high FP, rather than having a lower FP and the resulting relatively high FN.

In Fig. 3 we show the qualitative results of our proposed DGCAN on the MOT17 challenge test sequence using the provided SDP detections (three representative images per sequence). Consistency of the estimated trajectories is indicated by bounding boxes with the same color and the same ID number over time. It can be observed that our tracker yields visually plausible results even on challenging scenarios with many pedestrians or occlusions.

MOT16 and 2D MOT15 results. The MOT16 results in Table II reveal that our proposed tracker outperforms all the previous trackers by a large margin and sets a new state-of-the-art. The proposed DGCAN_C achieves a 65.4 MOTA score on the MOT16, outperforming the previous best method by 3.0 MOTA score. Similar results on 2D MOT15 are represented in Table III. The proposed DGCAN_C achieves a 49.2 MOTA score on the 2D MOT15, outperforming the previous state-of-the-art online method Tracktor++ by 5.1 MOTA score.

D. Ablation Studies

In this section we validate our proposed method DGCAN by conducting a series of ablation studies on the MOT17-SDP training dataset (we also conducted experiments on the MOT17-DPM and MOT17-FRCNN datasets, and achieved similar results). First, we determined the value of some important hyper-parameters:

The number of potential associations K: To reduce computation cost, we do not take every detection-detection pair each from a separate frame as a potential association. Instead, we only select the most similar K detection responses in the second frame with a detection in the first frame to construct potential associations. Then we take these potential associations as vertices. The evaluation metrics for different K is shown in Table IV, where we show the results of taking all detection-detection pairs as vertices in the last row. We show the computation time of graph convolution layers in the fifth column. In order to validate the usability of taking associations as vertices, we compare our method with graph neural network based methods which take detection responses as vertices. Specifically, we present ratio of the number of vertices between taking associations as vertices and taking detection responses as vertices in the sixth column of Table IV,

TABLE I
MOT17 CHALLENGE TEST SET RESULTS. VALUES IN BOLD HIGHLIGHT THE BEST RESULTS OF *online methods*. THE SYMBOL ↑ INDICATES THAT HIGHER VALUES ARE BETTER, AND ↓ IMPLIES LOWER VALUES ARE FAVORED. METHODS WITH $^+$ USE "FASTERRCNN" [1] TO REFINE THE PROVIDED PUBLIC DETECTIONS. METHODS WITH * USE "CENTERTRACK" [32] TO REFINE THE PROVIDED PUBLIC DETECTIONS.

Tracker	Type	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓	FRAG↓
MHT_DAM [33]	offline	50.7	47.2	77.5	20.8	36.9	22875	252889	2314	2865
jCC [34]	offline	51.2	54.5	75.9	20.9	37.0	25937	247822	1802	2984
FWT [35]	offline	51.3	47.6	77.0	21.4	35.3	24101	247921	2648	4279
eTC17 [36]	offline	51.9	58.1	76.3	23.1	35.5	36164	232783	2288	3071
JBNOT [37]	offline	52.6	50.8	77.1	19.7	35.8	31572	232659	3050	3792
LSST17 [8]	offline	54.7	62.3	75.9	20.4	40.1	26091	228434	1243	3726
MPNTrack$^+$ [9]	offline	58.8	61.7	78.6	28.8	33.5	17413	213594	1185	2265
MOTDT17 [38]	online	50.9	52.7	76.6	17.5	35.7	24069	250768	2474	5317
STRN_MOT17 [39]	online	50.9	56.0	75.6	18.9	33.8	25295	249365	2397	9363
FAMNet [40]	online	52.0	48.7	76.5	19.1	33.4	14138	253616	3072	5318
DAN [14]	online	52.4	49.5	76.9	21.4	30.7	25423	234592	8431	14797
LSST17O [8]	online	52.7	57.9	76.2	20.4	40.1	22512	241936	2167	7443
DGCAN (ours)	online	54.4	54.1	77.4	17.8	37.4	12655	241868	2660	3991
Tracktor++$^+$ [41]	online	56.3	55.1	78.8	21.1	35.3	**8866**	235449	1987	3763
GSM_Tracktor$^+$ [21]	online	56.4	57.8	77.9	22.2	34.5	14379	230174	**1485**	**2763**
GCNNMatch$^+$ [22]	online	57.0	56.1	78.7	23.3	34.6	12283	228242	1957	2798
CenterTrack* [32]	online	61.5	**59.6**	**78.9**	26.4	31.9	14076	200672	2583	4965
UnsupTrack* [42]	online	61.7	58.1	78.3	27.2	32.4	16872	197632	1864	4213
DGCAN_C* (ours)	online	**64.4**	58.4	78.1	**28.3**	**29.6**	15071	**179148**	6500	7199

TABLE II
MOT16 CHALLENGE TEST SET RESULTS. VALUES IN BOLD HIGHLIGHT THE BEST RESULTS OF *online methods*. THE SYMBOL ↑ INDICATES THAT HIGHER VALUES ARE BETTER, AND ↓ IMPLIES LOWER VALUES ARE FAVORED. METHODS WITH $^+$ USE "FASTERRCNN" [1] TO REFINE THE PROVIDED PUBLIC DETECTIONS. METHODS WITH * USE "CENTERTRACK" [32] TO REFINE THE PROVIDED PUBLIC DETECTIONS.

Tracker	Type	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓	FRAG↓
NOMT [43]	offline	46.4	53.3	76.6	18.3	41.4	9753	87565	359	504
FWT [35]	offline	47.8	44.3	75.5	19.1	38.2	8886	85487	852	1534
eTC [44]	offline	49.2	56.1	75.5	17.3	40.3	8400	83702	606	882
HCC [45]	offline	49.3	50.7	79.0	17.8	39.9	5333	86795	391	535
NOTA [46]	offline	49.8	55.3	74.5	17.9	37.7	7248	83614	614	1372
MPNTrack$^+$ [9]	offline	58.6	61.7	78.9	27.3	34.0	4949	70252	354	684
MOTDT16 [38]	online	47.6	50.9	74.8	15.2	38.3	9253	85431	792	1858
STRN_MOT16 [39]	online	48.5	53.9	73.7	17.0	34.9	9038	84178	747	2919
LSST16O [8]	online	49.2	56.5	74.0	13.4	41.4	7187	84875	606	2497
DGCAN (ours)	online	53.9	53.7	77.0	16.7	39.4	**2514**	80892	631	894
Tracktor++$^+$ [41]	online	54.4	52.5	78.2	19.0	36.9	3280	79149	682	1068
GCNNMatch$^+$ [22]	online	56.9	55.9	**79.1**	22.3	35.3	3235	74784	564	818
GSM_Tracktor$^+$ [21]	online	57.0	58.2	**79.1**	22.0	34.5	4332	73573	**550**	**772**
UnsupTrack* [42]	online	62.4	58.5	78.3	27.0	31.9	5909	61981	588	1361
DGCAN_C* (ours)	online	**65.4**	**59.4**	78.0	**28.3**	**29.6**	5007	**55940**	2069	2309

TABLE III
2D MOT15 CHALLENGE TEST SET RESULTS. VALUES IN BOLD HIGHLIGHT THE BEST RESULTS OF *online methods*. THE SYMBOL ↑ INDICATES THAT HIGHER VALUES ARE BETTER, AND ↓ IMPLIES LOWER VALUES ARE FAVORED. METHODS WITH $^+$ USE "FASTERRCNN" [1] TO REFINE THE PROVIDED PUBLIC DETECTIONS. METHODS WITH * USE "CENTERTRACK" [32] TO REFINE THE PROVIDED PUBLIC DETECTIONS.

Tracker	Type	MOTA↑	IDF1↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓	FRAG↓
NOMT [43]	offline	33.7	44.6	71.9	12.2	44.0	7762	32547	442	823
QuadMOT [47]	offline	33.8	40.4	73.4	12.9	36.9	7898	32061	703	1430
JointMC [34]	offline	35.6	45.1	71.9	23.2	39.3	10580	28508	457	969
CEM [48]	offline	48.3	56.5	74.8	32.2	24.3	9640	21629	504	1023
MPNTrack$^+$ [9]	offline	51.5	58.6	76.0	31.2	25.9	7620	21780	375	872
STRN [39]	online	38.1	46.6	72.1	11.5	33.4	5451	31571	1033	2665
DAN [14]	online	35.8	39.6	72.4	7.8	39.0	**4065**	33669	1728	1312
DGCAN (ours)	online	38.8	41.9	74.7	11.1	40.5	5293	31404	**914**	**1050**
Tracktor++$^+$ [41]	online	44.1	**46.7**	**75.0**	18.0	26.2	6477	26577	1318	1702
DGCAN_C* (ours)	online	**49.2**	45.7	74.7	**33.1**	**20.2**	7747	**21124**	2319	1922

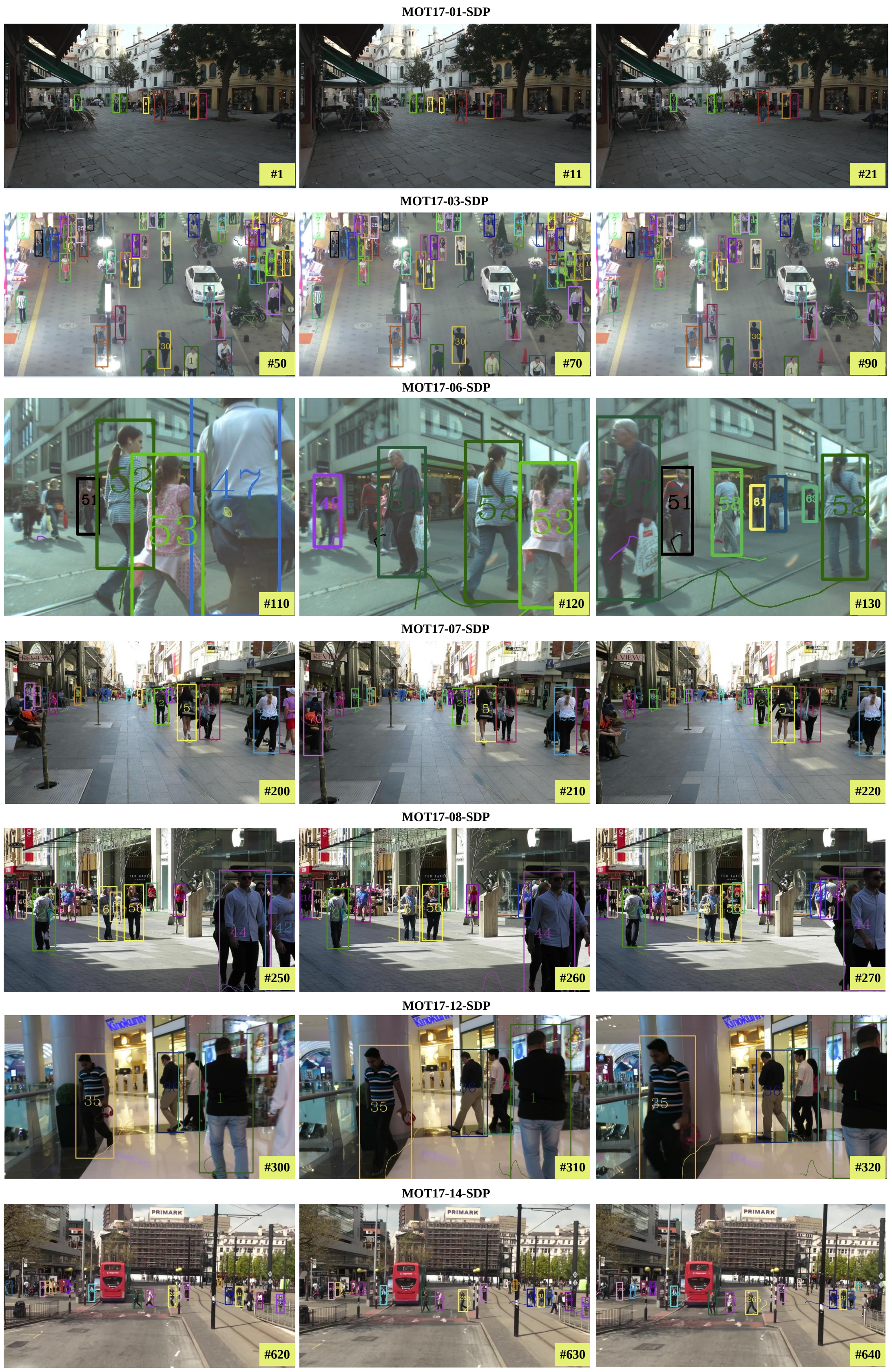


Fig. 3. Qualitative tracking results on different test sequences of the MOT17 dataset.

TABLE IV
EVALUATION METRICS OF DIFFERENT K ON MOT17-SDP TRAINING SEQUENCES.

K	IDF1↑	MOTA↑	IDS↓	GCN time(s)	vertex ratio	comp_edge ratio	coll_edge ratio	edge ratio
1	63.5	64.9	502	241.15	0.5	0.945	0.063	1.008
2	66.0	64.9	336	243.39	1.0	3.69	0.338	4.028
3	**66.5**	**65.0**	328	245.45	1.5	8.25	0.81	9.06
4	66.3	64.9	**321**	248.4	1.99	14.6	1.46	16.06
5	66.4	64.9	331	251.69	2.47	22.8	2.29	25.09
6	66.1	64.9	336	271.23	2.93	32.7	3.26	35.96
all	66.1	64.9	326	355.99	10.97	820	55.1	875.1

and ratio of the number of compatible edges, ratio of the number of colliding edges, ratio of the number of total edges in the seventh, eighth and ninth columns, respectively. From Table IV, the proposed top K scheme significantly reduces the number of vertices and edges, and therefore requires much less memory and computation resources. The best tracking performance was obtained when K is 3, thus we set $K = 3$ for the following experiments. Thus, instead of enlarging the dimensions of nodes and edges from N to N^2, the filter strategy only enlarges the dimensions of nodes and edges from N to $3N$.

Motion description dimension: The dimension of initial appearance features of nodes is 520. To better fuse motion and appearance, we select the dimension of motion dim_2 from $\{32, 64, 128, 256, 512\}$. From Fig. 4, the best performance were obtained for $dim_2 = 256$, we chose this value for all subsequent experiments.

Dimension of hidden layers of GCN: The dimension of initial features of nodes is $dim_1 + dim_2 = 776$, thus we select the dimension of hidden layers of GCN dim_3, dim_4 from $\{128, 256, 512, 1024, 2048\}$. From Fig. 5 and Fig. 6, we set $dim_3 = 1024$ and $dim_3 = 512$.

History buffer length: The maximum length of the history buffer τ_{his} in an important hyper-parameters because it is directly related to the depth of confidence-based track association. We performed experiments to find a good value out of a fixed set of choices. We varied τ_{his} from 1 to 14 with interval length 1 and compared the IDF1 and MOTA scores. The obtained results are shown in Fig. 7. Because the best performance were obtained for $\tau_{his} = 6$, we chose this value for all subsequent experiments.

Control parameter variation: Parameter λ_c controls the effect of the recent pedestrian confidence score. We measured IDF1 and MOTA scores for various λ_c settings as depicted in Fig. 8 and selected $\lambda_c = 3$ for the following experiments.

Effect of compatible and conflicting interactions: We included the conventional online tracking model (Fig. 1(a)) as the baseline in the experiments reported in Fig. 9(A). It utilizes 2 linear layers to estimate the affinity of each detection pair. To separately demonstrate the influence of compatible and mutually exclusive interactions, we designed two GCN models with 2 Laplacian smoothing layers (model (B) in Fig. 9) and 2 Laplacian sharpening layers (model (C) in Fig. 9), respectively. Finally, we also designed a GCN model with 1 Laplacian smoothing layer and 1 Laplacian sharpening layer (model (D) in Fig. 9) to incorporate both kinds of interactions among pedestrians. The reported results indicated that incorporating relationship information and topological structure information by GCNs can improve the performance of multiple-pedestrian tracking. Specifically, compared with model (A), the MOTA score is increased by 4.9% (B), 4.9% (C), and 5% (D); MOTP values are improved by 2.3% (B), 2.3% (C), and 2.3% (D); IDF1 is increased by 0.2% (B), 1.1% (C), and 2.8% (D); and IDS is decreased by 51.9% (B), 58% (C), and 61.3% (D). These comparisons demonstrate that either Laplacian smoothing or Laplacian sharpening works well with the developed graph data structure, while combining them achieves the best scores.

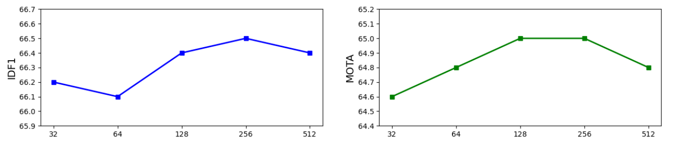


Fig. 4. Tracking performance for various motion dimensions dim_2.

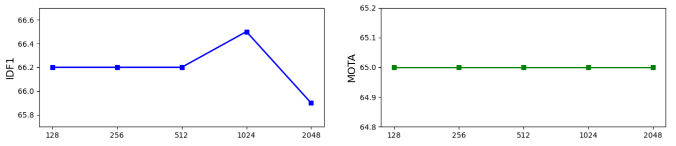


Fig. 5. Tracking performance for various dimensions dim_3 of the first hidden GCN layer.

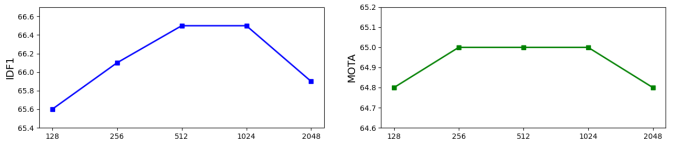


Fig. 6. Tracking performance for various dimensions dim_4 of the second hidden GCN layer.

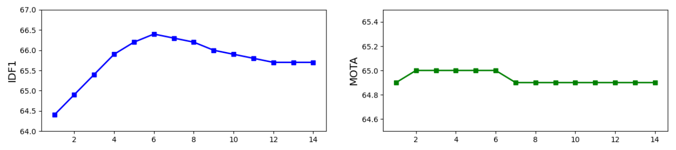


Fig. 7. Tracking performance for various history buffer lengths τ_{his}.

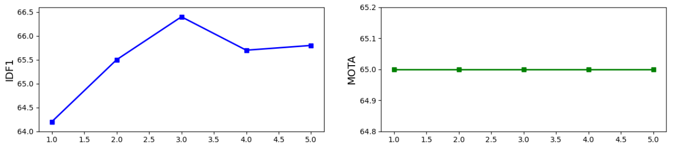


Fig. 8. Tracking performance for various λ_c values.

Effect of deep track association: To evaluate the effect of the confidence-based deep track association, we compare our method with a baseline without the history buffer, i.e., computing the affinity between a track and a detection only based on the recently matched detection of the track. Table V shows the performance of DGCAN with and without history buffer. We observe that the proposed deep track association outperforms the baseline on all evaluation metrics.

GCN Architecture: As the GCN architecture may influence the performance of DGCAN, we obtained performance values for different model settings that are shown in Table VI. The x-sm-y-sh model refers to a stack of x Laplacian smoothing

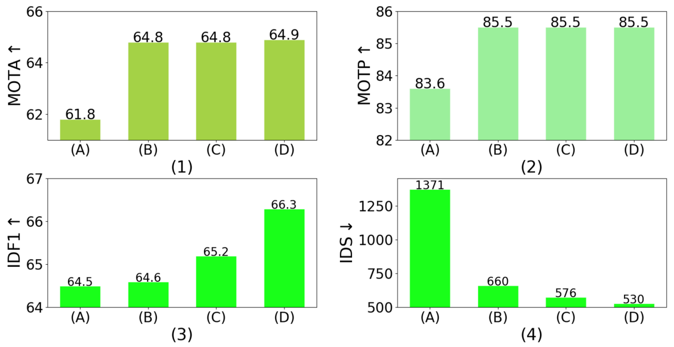


Fig. 9. Ablation studies on MOT17-SDP in terms of MOTA, MOTP, IDF1, and IDS, where ↑ means higher values are better, ↓ means lower values are favored. The selected models include (A) 2 linear layers, (B) 2 Laplacian smoothing layers, (C) 2 Laplacian sharpening layers, and (D) 1 Laplacian smoothing layer and 1 Laplacian sharpening layer.

TABLE V
EVALUATION METRICS WITH AND WITHOUT HISTORY BUFFER FOR DEEP ASSOCIATION ON MOT17-SDP TRAINING SEQUENCES.

Variant	MOTA↑	MOTP↑	IDF1↑	IDS↓
without history	64.8	**85.5**	63.8	526
with history	**65.0**	**85.5**	**66.5**	**501**

layers and y Laplacian sharpening layers. From the last row of Table VI, we observe that the architecture of two Laplacian smoothing layers and one Laplacian sharpening layer is the best setting in terms of all the metrics in our application.

In order to explore the influence of the ordering of the Laplacian smoothing layer and the Laplacian sharpening layer, we compared the performance of different GCN architectures by changing the order of the Laplacian smoothing and sharpening layers. From Table VII, the GCN architecture with the ordering of Laplacian smoothing, smoothing and sharpening demonstrated the best performance, implying that employing the compatible interactions first, and then applying the mutually exclusive interactions among associations works best.

E. Time Analysis

In Table VIII, we show the running time of our tracker on different test sets given the detections. Note that the running time varies between these sets of detections. This is mainly due to the fact that different detections feature different numbers of detection bounding boxes, which leads to the varying time for constructing the graph, the computation of the similarity matrix, and the process of deep data association. The quality of detection bounding boxes also affects the running time. For example, the number of original DPM detection bounding boxes is much larger than that of original SDP detection. However, many DPM detection bounding boxes are false positives, and these false positives can be removed by some heuristics, such as dropping detections whose confidence scores are lower than a threshold, or dropping tracked targets that do not associate detections for a period of time. The valid number of DPM detection bounding boxes is smaller than the valid number of SDP detections, and thus the processing speed of DPM detection (6.5 fps) is bigger than that of SDP detection (4.6 fps).

TABLE VI
EVALUATION METRICS OF DIFFERENT GCN ARCHITECTURES ON MOT17-SDP TRAINING SEQUENCES.

Variant	MOTA↑	MOTP↑	IDF1↑	IDS↓
1-sm-1-sh	64.9	**85.5**	66.3	530
1-sm-2-sh	64.9	**85.5**	65.8	550
2-sm-1-sh	**65.0**	**85.5**	**66.5**	**501**
2-sm-2-sh	**65.0**	**85.5**	66.0	512

TABLE VII
EVALUATION METRICS OF DIFFERENT ORDERING OF LAPLACIAN SMOOTHING AND SHARPENING LAYERS ON MOT17-SDP TRAINING SEQUENCES.

Variant	MOTA↑	MOTP↑	IDF1↑	IDS↓
sh-sm-sm	64.5	**85.5**	65.1	562
sm-sh-sm	64.9	**85.5**	66.4	520
sm-sm-sh	**65.0**	**85.5**	**66.5**	**501**

TABLE VIII
RUNNING TIME (SECONDS)AND SPEED (FRAMES PER SECOND) OF OUR TRACKER ON DIFFERENT TEST SETS OF DETECTIONS, WHERE $t1$ DENOTES FEATURE EXTRACTION TIME, $t2$ DENOTES GCN BASED TRACK-DETECTION AFFINITY ESTIMATION TIME, $t3$ DENOTES HUNGARIAN ALGORITHM TIME, $t4$ DENOTES THE COST TIME FOR ALL REMAINING OPERATIONS, SUCH AS IMAGE LOADING, UPDATING TRACKS.

Detections	Frames	Boxes	Density	t1	t2	t3	t4	fps
DPM	5919	135376	22.87	57.5	558.8	6.7	284.1	6.5
FRCNN	5919	110141	18.61	75.4	694.3	5.8	281.5	5.6
SDP	5919	128653	21.74	86.8	887.4	13.0	292.3	4.6
total	17757	374170	21.07	219.7	2140.5	25.5	857.8	5.5

The average speed of our tracker is 5.5 frames per second (fps) on the MOT17 and 6.5 fps on the MOT16, when implemented using the PyTorch framework on a computer with one NVIDIA Tesla V100 GPU. Although the speed is not satisfactory from the point of real-time application, we believe the current work is a starting point towards real-time application by optimizing the implementation and using more advanced computation devices.

F. GFLOPs and Number of Parameters

Our proposed DGCAN is similar to the end-to-end DAN [14] in the sense that both DAN and DGCAN use the same backbone network for feature extraction. The difference between DAN and DGCAN is how they estimate affinities between the objects in a frame pair. Thus we compare the GFLOPs and number of parameters of the proposed DGCAN with DAN to verify its usability. We utilize the widely-used tool [49] to compute the GFLOPs and number of parameters.

From Table IX, compared with DAN, our proposed DGCAN only increases GFLOPs and number of parameters by 0.54% and 1.8%, respectively. However, DGCAN improves the MOTA and IDF1 by 3.82% and 9.3%, respectively. Our proposed DGCAN improves DAN by a large margin while only introduces marginal overhead, demonstrating its usability.

TABLE IX
COMPARISON OF GFLOPS AND NUMBER OF PARAMETERS BETWEEN DAN AND OUR PROPOSED DGCAN.

Model	GFLOPs	# params	MOTA	IDF1
DAN	571.386	26.572M	52.4	49.5
DGCAN	574.476 (+0.54%)	27.06M (+1.8%)	54.4 (+3.82%)	54.1 (+9.3%)

V. CONCLUSIONS

In this paper, we propose a graph convolutional network-based model to incorporate the interaction between detection-detection pairs from two different frames in the learning process of multiple-pedestrian tracking, which we call Deep Graph Convolutional Affinity Network (DGCAN). The model combines appearance, motion, and topology cues, and jointly learns pedestrian representation and affinity estimation in an end-to-end manner. Based on the efficient affinity estimation module of DGCAN, we present an online multiple-pedestrian tracker. Evaluations on three widely used multiple-pedestrian tracking benchmarks demonstrate that the proposed tracker outperforms most existing online and offline trackers and achieves state-of-the-art performance.

REFERENCES

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, p. 1137, 2017.

[2] 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 Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 2129–2137.

[3] P. Feng, W. Wang, S. Dlay, S. M. Naqvi, and J. Chambers, "Social force model-based MCMC-OCSVM particle PHD filter for multiple human tracking," *IEEE Transactions on Multimedia*, vol. 19, no. 4, pp. 725–739, apr 2017.

[4] P. Dai, X. Wang, W. Zhang, and J. Chen, "Instance segmentation enabled hybrid data association and discriminative hashing for online multi-object tracking," *IEEE Transactions on Multimedia*, vol. 21, no. 7, pp. 1709–1723, jul 2019.

[5] Z. Fu, F. Angelini, J. Chambers, and S. M. Naqvi, "Multi-level cooperative fusion of GM-PHD filters for online multiple human tracking," *IEEE Transactions on Multimedia*, vol. 21, no. 9, pp. 2277–2291, sep 2019.

[6] T. Gao, H. Pan, Z. Wang, and H. Gao, "A CRF-based framework for tracklet inactivation in online multi-object tracking," *IEEE Transactions on Multimedia*, pp. 1–1, 2021.

[7] L. Lan, X. Wang, S. Zhang, D. Tao, W. Gao, and T. S. Huang, "Interacting tracklets for multi-object tracking," *IEEE Transactions on Image Processing*, vol. 27, no. 9, pp. 4585–4597, 2018.

[8] W. Feng, Z. Hu, W. Wu, J. Yan, and W. Ouyang, "Multi-object tracking with multiple cues and switcher-aware classification," *arXiv:1901.06129*, 2019.

[9] 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*, 2020, pp. 6247–6257.

[10] W. Feng, L. Lan, X. Zhang, and Z. Luo, "Learning sequence-to-sequence affinity metric for near-online multi-object tracking," *Knowledge and Information Systems*, vol. 62, no. 10, pp. 3911–3930, 2020.

[11] W. Feng, L. Lan, Y. Luo, Y. Yu, X. Zhang, and Z. Luo, "Near-online multi-pedestrian tracking via combining multiple consistent appearance cues," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1540–1554, 2021.

[12] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, "Learning by tracking: Siamese cnn for robust target association," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2016, pp. 33–40.

[13] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[14] S. Sun, N. Akhtar, H. Song, A. S. Mian, and M. Shah, "Deep affinity network for multiple object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[15] X. Shi, H. Ling, W. Hu, C. Yuan, and J. Xing, "Multi-target tracking with motion context in tensor power iteration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 3518–3525.

[16] L. Lan, D. Tao, C. Gong, N. Guan, and Z. Luo, "Online multi-object tracking by quadratic pseudo-boolean optimization." in *Proceedings of the International Joint Conferences on Artificial Intelligence*, 2016, pp. 3396–3402.

[17] J. Park, M. Lee, H. J. Chang, K. Lee, and J. Y. Choi, "Symmetric graph convolutional autoencoder for unsupervised graph representation learning," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2019, pp. 6519–6528.

[18] B. Yang and R. Nevatia, "An online learned crf model for multi-target tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2034–2041.

[19] C. Ma, Y. Li, F. Yang, Z. Zhang, Y. Zhuang, H. Jia, and X. Xie, "Deep association: End-to-end graph-based learning for multiple object tracking with conv-graph neural network," in *Proceedings of the International Conference on Multimedia Retrieval*. ACM, 2019, pp. 253–261.

[20] J. Li, X. Gao, and T. Jiang, "Graph networks for multiple object tracking," in *Proceedings of The IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2020, pp. 719–728.

[21] 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*. International Joint Conferences on Artificial Intelligence Organization, 2020, pp. 530–536.

[22] 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.

[23] Q. Li, Z. Han, and X.-M. Wu, "Deeper insights into graph convolutional networks for semi-supervised learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 2018.

[24] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *Advances in Neural Information Processing Systems Workshops*, 2017.

[25] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

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

[27] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv:1603.00831*, 2016.

[28] 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*, vol. 32, no. 9, pp. 1627–1645, 2009.

[29] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[30] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–10, 2008.

[31] B. Yang and R. Nevatia, "Multi-target tracking by online learning a crf model of appearance and motion patterns," *International Journal of Computer Vision*, vol. 107, no. 2, pp. 203–217, 2014.

[32] X. Zhou, V. Koltun, and P. Krähenbühl, "Tracking objects as points," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 474–490.

[33] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2015, pp. 4696–4704.

[34] 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*, vol. 42, no. 1, pp. 140–153, 2018.

[35] R. Henschel, L. Leal-Taixé, D. Cremers, and B. Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2018, pp. 1428–1437.

[36] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. Luk Chan, and G. Wang, "Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2016, pp. 1–8.

[37] R. Henschel, Y. Zou, and B. Rosenhahn, "Multiple people tracking using body and joint detections," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2019, pp. 0–0.

[38] 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*. IEEE, 2018, pp. 1–6.

[39] J. Xu, Y. Cao, Z. Zhang, and H. Hu, "Spatial-temporal relation networks for multi-object tracking," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2019, pp. 3988–3998.

[40] 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 International Conference on Computer Vision*. IEEE, 2019, pp. 6172–6181.

[41] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2019, pp. 941–951.

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

[43] W. Choi, "Near-online multi-target tracking with aggregated local flow descriptor," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2015, pp. 3029–3037.

[44] G. Wang, Y. Wang, H. Zhang, R. Gu, and J.-N. Hwang, "Exploit the connectivity: Multi-object tracking with trackletnet," in *Proceedings of the ACM International Conference on Multimedia*. ACM, 2019, pp. 482–490.

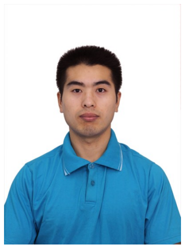
[45] L. Ma, S. Tang, M. J. Black, and L. Van Gool, "Customized multi-person tracker," in *Proceedings of the Asian Conference on Computer Vision*. Springer, 2018, pp. 612–628.

[46] L. Chen, H. Ai, R. Chen, and Z. Zhuang, "Aggregate tracklet appearance features for multi-object tracking," *IEEE Signal Processing Letters*, vol. 26, no. 11, pp. 1613–1617, 2019.

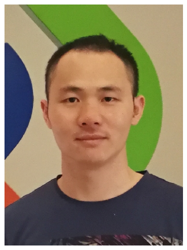
[47] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 5620–5629.

[48] A. Milan, S. Roth, and K. Schindler, "Continuous energy minimization for multitarget tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 1, pp. 58–72, 2013.

[49] L. Zhu, "Pytorch opcounter," https://github.com/Lyken17/pytorch-OpCounter.



Weijiang Feng is currently a lecture with the PLA Information Engineering University. He received the Ph.D. degree in computer science from the National University of Defense Technology in 2021. His research interests include multi-object tracking, computer vision, and quantum computing.



Long Lan is currently a lecturer with College of Computer, National University of Defense Technology. He received the Ph.D. degree in computer science from National University of Defense Technology 2017. He was a visiting Ph.D. student in University of Technology, Sydney from 2015 to 2017. His research interests include multi-object tracking, computer vision and discrete optimization.

Michael Buro is a professor in the computing science department at the University of Alberta in Edmonton, Canada. He received his PhD in 1994 for his work on Logistello - an Othello program that later defeated the reigning human World champion 6-0. His current research interests include heuristic search, machine learning, abstraction, state inference, and agent modeling applied to games.

Zhigang Luo received the B.S., M.S., and Ph.D. degrees from the National University of Defense Technology in 1981, 1993, and 2000, respectively. He is currently a Professor with the College of Computer, National University of Defense Technology. His current research interests include machine learning, computer vision, and bioinformatics.