Deep Association: End-to-end Graph-Based Learning for Multiple Object Tracking with Conv-Graph Neural Network

Cong Ma, Yuan Li*, Fan Yang, Ziwei Zhang, Yueqing Zhuang, Huizhu Jia, Xiaodong Xie
{Cong-Reeshard.Ma,yuanli,fyang.eecs,zhuangyq,zhangziw,hzjia,donxie}@pku.edu.cn
National Engineering Laboratory for Video Technology, Peking University
Beijing, China

ABSTRACT

Multiple Object Tracking (MOT) has a wide range of applications in surveillance retrieval and autonomous driving. The majority of existing methods focus on extracting features by deep learning and hand-crafted optimizing bipartite graph or network flow. In this paper, we proposed an efficient end-to-end model, Deep Association Network (DAN), to learn the graph-based training data, which are constructed by spatial-temporal interaction of objects. DAN combines Convolutional Neural Network (CNN), Motion Encoder (ME) and Graph Neural Network (GNN). The CNNs and Motion Encoders extract appearance features from bounding box images and motion features from positions respectively, and then the GNN optimizes graph structure to associate the same object among frames together. In addition, we presented a novel end-to-end training strategy for Deep Association Network. Our experimental results demonstrate the effectiveness of DAN up to the state-of-the-art methods without extra-dataset on MOT16 and DukeMTMCT.

CCS CONCEPTS

• **Computing methodologies** → **Tracking**.

KEYWORDS

Surveillance Retrieval, Computer Vision, Multiple Object Tracking, Deep Association, Graph Neural Network Deep Learning

ACM Reference Format:
Cong Ma, Yuan Li*, Fan Yang, Ziwei Zhang, Yueqing Zhuang, Huizhu Jia, Xiaodong Xie. 2019. Deep Association: End-to-end Graph-Based Learning for Multiple Object Tracking with Conv-Graph Neural Network. In *International Conference on Multimedia Retrieval (ICMR '19), June 10–13, 2019, Ottawa, ON, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3323873.3325010

ICMR '19, June 10–13, 2019, Ottawa, ON, Canada

ACM ISBN 978-1-4503-6765-3/19/06…$15.00
https://doi.org/10.1145/3323873.3325010

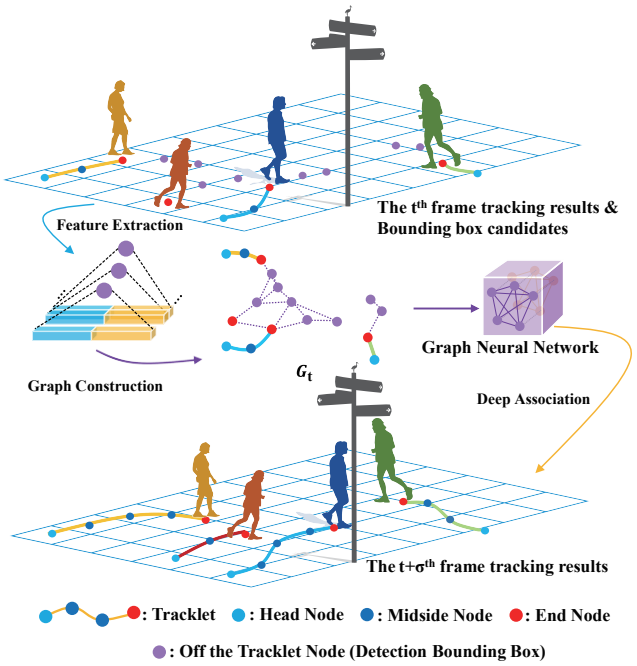

Figure 1: Deep Association Network for Multiple Object Tracking

1 INTRODUCTION

Multiple Object Tracking (MOT) is one of the significant components in computer vision, such as video surveillance retrieval, scene understanding and autonomous driving. MOT task is a process of acquiring trajectories, which identifies each individual object and associates them as several contiguous tracklets in a video sequence. The tracking results obtained by the tracker can be used for action recognition or retrieval information supplement. However, MOT is still a developable problem due to the negative influence of occlusion, scene complexity and indistinguishable objects.

Tracking-by-detection is a dominant solutions of MOT, which compares the similarity of each object within the inter-frames of the video on account of the general characteristic of the same individual. Tracking-by-detection usually depends on the bounding boxes detected by detector in every-frame or some of the frame. Therefore the current approaches extract the bounding boxes' features (e.g. appearance, motion and interactions) to associate each object in sequence.

Data Association is the fundamental of the MOT framework, which is generally divided into feature extraction and graph optimization. Feature extraction aims to describe effective information of bounding boxes, which include object colors, textures, positions, boundaries, etc. Traditional methods tend to extract hand-crafted features and keypoints.

Recently, some deep neural network such as Convolutional Neural Network (CNN) has gradually replaced conventional approaches mentioned before because of their excellent performance on feature extraction. Each bounding box is regarded as a node on the graph, while the similarity of nodes computed by features represents edge weight between nodes. Then these nodes and edges compose a Bipartite Graph for online tracking or Network Flow for offline tracking. Graph optimization focuses on connecting and eliminating the edge of graph so that each sub-graph is the same individual. However, the graph optimization on MOT has not relied on deep learning and continues to utilize the traditional solution such as Hungarian Algorithm [31]. Although the performance is gradually improving at the MOT Challenges [26] and DukeMTMCT [27], so far feature extraction and graph optimization are still independent tasks, They still havent been combined as an end-to-end model to be trained together, which causes the model not to learn the interaction of nodes and also reduces the processing efficiency due to data transmission.

In this paper, we address a distinctive MOT framework, Deep Association Network (as illustrated in Figure 1) and corresponding end-to-end graph-based training strategy. In the first half of the framework we still utilize CNN to extract features, while in the second half Graph Neural Network (GNN) replaces hand-crafted algorithm to optimize the graph. GNN has the ability to learn the interaction and relationship between nodes through a large amount of data. The advantage of GNN is that it can input arbitrary graph structures. Through specific loss function and large-scale tracking training data, GNN propagates node features on graph structures, ultimately nodes which belong to the same individual tend to be together. Besides, we design a Motion Encoder to describe the bounding box information such as position, size and shape. We connect three parts sequentially for end-to-end learning. Our contributions of the framework are as follows:

- End-to-end MOT model framework: We firstly present a end-to-end model which combines CNN, Motion Encoder and GNN. CNN and Motion Encoder are used for extracting appearance and motion features of bounding box respectively, and GNN optimizes each graph.
- Novel Training Strategy: We design graph dataset for training DAN. Each epoch includes several graphs constructed by bounding boxes, and the ground truth relationship of every node is utilized for supervising DAN.
- Developable MOT baseline: Deep Association Network is an unprecedented model structure for multiple object tracking, therefore DAN is worth continuing to explore and research how to improve the performance of MOT.

2 RELATED WORK

Multiple Object Tracking has attracted people's attention. An increasing number of researchers participate in this field. The performance of MOT improves gradually at the MOT benchmark[26].

For Multiple Object Tracking: Tracking-by-detection has become one of the most popular tracking frameworks. Among the methods of MOT, [5, 6, 14, 16, 36, 40] designed an ingenious data association or multiple hypothesis. [32] firstly combined feature extraction part and hand-crafted graph structure to learn together. [20, 24] presented network flow and graph optimization which are powerful approaches. [30, 36, 42] train the CNN on the basis of person re-identification to extract the image features, and [35] utilizes the quadruplet loss to enhance the feature expression. [7] builds the CNN model to generate visibility maps to solve the occlusion problem. In addition, [14] uses a novel multi-object tracking formulation to incorporate several detector into a tracking system. [16] extends the multiple hypothesis by enhancing the detection model. [23] addressed a sophisticated model to process trajectories. [11, 48] proposed spatial and temporal attention mechanisms to enhance the performance of MOT. The motion model is divided into linear position prediction [35] and non-linear position prediction [8]. [15] designs the structural constraint by the location of people to optimize assignment. Following the success of RNN models for sequence prediction tasks, [2] proposes social-LSTM to predict the position of each person in the scene.

For Graph Neural Network: GNN was previously applies to Natural Language Programming (NLP), physical simulation and etc. For instance, [3] summarized the principle and application of GNN. [19, 21, 38] respectively proposed the GNN variant structure, GGSNN (Gated Graph Sequence Neural Network), GCN (Graph Convolutional Network) and GAT (Graph Attention Network). [9] focuses on molecule feature descriptor, and each molecule is composed by atoms as the graph structure. [18] aims to research physical simulation by GNN, more specifically the interaction of dynamical particles system, meanwhile they realized NBA player trajectories prediction. Recently, GNN has been utilized for computer vision. GNN-based few-shot transfer learning presented by [12], and polygon refinement for instance segmentation addressed by [1]. [41] adopted spatial-temporal skeleton graph for action recognition, and [33] constructed relationship graph to train re-identification model.

3 DEEP ASSOCIATION FRAMEWORK

The major applications of MOT focus on pedestrian tracking, whose purpose is to estimate the locations of each person at different times in the video. Most methods have taken advantage of diversified cues to improve tracking precision. On this basis, we propose an end-to-end training framework, Deep Association Netowrk (DAN), which combines multiple cues (Appearance feature, Motion Position and Interaction) to co-learn the behavioral pattern of inter-individual. In this Section, our model pipeline skeleton and the definition of

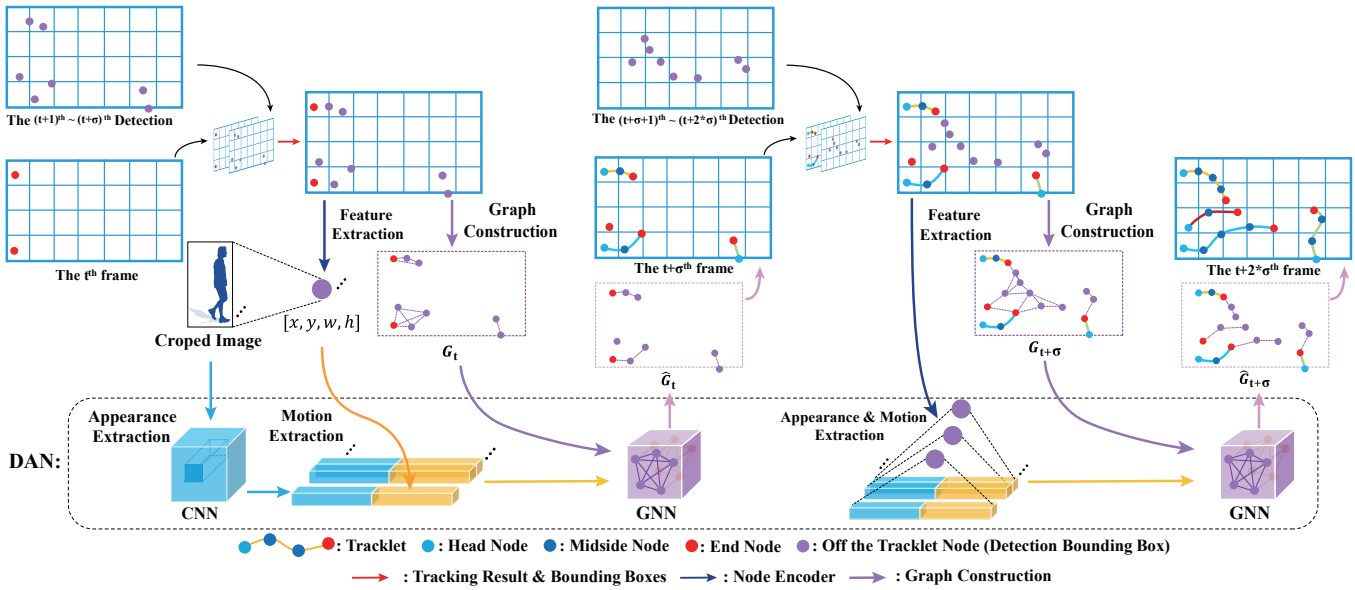

Figure 2: The Framework of Deep Association Network

our tasks are described in Sec.3.1. The details of feature extraction are introduced in Sec.3.2. We demonstrate how to construct graph in Sec.3.3. Lastly Sec.3.4 gives the strategy for training GNN.

3.1 Deep Association Pipeline

The traditional MOT frameworks are based on tracking-by-detection strategy, which associate bounding boxes by appearance feature, motion prediction and association optimization (Network Flow [32], Hungarian Algorithm [31]). In our framework, Deep Association Newtork (DAN) is composed of Convolutional Neural Network (CNN), Motion Encoder (ME) and Graph Neural Network (GNN) (as described in Figure 2). Firstly, we treat the bounding boxes as nodes, which are located by detection model frame by frame. CNN is applied to extract appearance features from images in bounding box, meanwhile ME is utilized for encoding corresponding bounding boxes' information, which contains position, width, height and velocity of the bounding box as the motion features. We build an adjacency matrix according to the spatial-temporal cues of bounding boxes to represent the graph structure of the relationship between nodes. The adjacent matrix and concatenations of appearance and motion features are feed into GNN. GNN propagates node features each other on graph structure and learns the relationship between nodes. Finally, the nodes feature is projected on high-dimensional space. The features in the same aggregation range can be treated as the same person, which are sequentially linked on the timeline to form a complete trajectory.

We formulate the near-online tracking problem as the local bounding boxes association task between tracked candidate results and current detection results within video fragment. we define the set of t-th to $(t+\eta)$-th frames detection results as $\mathcal{D}_t$ ($d_t^k \in \mathcal{D}_t$), η is the width of subsequence on video, and the set of the first t-th frames tracked candidate results as $\mathcal{C}_t$ ($c_n^k \in \mathcal{C}_t; n \leq t, \mathcal{C}_t = \mathcal{C}_{t-1} \bigcup \mathcal{D}_{t-1}$), where d_t^k and c_t^k are k-th detection and candidate in frame t, respectively. Bounding boxes association can be perceived as graph optimization. Therefore, we construct a global graph structure $\mathcal{G}(G_t \in \mathcal{G}, G_t = (\mathcal{V}_t, \mathcal{E}_t))$, the global graph $\mathcal{G} = \{G_1, G_{1+\delta}, G_{1+2\delta} ..., G_L\}$ consists of several local graphs, where G_t is the local graph constructed from video fragment of t-th to $(t+\eta)$-th frames, δ is the stride of video fragment on timeline, L is the total length of the video, $\mathcal{V}_t$ ($v_\xi^k \in \mathcal{V}_t, \xi \in [t, t+\eta], \mathcal{V}_t = \mathcal{C}_t \cup \mathcal{D}_t$) indicates the set of nodes, each node stands for a bounding box and v_ξ^k denotes the k-th node in frame ξ, and nodes are defined as 7 dimensions $[t, id, x, y, w, h, s]$ that contain the tracklet id by tracker, the object time, the center position (x, y), width and height of the bounding box, and the statement of node ("Unallocated", "Tracked", "Lost", "Quitted"). $e_t^{ij} \in \mathcal{E}_t$ is the edge between v_t^i and v_t^j on G_t. The formulation of optimized graph is given by

$$argmin\Big(\sum_{G_t \in \mathcal{G}} F_S(v_t^i, v_t^j) e_t^{ij} + \sum_{G_{t_1}, G_{t_2} \in \mathcal{G}} F_S(v_{t_1}^i, v_{t_2}^j) e_t^{ij}\Big) \quad s.t.\ G_{t_1} \cap G_{t_2} \neq \varnothing \tag{1}$$

where $F_S(v_t^i, v_t^j)$ measures the similarity between nodes, $e_t^{ij} \in \{0, 1\}$ indicates whether two nodes, v_t^i and v_t^j, are connected on graph G_t. Eq.1 includes local graph optimization and inter-graph optimization. The goal of local graph optimization is to associate the nodes in batch, and the purpose of the inter-graph optimization aims to connect the cross-nodes between batches.

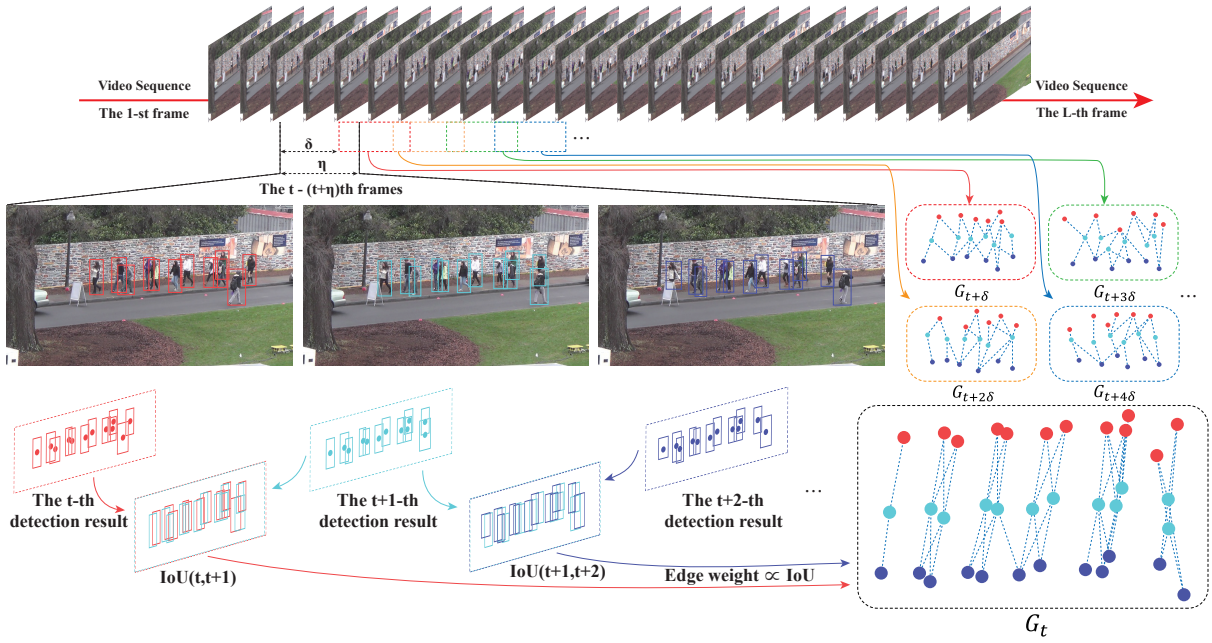

Figure 3: Illustration of the Graph Construction.

3.2 Feature Extraction

Feature Extraction is used to describe the characteristics of the individual and distinguish the differences between the nodes (bounding boxes). For the same individual in different time, it has the similar features for a period of time such as wearing, position, body size and velocity. These cues are totally summarized as two parts: appearance features and motion features. In our framework, the appearance features are extracted by several shared-weight CNNs, and the motion features are encoded by fully-connected networks.

Appearance model extracts the pedestrian features (e.g. color, shape and texture) from each bounding box located by detection model. we treats the appearance model as a person re-identification (Re-ID) task initially to obtain the pre-train model for CNNs on DAN. We combine the three public Re-ID datasets (Market1501 [45], DukeMTMC-ReID [46] and CUHK03 [47]) to train the homostructural CNNs model as DAN. $f^i_{a,t}$ and $f^i_{cls,t}$ indicate the output of CNN's appearance feature and classification vector respectively for node v^i_t, the $f^i_{a,t}$ is the k-dimensional vector, and the $f^i_{cls,t}$ is mapped to the n-dimensional vector by fully-connected layer from $f^i_{a,t}$, n denotes the training set classes number. $F_a(*)$ represents the model forward function of appearance model

$$f^i_{cls,t}, f^i_{a,t} = F_A(I^i_t) \tag{2}$$

where I^i_t indicates the croped image of the node v^i_t. We use the cross-entropy loss $\mathcal{L}_{cls}(*)$ in the multi-classification task for identification:

$$\mathcal{L}_{cls}(f^i_{cls,t}) = \sum_{i=1}^{K} -p^i_t log(\hat{p^i_t}), \quad \hat{p^i_t} = softmax(f^i_{cls,t}) \tag{3}$$

When the identification loss tends to convergence, all of the parameters will be loaded into CNNs from DAN as the pre-train model.

Motion Encoder is utilized for encoding the bounding boxes' information(position and shape). The ME model projects the 4 dimensional vector into a m-dimensional vector by fully-connected network, and $f^i_{m,t}$ denotes the motion feature for node v^i_t, $F_M(*)$ is the model forward function of motion model:

$$f^i_{m,t} = F_M(B^i_t) \tag{4}$$

where B^i_t is the bounding box information $[x^i_t, y^i_t, w^i_t, h^i_t]$ of v^i_t. We concatenate the appearance feature $f^i_{a,t}$ and motion feature $f^i_{m,t}$ together as the node representation to feed into GNN.

3.3 Graph Construction

Before feeding into GNN, we transform the video sequence information into the input standardization formats of GNN. The graph structure consists of nodes and edges represented as $G = (\mathcal{V}, \mathcal{E})$ (mentioned in Sec 3.1), where the nodes $v \in \mathcal{V}$ represent the bounding boxes from frames images, and the edges $e \in \mathcal{E}$ denote the spatial-temporal relationship between nodes. The graph construction is described in Figure 3. We divide the whole video sequence into several video fragments (sub-sequence). η is the length of video fragments, for instance in order to generate the graph G_t, we firstly extract the t-th frame to the $(t+\eta)$-th frame images and select nodes from sub-sequence in terms of the node statement. The statement of the node can be divide into "Tracked", "Lost", "Unallocated", and "Quitted". The nodes that have already been allocated

and are not the tail of the tracklet are treated as "Tracked" nodes. All tails of the tracklet nodes belong to "Lost" nodes (e.g. $\mathcal{C}_t$ includes all of the $[t, t+\eta]$-th frames lost nodes), and the fresh nodes which haven't been allocated are regarded as "Unallocated" nodes (e.g. $\mathcal{D}_t$ includes all of the $[t, t+\eta]$-th frames unallocated nodes). If the tracklet ultimately walks out of the image boundary, all of its nodes are classified as "Quitted". Two types of nodes, "Lost" and "Unallocated", are used for construct the graph structure. We calculate the node's bounding boxes IoU (Intersection over Union) between adjacent frames, and the edge weight is proportional to IoU value and inverse correlation to the frame interval. The edge weight $e_{\varepsilon i,\xi j}$ of graph G_t between B^i_ε and B^j_ξ is defined as:

$$e_{\varepsilon i,\xi j} = IoU(B^i_\varepsilon, B^j_\xi) * (1 - \mu * min(abs(\varepsilon - \xi), \lambda)) \\ s.t. \;\; \varepsilon, \xi \in [t, t+\eta], \;\; \varepsilon < \xi, \;\; \mu, \lambda > 0 \tag{5}$$

where

$$IoU(B^i_\varepsilon, B^j_\xi) = \frac{area(B^i_\varepsilon \cap B^j_\xi)}{area(B^i_\varepsilon \cup B^j_\xi)} \tag{6}$$

where constant μ is used for adjusting the influence of frame interval on edge weight, and constant λ is the upper limit of frame interval. The representation of graph G_t includes adjacent matrix $A_t \in \mathbb{R}^{N*N}$, $A_t[i,j] = e_{i,j}$ and features matrix $X_t \in \mathbb{R}^{N*(n+m)}$, $X_t[i] = [f^i_a, f^i_m]$, where N is the number of nodes in sub-sequence, each raw of X_t is the concatenation of appearance feature f_a and motion feature f_m.

3.4 Graph Optimization by GNN

Graph Neural Network (GNN) aims to learn the topology data pattern and represent the graph structure feature, which encodes the node features and updates the representation vector from neighborhood the other nodes aggregation on the graph. Better than CNN and RNN, GNN has more significant effects on the graph structure based task, such as molecules classification and particles interaction simulation.

The target of multiple object tracking task is to local every pedestrians position at each moment. So we associate the node ID and connect the nodes which belong to the same person as the tracking result. The MOT method address this problem by Data Association, which involves network flow, graph-cut and feature clustering, so GNN is able to optimize the graph nodes feature and edge weights between nodes. we adopt the Graph Convolutional Network (GCN) [19] as the network backbone. The adjacent matrix A_t and features matrix X_t are denoted as the GCN input, and the GCN outputs include updated $\hat{A}_t$ and $\hat{X}_t$. $F_G(*)$ indicates the model forward function of GCN:

$$\hat{A}, \hat{X} = F_G(A, X) \tag{7}$$

The internal implementation of GCN network is defined as:

$$\hat{X} = ReLU(\tilde{\Gamma}^{-\frac{1}{2}} \tilde{A} \tilde{\Gamma}^{-\frac{1}{2}} X \Theta) \tag{8}$$

$$\tilde{A} = A + \Psi_N, \;\; \tilde{\Gamma}[i,i] = \sum_j \tilde{A}[i,j] \tag{9}$$

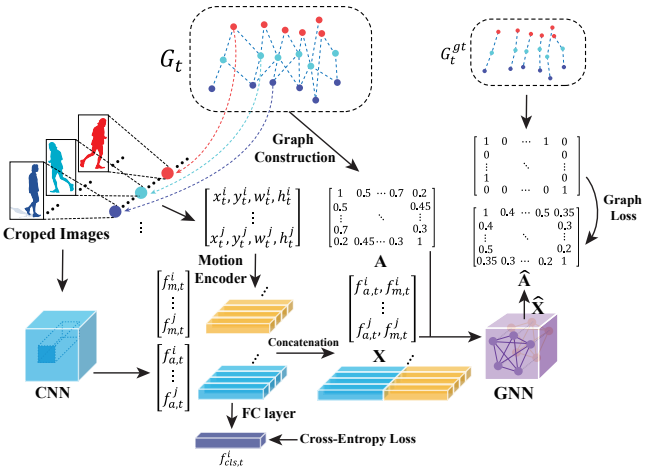

Figure 4: The explanation of DAN training and Loss Function.

Ψ_N is the identity self-connections matrix, and the $\tilde{A}$ is the combination of adjacency matrix and self-connections. $\tilde{\Gamma}$ indicates a degree matrix of graph G, and $\Theta \in \mathbb{R}^{(n+m)*p}$ is a learnable parameters on GCN, and feature matrix $\tilde{X} \in \mathbb{R}^{N*p}$ denotes one of the GCN output. The updated adjacency matrix $\hat{A} \in \mathbb{R}^{N*N}$ is given by:

$$\hat{A} = \frac{(norm(\hat{X}) * norm(\hat{X})^T) + 1}{2}, \;\; norm(\hat{X}) = \frac{\hat{X}}{|\hat{X}|} \tag{10}$$

where $\hat{A}[i,j]$ indicates the cosine distance between node features $\hat{X}[i]$ and $\hat{X}[j]$, and we normalize the features to $[0,1]$. The multi-layers GCN feedward function is shown as:

$$\hat{A}_1, \hat{X}_1 = F_{G_1}(A, X) \tag{11}$$

$$\hat{A}_\zeta, \hat{X}_\zeta = F_{G_{\zeta-1}}(\hat{A_{\zeta-1}}, \hat{X_{\zeta-1}}), \;\; \zeta > 1 \tag{12}$$

Finally, to train the DAN, we design a Graph Loss $\mathcal{L}_G(*)$, which is defined as:

$$\mathcal{L}_G(\hat{A}_\zeta, G^{gt}_t) = \sum_{e^{gt}_{ij} \in \mathcal{E}^{gt}_t} (e^{gt}_{ij} - e_{ij}) + \sum_{e^{gt}_{ij} \notin \mathcal{E}^{gt}_t} \sigma * (e_{ij} - e^{gt}_{ij}) \tag{13}$$

where G^{gt}_t is the ground truth graph structure which is computed previously, $G^{gt}_t = (\mathcal{V}^{gt}_t, \mathcal{E}^{gt}_t)$, $e^{gt}_{ij} \in \mathcal{E}^{gt}_t$, $e^{gt}_{ij} = \{0, 1\}$, and σ is the loss weight of Graph Loss. The total loss $\mathcal{L}$ of the training DAN includes cross-entropy loss $\mathcal{L}_{cls}$ for appearance model and graph loss $\mathcal{L}_G$ for motion encoder and GCN: $\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_G$ (described in Figure 4).

4 EXPERIMENTS

4.1 MOT Datasets

To train the DAN, we prepare the training dataset MOT16 [26] and DukeMTMCT [27] which contain the ground truth location of each frame bounding box by annotator.

DukeMTMCT is a large scale dataset for multiple camera multiple object tracking, which the videos captured by

Table 1: Results on the DukeMTMCT test dataset

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
PT_BIPCC [25]	59.3	71.2	666	234	71381	361673	298	799
BIPCC [28]	59.4	70.1	665	234	68634	361589	290	783
MTMC_ReIDp [44]	70.7	79.2	726	143	52408	277762	449	1060
MTMC_CDSC [37]	70.9	77.0	740	110	38655	268398	693	4717
MYTRACKER [43]	73.8	80.3	914	72	**35580**	193253	406	1116
TAREIDMTMC [5]	83.3	83.8	1051	17	44691	131220	383	2428
DeepCC [29]	**87.5**	**89.2**	**1103**	29	37280	**94399**	**202**	**753**
DAN(Ours)	86.7	82.0	1088	**9**	37073	102930	928	4357

Table 2: Results on the MOT16 test dataset

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDSw.↓	Frag↓
QuadMOT16 [35]	44.1	38.3	14.6%	44.9%	6388	94775	745	1096
EDMT [5]	45.3	47.9	17.0%	39.9%	11122	87890	639	946
MHT_DAM [16]	45.8	46.1	16.2%	43.2%	6412	91758	590	781
STAM16 [7]	46.0	50.0	14.6%	43.6%	6895	91117	473	1422
NOMT [6]	46.4	53.3	18.3%	41.4%	9753	87565	**359**	504
AMIR [30]	47.2	46.3	14.0%	41.6%	**2681**	92856	774	1675
NLLMPa [39]	47.6	47.3	17.0%	40.4%	5844	89093	629	768
MOTDT [22]	47.6	50.9	15.2%	38.3%	9253	**85431**	792	1858
FWT [13]	47.8	44.3	**19.1**%	**38.2**%	8886	85487	852	1534
GCRA [23]	48.2	48.6	12.9%	41.1%	5104	88586	821	1117
TLMHT [34]	48.7	**55.3**	15.7%	44.5%	6632	86504	413	642
LMP [36]	**48.8**	51.3	18.2%	40.1%	6654	86245	481	**595**
DAN(Ours)	48.6	49.3	13.2%	43.5%	5854	87260	594	806

8 surveillance cameras at different viewing angles include 2800 identities (person) on the Duke University . The video duration of each camera is 86 minutes, which is splited into training set (0-50 min) and testing set (50-86 min). In addition, the dataset provided DPM [10] and Openpose [27] detection results for each frame as the tracker input.

MOT16 is a classical evaluation dataset comparing several tracking methods on MOT Challenge, which include 14 sequences captured from surveillance, hand-held shooting and driving recorder by static camera and moving camera. The length of each video is about 500-1500 frames. And the dataset also provides the detections DPM.

4.2 DAN training Strategy

To train the DAN, we firstly divide training sequence into many shot sub-sequence, and the length of each sub-sequence is about 20-30 frames. If every frame includes 15-20 pedestrians, we obtain 300-600 nodes in total from sub-sequence and construct them as the graph. For the whole sequence, we can get about 2k-10k graphs, and the number of graph depends on sub-sequence length and sample stride length. And we shuffle these graphs for each epoch, and our experiment implements on the Pytorch framework by 4 Nvidia Titan X GPUs, and device 0,1,2 are used for loading the CNN model to extract appearance features, and then the features transfers to device 3 to calculate motion encoder and GCN model. The processing of network feedward and backward are shown as **Algorithm 1**.

Algorithm 1: Training Deep Association Network

Input: $\mathcal{G} = \{G_1, G_{1+\delta}, G_{1+2\delta}..., G_L\},\ G_t = \{I_t, A_t, B_t\}$
Output: F_A, F_M, F_G *model weight*

for $epoch = 1 : max\_epoch$ **do**
  $\mathcal{G}' =$ shuffle($\mathcal{G}$);
  for $I_t^i, A_t, B_t, f_{cls,t}^{gt}, G_t^{gt}$ *in* $\mathcal{G}'$ **do**
    $f_{cls,t}, f_{a,t} = F_A(I_t)$;
    $f_{m,t} = F_M(B_t)$;
    $X_t = concatenate[f_{a,t}, f_{m,t}]$;
    $\hat{A}_t, \hat{X}_t = F_G(A_t, X_t)$;
    $cls\_loss = \mathcal{L}_{cls}(f_{cls,t}^i, f_{cls,t}^{gt})$;
    $graph\_loss = \mathcal{L}_G(\hat{A}_t, G_t^{gt})$;
    $cnn\_loss = cls\_loss + graph\_loss$;
    $F_A.backward(cnn\_loss, lr_1)$;
    $F_M.backward(graph\_loss, lr_2)$;
    $F_G.backward(graph\_loss, lr_3)$;
Return: F_A, F_M, F_G

4.3 Implementation Details

In out experiments, DAN consists of CNN, ME and GNN. The training set graph size is restricted to 64-512 nodes per graph, the length of sub-sequence η is 30 frames, and the stride of sequence δ is 20. For each step of epoch, we replace data batch size with graph size to feed CNN model. We train the CNN model as appearance model with ResNet-50, and the images are resized to 256 × 256 from croped images and the outputs

Figure 5: The visualization result on DukeMTMCT and MOT16/17

of CNN produces appearance feature $f_{a,t}$, a 2048-dimensional vector to describe image. Motion Encoder (ME) is composed by 2-layers fully-connected network, batch-normalization and ReLU. Bounding box information [x,t,w,h], a 4-dimensional vector is raised to $4 \rightarrow 64 \rightarrow 512$ dimensional vector finally by ME. For graph construction, the frame interval impact factor μ is 0.08 and the upper limit constant λ is 10. The input of GCN is a N $\times$ 2560-dimensional vector, where N is the number of nodes on graph, and we adopt a two-layers GCN and the graph loss weight σ is 2. The training optimizer is the AdamOptimizer [17], and initially learning rate is set to 0.001. The model converges finally at the 150th epoch.

4.4 MOT Evaluation Metrics

The MOT Challenge Benchmark adopted the standard metrics [4] [28] for evaluation MOT performance. The main metrics for MOT are MOTA and IDF1. MOTA (Multiple Object Tracking Accuracy) measures the effect of tracking for each tracklet, which depends on False Positives (FP), False Negatives (FN) and Id Switches (IDSw). The IDF1 (ID F1 Score) is the ratio of correctly identified detection over the average number of true and computed detections. Mostly tracked targets (MT), Mostly lost targets (ML) and the total number of Fragment (Frag) are used for evaluating tracklets integrity as the reference indexes.

4.5 Trackers Results Comparison

Here we present our results on the MOT Challenge testing set, and compare our method with the best published results on the benchmark. The trackers results comparison on MOT16 and DukeMTMCT are shown in Table 1 and Table 2. Compared with the previous result, the MOTA performance of our proposal on the MOT16 rank 3, however the TLMHT [34] method includes post-processing to associate trajectories. LMP [36] adopts the person keypoints to extract features in detail. And the other reason is the MOT16 detection results provided by Benchmark exists the false or incorrect-position bounding boxes, but our proposal is base on the accurate bounding boxes to train the model. On DukeMTMCT, we improve MOTA by 3.4%, and for MT, ML and FN, we also achieve the preferable performance on Benchmark.

5 CONCLUSION

We propose a novel network framework for MOT, which is combined CNN, ME and GNN. CNN and ME are utilized for extracting with node features and GNN optimizes graph. Compared with existing methods, the approach is a bold attempt to end-to-end training network on MOT task as developable baseline in the future. The algorithm achieves MOTA up to 48.2, 86.7 and IDF1 up to 48.6, 82.0 on MOT16 and DukeMTMCT respectively that approaches the state-of-the-art methods. The visualization MOT results are shown in Figure 5. Currently, our proposal still hasn't done the excellent effect on occlusion and moving camera. As for future work, we will continue to explore the principle of GNN and improve the performance of DAN.

6 ACKNOWLEDGMENTS

This work is partially supported by the National Key Research and Development Program of China under contract No. 2016YFB0401904, the Major National Scientific Instrument and Equipment Development Project of China under contract No. 2013YQ030967, National Key Research and Development Program of China (2016YFB0401904), we additionally thank NVIDIA for generously providing DGX-1 super-computer and support through the NVAIL program.

REFERENCES

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. 2018. Efficient Interactive Annotation of Segmentation Datasets With Polygon-RNN++. In *CVPR*. 859–868.
[2] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. 2016. Social lstm: Human trajectory prediction in crowded spaces. In *CVPR*. 961–971.
[3] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* (2018).
[4] Keni Bernardin and Rainer Stiefelhagen. 2008. Evaluating multiple object tracking performance: the CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing* 2008, 1 (2008), 246309.
[5] Jiahui Chen, Hao Sheng, Yang Zhang, and Zhang Xiong. 2017. Enhancing Detection Model for Multiple Hypothesis Tracking. In *CVPR Workshops*. 18–27.
[6] Wongun Choi. 2015. Near-online multi-target tracking with aggregated local flow descriptor. In *ICCV*. 3029–3037.
[7] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. 2017. Online Multi-Object Tracking Using CNN-Based Single Object Tracker With Spatial-Temporal Attention Mechanism. In *CVPR*. 4836–4845.
[8] Caglayan Dicle, Octavia I Camps, and Mario Sznaier. 2013. The way they move: Tracking multiple targets with similar appearance. In *ICCV*. 2304–2311.
[9] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232.
[10] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. 2010. Object detection with discriminatively trained part-based models. *IEEE TPAMI* 32, 9 (2010), 1627–1645.
[11] Xu Gao and Tingting Jiang. 2018. OSMO: Online Specific Models for Occlusion in Multiple Object Tracking Under Surveillance Scene. In *2018 ACM Multimedia Conference on Multimedia Conference*. 201–210.
[12] Victor Garcia and Joan Bruna. 2018. Few-shot learning with graph neural networks. *ICLR* (2018).
[13] Roberto Henschel, Laura Leal-Taix, Daniel Cremers, and Bodo Rosenhahn. 2017. A Novel Multi-Detector Fusion Framework for Multi-Object Tracking. (2017).
[14] Roberto Henschel, Laura Leal-Taixé, Daniel Cremers, and Bodo Rosenhahn. 2018. Fusion of head and full-body detectors for multi-object tracking. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*.
[15] Ju Hong Yoon, Chang-Ryeol Lee, Ming-Hsuan Yang, and Kuk-Jin Yoon. 2016. Online multi-object tracking via structural constraint event aggregation. In *CVPR*. 1392–1400.
[16] Chanho Kim, Fuxin Li, Arridhana Ciptadi, and James M Rehg. 2015. Multiple hypothesis tracking revisited. In *ICCV*. 4696–4704.
[17] Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *ICLR* (2015).
[18] Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. 2018. Neural relational inference for interacting systems. *ICML* (2018).
[19] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *ICLR* (2017).
[20] Evgeny Levinkov, Jonas Uhrig, Siyu Tang, Mohamed Omran, Eldar Insafutdinov, Alexander Kirillov, Carsten Rother, Thomas Brox, Bernt Schiele, and Bjoern Andres. 2017. Joint Graph Decomposition and Node Labeling: Problem, Algorithms, Applications. (2017).
[21] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2016. Gated graph sequence neural networks. *ICLR* (2016).
[22] Zijie Zhuang Chong Shang Long Chen, Haizhou Ai. 2018. Real-time Multiple People Tracking with Deeply Learned Candidate Selection and Person Re-identification. *ICME* (2018).
[23] Cong Ma, Changshui Yang, Fan Yang, Yueqing Zhuang, Ziwei Zhang, Huizhu Jia, and Xiaodong Xie. 2018. Trajectory Factory: Tracklet Cleaving and Re-connection by Deep Siamese Bi-GRU for Multiple Object Tracking. *ICME* (2018).
[24] Andrii Maksai, Xinchao Wang, Francois Fleuret, and Pascal Fua. [n. d.]. Globally Consistent Multi-People Tracking using Motion Patterns. ([n. d.]).
[25] Andrii Maksai, Xinchao Wang, Francois Fleuret, and Pascal Fua. 2017. Non-markovian globally consistent multi-object tracking. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2563–2573.
[26] Anton Milan, Laura Leal-Taixé, Ian D. Reid, Stefan Roth, and Konrad Schindler. 2016. MOT16: A Benchmark for Multi-Object Tracking. *CoRR* abs/1603.00831 (2016). arXiv:1603.00831 http://arxiv.org/abs/1603.00831
[27] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. 2016. Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. In *ECCV workshop on Benchmarking Multi-Target Tracking*.
[28] Ergys Ristani, Francesco Solera, Roger Zou, Rita Cucchiara, and Carlo Tomasi. 2016. Performance measures and a data set for multi-target, multi-camera tracking. In *European Conference on Computer Vision*. Springer, 17–35.
[29] Ergys Ristani and Carlo Tomasi. 2018. Features for Multi-Target Multi-Camera Tracking and Re-Identification. *CVPR* (2018).

[30] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. 2017. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *ICCV* (2017).
[31] Bima Sahbani and Widyawardana Adiprawita. 2017. Kalman filter and iterative-hungarian algorithm implementation for low complexity point tracking as part of fast multiple object tracking system. In *ICSET*. 109–115.
[32] Samuel Schulter, Paul Vernaza, Wongun Choi, and Manmohan Chandraker. 2017. Deep Network Flow for Multi-Object Tracking. In *CVPR*. 6951–6960.
[33] Yantao Shen, Hongsheng Li, Shuai Yi, Dapeng Chen, and Xiaogang Wang. 2018. Person Re-identification with Deep Similarity-Guided Graph Neural Network. In *ECCV*. Springer, 508–526.
[34] Hao Sheng, Jiahui Chen, Yang Zhang, Wei Ke, Zhang Xiong, and Jingyi Yu. 2018. Iterative Multiple Hypothesis Tracking with Tracklet-level Association. *IEEE Transactions on Circuits and Systems for Video Technology* (2018).
[35] Jeany Son, Mooyeol Baek, Minsu Cho, and Bohyung Han. 2017. Multi-Object Tracking With Quadruplet Convolutional Neural Networks. In *CVPR*. 5620–5629.
[36] Siyu Tang, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. 2017. Multiple people tracking by lifted multicut and person reidentification. In *CVPR*. 3539–3548.
[37] Yonatan Tariku Tesfaye, Eyasu Zemene, Andrea Prati, Marcello Pelillo, and Mubarak Shah. 2017. Multi-target tracking in multiple non-overlapping cameras using constrained dominant sets. *arXiv preprint arXiv:1706.06196* (2017).
[38] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *ICLR* (2018). https://openreview.net/forum?id=rJXMpikCZ accepted as poster.
[39] Bing Wang, Li Wang, Bing Shuai, Zhen Zuo, Ting Liu, Kap Luk Chan, and Gang Wang. 2016. Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association. In *CVPR Workshops*. 1–8.
[40] Yu Xiang, Alexandre Alahi, and Silvio Savarese. 2015. Learning to track: Online multi-object tracking by decision making. In *ICCV*. 4705–4713.
[41] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial temporal graph convolutional networks for skeleton-based action recognition. *AAAI* (2018).
[42] Fan Yang, Ke Yan, Shijian Lu, Huizhu Jia, Xiaodong Xie, and Wen Gao. 2019. Attention driven person re-identification. *Pattern Recognition* 86 (2019), 143 – 155. https://doi.org/10.1016/j.patcog.2018.08.015
[43] Kwangjin Yoon, Young-min Song, and Moongu Jeon. 2018. Multiple hypothesis tracking algorithm for multi-target multi-camera tracking with disjoint views. *IET Image Processing* (2018).
[44] Zhimeng Zhang, Jianan Wu, Xuan Zhang, and Chi Zhang. 2017. Multi-Target, Multi-Camera Tracking by Hierarchical Clustering: Recent Progress on DukeMTMC Project. *arXiv preprint arXiv:1712.09531* (2017).
[45] Liang Zheng, Liyue Shen, Lu Tian, Shengjin Wang, Jingdong Wang, and Qi Tian. 2015. Scalable person re-identification: A benchmark. In *Proceedings of the IEEE International Conference on Computer Vision*. 1116–1124.
[46] Zhedong Zheng, Liang Zheng, and Yi Yang. 2017. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. *arXiv preprint arXiv:1701.07717* 3 (2017).
[47] Zhun Zhong, Liang Zheng, Donglin Cao, and Shaozi Li. 2017. Re-ranking person re-identification with k-reciprocal encoding. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 3652–3661.
[48] Ji Zhu, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. 2018. Online Multi-Object Tracking with Dual Matching Attention Networks. In *ECCV*.