Neurocomputing 368 (2019) 69–83

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A dual CNN–RNN for multiple people tracking

Maryam Babaee*, Zimu Li, Gerhard Rigoll

Institute for Human-Machine Communication, Department of Electrical and Computer Engineering, Technical University of Munich, Arcisstr. 21, Munich, Germany

ARTICLE INFO

Article history:
Received 21 September 2018
Revised 22 June 2019
Accepted 2 August 2019
Available online 12 August 2019

Communicated by Dr. Tianzhu Zhang

Keywords:
Tracking
Motion estimation
Multicut graph decomposition
Deep learning

ABSTRACT

In this paper, we present a deep learning-based approach, namely a dual CNN–RNN for multiple people tracking. We follow tracking-by-detection paradigm by first training a CNN to measure the similarity of two detection boxes. To solve the data association (DA) problem, we build a graph with nodes as detections and edge costs that are the outputs of a CNN. The general minimum cost lifted multi-cut problem (LMP) and corresponding optimization algorithms are utilized to solve the DA problem. To tackle occlusion and ID-switch problems, an RNN network is proposed to predict the nonlinear motion of people. Moreover, we utilize target motion information to stitch tracklets and build long trajectories. The results of our experiments conducted on Multiple Object Tracking Benchmark 2016 (MOT2016) confirm the efficiency of the proposed algorithm.

1. Introduction

Multiple object tracking (MOT) is one of the most active research areas in computer vision, video surveillance, and biometrics. Although there have been many algorithms proposed for tracking [1–4], there are still some challenges like ID-switch and long term occlusion that have not been addressed thoroughly. These issues are getting more difficult to solve, particularly in crowded scenes where other effective circumstances like target interactions happen.

Tracking-by-detection (TBD) has been the most common proposed tracking framework in the past two decades [4–6]. In this framework, targets are first detected in all frames of a video sequence and then a data association algorithm is utilized to associate detections over all frames in order to build the trajectories. Obviously, advancements in both target detection and data association lead to an enhanced performance in tracking [7,8]. In multiple people tracking, the data association of detections is often a very challenging task due to missing detections, occlusion and target interactions in crowded environments. In recent years, there has been much effort on the association part of tracking, which led to many sophisticated data association algorithms such as joint probabilistic data association (JDPA) [9], integer linear programming (ILP) [10], multiple hypothesis tracking (MHT) [11], quadratic boolean program (QBP) [12], continuous and discrete optimization [13], and generalized clique graph [14,15]. In these approaches, the cost of pairing two detections in successive frames is computed from geometric or appearance cues of detection boxes. However, the appearance descriptors, which determine whether two detections belong to the same person, remained relatively simple.

Recently, deep learning algorithms have demonstrated significant improvement in image and video content representation. They provide rich feature representations that are more reliable than handcrafted features, which lead to improvement in many different computer vision tasks including image classification and object detection [16,17]. Motivated by this, we design a dual CNN–RNN for the multiple people tracking problem. Specifically, we first propose a convolutional neural network (CNN) that provides the similarity score of two detections. Next, a graph is built to get the primal tracklets (i.e. target trajectories), where the detections are the nodes and the edge costs are the computed cutting costs. Here, the data association task is cast to a lifted multicut (LM) problem which is solved here by Kernighan–Lin algorithm with Joins [18]. In order to handle long term occlusions and ID-switches, we propose a novel recurrent neural network (RNN) that is capable to predict the motion of targets. Furthermore, we use this information to merge several tracklets to build long term trajectories. The provided motion estimation also boils down the ID-switch problem where targets are crossing each other. In summary the three major contributions of this technique are:

1. a CNN for determining the pairwise detection affinity cost;
2. an RNN for nonlinear motion prediction of targets applied in both static and moving camera scenarios;
3. a tracklet stitching algorithm using motion patterns obtained by the RNN in order to handle long occlusions.

* Corresponding author.
E-mail address: maryam.babaee@tum.de (M. Babaee).

https://doi.org/10.1016/j.neucom.2019.08.008
0925-2312/

The rest of the paper is organized as follows: Section 2 reviews related works. The proposed methodology is explained in Section 3. Experiments and evaluation results are presented in Section 4. In the end, Section 5 provides a short summary and conclusion.

2. Related work

Multiple people tracking has been widely investigated in computer vision and pattern recognition. However, it becomes more challenging in presence of effective factors on tracking results such as illumination, crowded environment, moving camera, and target interaction. Most proposed approaches are based on the tracking-by-detection paradigm [10,19–21], where the tracking problem is cast into a data association problem between all detections extracted from the video by person detectors [22,23]. Having high accurate detections, tracking can be done by simply associating detections using their spatial overlap between frames [24]. Dealing with inaccurate detections, Sanchez-Matilla et al. [25] suggest to classify all detections to strong (certain) and weak (uncertain) detections. Strong detections are used for initialization and tracking, and weak detections are used only to support the continuation of an existing track when strong detections are missing. Chen et al. [26] extend the MHT algorithm by enhancing detection model that include detection-scene analysis and detection–detection analysis.

Tracking or data association can be performed either on individual detections [19,27] or on a set of confident short tracklets [28–31] which are generated by first performing a low level data association to group detections. A well-known representation of the tracking-by-detection paradigm is to present each detection as a node in a graph, where each edge represents the likelihood that connected detections belong to the same person. This data association problem can be formulated as a Conditional Random Field (CRF) inference [32], network flow optimization [33], global optimization based on maximum multi-clique [14], globally-optimal greedy algorithms [34], or subgraph decomposition [35].

As a challenge in a tracking problem, occlusion greatly affects the performance of tracking methods. In order to address this issue, some approaches use information from other views if they are available [10,36,37]. However, these methods are not applicable in single view tracking. Several efforts have been made to tackle the occlusion problem by target segmentation during tracking [38] or body joint tracking [2,39]. Milan et al. [2] cast a joint segmentation and tracking problem as a graphical model (i.e. Conditional Random Field). In this method, the super-pixels need to be extracted first which causes a high computational cost.

By learning discriminative feature representations, deep learning has greatly helped in many computer vision applications such as visual understanding [40,41], semantic image segmentation [42], and pedestrian detection [43]. In the context of tracking, several online tracking approaches utilize CNN in order to learn rich feature representation of targets instead of using heuristic and hand-crafted features [44–46]. In [44], the authors use target features obtained by an offline pre-trained CNN to do data association in an online manner, unlike [45] where a pre-trained CNN is tuned during tracking continuously to adopt the appearance of targets tracked in the observed frames. As a different problem formulation, Fan et al. in [47] designed a CNN to estimate the position and the scale of objects in the next frame using the observations of the current and previous frames. Recently, CNN has been explored for modeling the similarity between pairs of detections [48,49]. In [48], a Siamese CNN is proposed to learn spatio-temporal affinity between two image patches. The learned features are then combined with other contextual features using gradient boosting. In addition to the image patches in RGB format, the associated optical flows are also fed to this CNN. Tang et al. [49] exploit body part detections stacked with RGB person images as input to a CNN for measuring the similarity. Son et al. [50] model the appearance with temporal coherency by designing a quadruplet CNN. As a different network structure, Milan et al. propose an end-to-end Recurrent Neural Network (RNN) for the data association problem in online multi-target tracking [51]. They use RNNs for temporal prediction as well as track's birth/death determination in each frame. In [27], several RNN networks have been proposed to model the appearance, motion and interaction of targets. However, training several RNNs requires much more training data as well as computational resources. Bewley et al. [52] propose a simple and real time online tracking method by employing the Kalman filter (as a linear motion predictor) and the Hungarian algorithm (as data association). Wojke et al. [46] extend the work of Bewley et al. [52] by applying a deep association metric based on visual appearance. Despite real time tracking speed, online tracking approaches have a lower overall performance due to not seeing subsequent video frames at each time step. We believe that most ID-switches could be avoided by considering detections before and after a long occlusion.

In our work, the association problem is formulated as a clustering task by solving a minimum cost lifted multicut problem in an offline manner. A CNN is proposed to reason about the similarity of each pair of detections. This similarity measurement in cooperation with other simply computed affinities form the likewise potential for the graph optimization. In the end, by introducing a mechanism for path prediction, the more likely tracklets to show the same person are stitched together. The tracking method in [49] is the most similar approach to ours. It applies DeepCut [53] and DeepMatching [54] techniques in order to build the pairwise affinity. In contrast, our aim is to use the simple data input without help of other cues such as optical flow, pose, or body part detections.

3. Approach

The diagram of the proposed approach is depicted in Fig. 1. The input is a sequence of video frames. We utilized a person detector to extract the detection boxes from all frames. These detection boxes are fed into our CNN for training and ultimately computing the similarity of detections. We also extracted some geometric cues such as width, height, and position from detection boxes. Both CNN's output and geometric cues are considered as edge costs in the graph composition process. Here, a lifted tracking graph whose nodes are detections is constructed. This graph is optimized by Kernighan–Lin optimizer to obtain a set of primal tracklets. The proposed RNN uses the tracklets to estimate the motion of detections in successive frames. This estimation is used in order to (1) handle ID-switches and long-term occlusions, and (2) stitch several tracklets in order to build a long trajectory of a target. In the following, we will discuss each process in details.

3.1. Target matching using CNN

We present a CNN to measure the similarity of detections extracted from video sequences. Later, we use these measures to define a cost for each edge connecting a pair of detections (nodes) in the data association graph which will be explained later. A cost is to be paid if two detections with high probability of being the same person are clustered to different groups. To quantify how likely a pair of detections identify the same person, we propose a multi-layer deep CNN which takes two RGB detection images as input and then outputs the similarity score of the two detections in percentage (see Fig. 2). For the input of the CNN, the RGB image patches are first resized to a fixed size of 140×60 pixels and then stacked depth-wise to form a 6-channel data tensor.

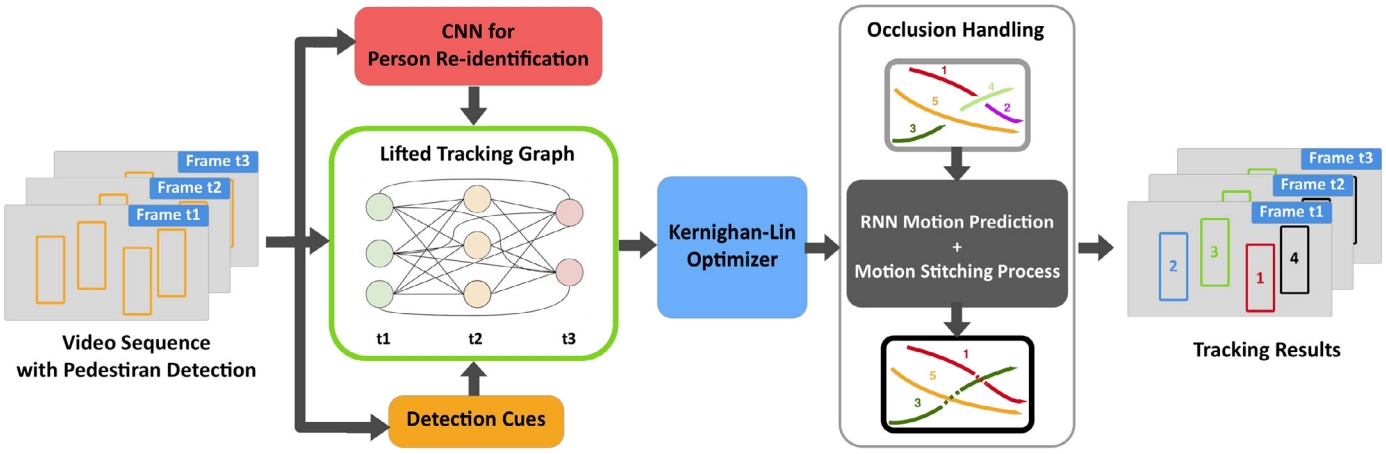

Fig. 1. The diagram of the proposed multiple people tracking algorithm.

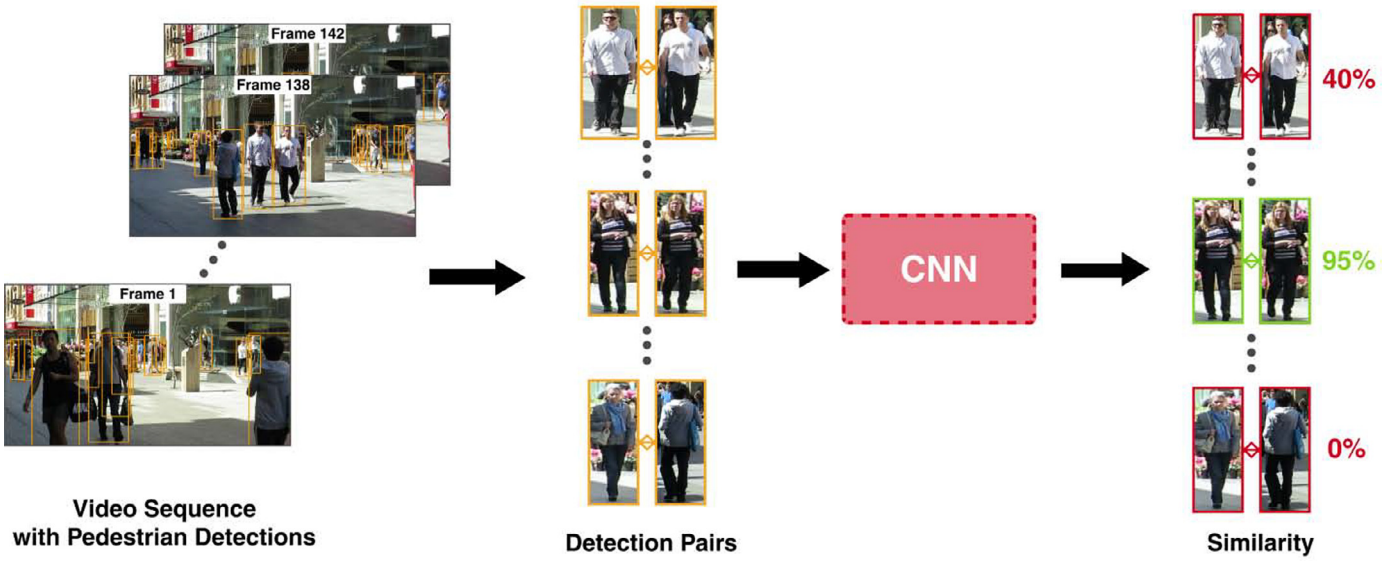

Fig. 2. Detections similarity computation using a CNN. The input of the network is detection pairs chosen from the video sequence, and output is the similarity score of each detection pair (video sequence from MOT Challenge 2016 dataset).

For training the network, we use the training data from MOT Challenge 2016 Benchmark. The CNN has 9 layers after the input layer. The input data is first processed by three convolutional layers, each of them followed by a ReLU nonlinear activation function. We employ stride with length two during the convolution to reduce the size of the images. Afterwards, there are four fully connected layers to capture correlations between features in distant parts of the images. To prevent over-fitting and improve overall performance, we used a dropout rate of 50% in all fully connected layers. The output of the last fully connected layer is fed into a binary softmax classifier which produces a distribution over the two class labels (first/second class indicates the detections belong to the same/different person(s)). The output of the first class, which indicates the similarity of the two input detections, is used in our tracking method. Fig. 3 illustrates the structure of the designed CNN.

3.2. Pairwise affinity measure in graph

Since the CNN can only model the appearance of the targets, we need to consider further constraints to define the edge costs in the data association graph, in case of mismatch errors from the network or two different targets with very similar appearances. In other words, we need to assign a larger penalty to those edges that have a high appearance similarity, while the position or the scale of the detection bounding boxes do not match. Moreover, in order to handle false positive detections, we incorporate a detection confidence into the edge costs. Here, we introduce our three additional edge costs: (1) position C_p, (2) height C_h and (3) detection confidence C_s. Each detection bounding box $v \in V$ has the following properties: (1) spatio-temporal location (t_v, x_v, y_v); (2) scale h_v; and (3) detection confidence s_v. Given two detections v and w connected by the edge $\{v, w\} = e \in E \cup E'$, we get a similarity score from the CNN denoted by C_e. The Euclidean distance of two detection bounding boxes in pixels is denoted by $dist_{vw}$, and fr denotes the frame rate of the video sequence. These three additional edge costs are described as follows:

Position:

$$dist_s = \frac{dist_{vw}}{\min(h_v, h_w)} \times \frac{fr}{abs(t_v - t_w)} \tag{1}$$

$$C_p = \begin{cases} 1, & if\ dist_s < d_{min} \\ 1 - \dfrac{dist_s - d_{min}}{d_{max} - d_{min}}, & if\ d_{min} \le dist_s \le d_{max} \\ 0, & if\ dist_s > d_{max} \end{cases}. \tag{2}$$

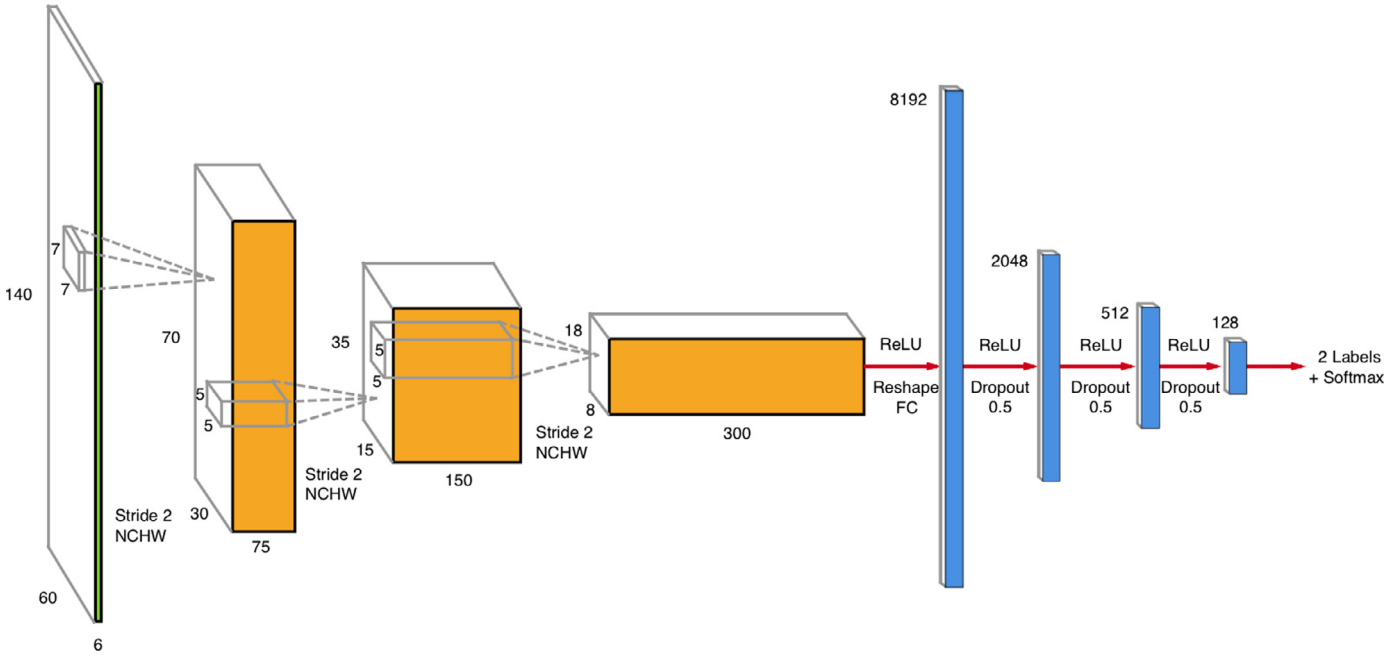

Fig. 3. The structure of the proposed CNN for similarity computation of two detections.

Height:

$$height_s = \frac{abs(h_v - h_w)}{\min(h_v, h_w)} \times \frac{fr}{abs(t_v - t_w)} \tag{3}$$

$$C_h = \begin{cases} 1, & if\ height_s < h_{min} \\ 1 - \dfrac{height_s - h_{min}}{h_{max} - h_{min}}, & if\ h_{min} \le height_s \le h_{max} \\ 0, & if\ height_s > h_{max} \end{cases}. \tag{4}$$

Detection score:

$$score = min(s_v, s_w) \tag{5}$$

$$C_s = \begin{cases} 1, & if\ score \ge s_{threshold} \\ score, & if\ score < s_{threshold} \end{cases}. \tag{6}$$

The final edge cost is:

$$C_t(v, w) = C_e * C_p * C_h * C_s \tag{7}$$

In Eqs. (1) and (3), the Euclidean distance and height difference are first normalized by the height of the detection boxes. Since the height is related to the distance of the target from the camera, it can be considered as a scale factor. If a target is close to the camera, the changes in distance or height would also be large and vice versa. This normalization step is especially helpful for relatively low camera position. Then the position and height changes are divided by the frame gap and multiplied by the frame rate. As a result, the $dist_s$ and $height_s$ are the normalized change rate of the distance and height. The parameters $\{d_{max}, d_{min}, h_{max}, h_{min}\}$ are input parameters for each video sequence and are calculated using all video sequences in MOT2016 training dataset. In the final implementation, we prepared two different settings for these parameters; one for static cameras and the other for moving cameras, because the movement and height changes of moving cameras are larger than static cameras on average.

As for detection cost, if the minimum value of the two confidence scores is less then a predefined threshold, then we assign a lower penalty to the edge cost. The detection confidence score depends on the performance of the person detector, which might be affected by light condition, contrast of the image frame, crowdedness of the scene etc.

3.3. Graph decomposition by lifted multicut

For tracking multiple people, we use the popular tracking-by-detection framework, which utilizes a person detector to generate detection hypotheses for a video sequence. Then the tracking problem is basically simplified to an association task between detection hypotheses across video frames. In our framework, we formulate this problem as a minimum cost lifted multicut problem. Here each node of the graph represents one single detection and edges link detections over video frames. A feasible solution for such a multicut problem clusters detections jointly over time and space. Therefore, the number of the tracks does not need to be specified or constrained. Here, the high edges costs encourage the underlined nodes to be labeled as the same target. That means, this edge should not be cut for decomposing the graph. The optimization process partitions the graph into distinct components, each representing one person's track.

3.3.1. Definition of lifted multicut problem

Minimum Cost Lifted Multicut [18] is an optimization framework for multi-labeling problems and partitioning graphs. There is a one-to-one relationship between the decompositions of the graph and labeling of the nodes in its feasible solutions. For any undirected graph $G = (V, E)$, any set of lifted edges $E' \subseteq \binom{V}{2} E$ and edge cost function $c : E \cup E' \to \mathbb{R}$, we can define a sample of the Minimum Cost Lifted Multicut Problem which is formulated as:

$$Y = \underset{y \in Y_{EE'}}{\operatorname{argmin}} \sum_{e \in E \cup E'} c_e y_e \tag{8}$$

where set $Y_{EE'} \subseteq \{0, 1\}^{E \cup E'}$, whose elements $y \in Y_{EE'}$ are 01-labelings of all edges $E \cup E'$, are feasible solutions. Additionally, this optimization must hold the following three conditions:

$$\forall vw \in E \cup E' P \in vw - paths(G) : \ y_{vw} \le \sum_{e \in P} y_e \tag{9}$$

$$\forall vw \in E \cup E' C \in vw - cuts(G) : \ 1 - y_{vw} \le \sum_{e \in C} (1 - y_e) \tag{10}$$

Here, the inequality (10) and (9) guarantee that the lifted edge, connecting nodes v and w, is labeled 0, if and only if there exists

Fig. 4. Long-term occlusion. An example in video sequence MOT16-08 from frame 45 to frame 114, not handled by the lifted graph optimization. ID 4 and ID 44 belong to the same person but are occluded by ID 1. Similarly, ID 27 and ID 76 belong to the same person but are occluded by ID 44.

a path in G between these two nodes along which all edges are labeled 0.

$$\forall C \in cycles(G)\ \forall e \in C:\ y_e \leq \sum_{e' \in C \setminus \{e\}} y_{e'} \tag{11}$$

These cycle constraints (11) are transitivity constraints which guarantee that a feasible solution y well-defines a clustering of the graph nodes into tracks. More details could be found in [18].

3.3.2. Optimization algorithms for lifted multicut problem

To accelerate the graph optimization and node labeling process, we first use an adaptation of greedy agglomeration to initialize the label of all the nodes. The algorithm is also called Greedy Additive Edge Contraction (GAEC) and was introduced in [18]. Kernighan–Lin algorithm with joins (KLj) [18] is an extension of the Kernighan–Lin algorithm [55]. It starts from an initial decomposition provided as input. We use the output of GAEC as the initial decomposition. Each iteration tries to refine the current decomposition (i.e. a decomposition with lower objective value) by one of the following transformations: (1) moving nodes between two neighboring components, (2) moving nodes from one component to an additional, newly introduced component, (3) joining two neighboring components [18].

After the optimization, we generate tracks from detection clusters. The multicut optimization clusters the detections jointly over space and time for each person. First in each frame, we obtain the location and scale of each target by calculating the location and scale average of all detections that belong to that target. Then, we connect these averages across all frames to obtain a smooth trajectory for each target.

3.4. Motion prediction

When the lifted edges fail to handle some occlusions, either due to the limitation of the CNN or the high crowdedness of the video sequences, we would need the motion information of the targets in order to tackle the long term occlusion problem. In Fig. 4, a long term occlusion case which is not handled by the multicut graph is shown. We use the motion information of primal tracklets gained from multicut optimization to stitch the tracklets that belong to the same person. As it will be shown in the next section, it reduces the number of false negatives and ID-switches and thus improves the overall tracking performance. The idea of reducing ID-switch happening during occlusion, using motion information, is illustrated in Fig. 5. At each time, we select two tracklets such that the first tracklet finishes sooner than the beginning of the second tracklet. Here, we denote the frame gap between these two tracklets as Δt, and the detection box of T_i at time step t is denoted as $T_i\{t\}$. Then by adopting our motion prediction algorithm, we calculate the predicted bounding boxes of the first tracklet at time step $t + \Delta t$, denoted as $P_1\{t + \Delta t\}$, and the predicted bounding boxes of the second tracklet at time step t, denoted as $P_2\{t\}$. The two tracklets will be recognized as the same target and stitched together in the final solution if they satisfy the following condition:

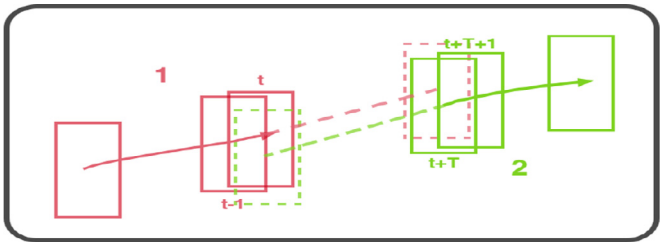

Fig. 5. Motion prediction of two tracklets; estimation of the bounding box's position of tracklet 1 at time $t + \Delta t$ and the bounding box of tracklet 2 at time t.

$$overlap_1 + overlap_2 \geq stitch\_thr \tag{12}$$

where:

$$overlap_1 = \frac{T_1\{t\} \cap P_2\{t\}}{T_1\{t\} \cup P_2\{t\}} \tag{13}$$

$$overlap_2 = \frac{T_1\{t + \Delta t\} \cap P_2\{t + \Delta t\}}{T_1\{t + \Delta t\} \cup P_2\{t + \Delta t\}} \tag{14}$$

Fig. 6. Occlusion handling on PETS09-S2L1 sequence; (first row) raw detections, (second row) output of lifted multicut optimization, (third row) tracking result using motion prediction.

Here, if the intersection over union area of the predicted and tracked detection bounding boxes for both tracklets is over a threshold, then we merge the two tracklets together as a new tracklet. If there are multiple tracklets that can be stitched to T_i, then we choose the one with the largest $overlap_1$ and $overlap_2$. In order to predict the position and scale of detection in the next frame, we consider a nonlinear motion prediction. In Fig. 6, we show the tracking results on PETS09-S2L1 sequence with and without the occlusion handling using motion prediction. The raw detections are shown in the first row of the figure. The outputs of the LM optimization are presented in the second row, noticing the person in blue is temporarily hidden by the person in black in frame 131 and 132, and the lifted edges fail to overcome this occlusion during the optimization. In the third row, the final results after occlusion handling are shown. It can be seen that the person in blue is tracked correctly before and after the occlusion.

Regarding motion prediction, there are some challenges we need to overcome: (1) noisy detections which can lead to nonlinear predictions of the target motion even when the actual motion is linear; (2) When the video is taken from an eye-level camera position, the width of the bounding box changes to adapt the walking pose of the person. Handling this issue is not straightforward, since the pattern with which the width changes, is different for different sizes of the bounding boxes, walking speed and even walking habits of the pedestrians. We illustrate one example of this observation in Fig. 7. This is why we designed an RNN to learn how to generate the future walking pattern given the past walking pattern of the targets.

3.4.1. Recurrent neural network for nonlinear motion prediction

In order to check whether two tracks belong to the same target, we need accurate predictions of the future or past positions and scales of the detection bounding boxes. Hereby, we design an RNN that is capable of predicting the position $\{X, Y\}$ and scale factor $\{W, H\}$ in the next time steps, based on the bounding box parameters at current and the bounding box sequence in all the past time steps.

Fig. 8 illustrates the unrolled structure of our RNN for motion prediction. As input of the RNN at each time step, we use the vector $\{\Delta X_i, \Delta Y_i, \Delta W_i, \Delta H_i\} = \{X_i - X_{i-1}, Y_i - Y_{i-1}, W_i - W_{i-1}, H_i - H_{i-1}\}$ which represents the difference between adjacent detection bounding boxes. There is a fully connected layer without activation function (as a linear layer) between the input layer and RNN cell. The linear layer output is a vector of size M. Using this linear layer, we expand the size of the input feature vector to make the training process more effective. The expanded input vector then goes through an RNN cell with internal size of N. Two outputs of the RNN cell are illustrated by two outgoing arrows from each RNN cell in Fig. 8 (both are vectors of size N). One is the output vector of the cell and the other is the hidden state of the RNN. As for the output vector of the RNN, we use another linear layer to bring the vector size down to 4. This vector shows the bounding box changes at the next time step. At training time, we shift all the training input sequences one time step forward and use them as our training labels. The labels are then compared with the final output of the network using L2-Loss function. The loss is then minimized to update the weights and biases of the network through the back-propagation algorithm.

The main reason for using the *changes* of the bounding boxes parameters instead of the bounding box parameters directly, is that the raw bounding box parameters are non-stationary. To illustrate this, we plot the width of the bounding boxes of the person shown in Fig. 7 on the left side of Fig. 9. We can see there is a decreasing trend in the data. This means there is a

Fig. 7. The width of the detection bounding box changes periodically according to the changes of the walking pose (video sequence MOT16-10).

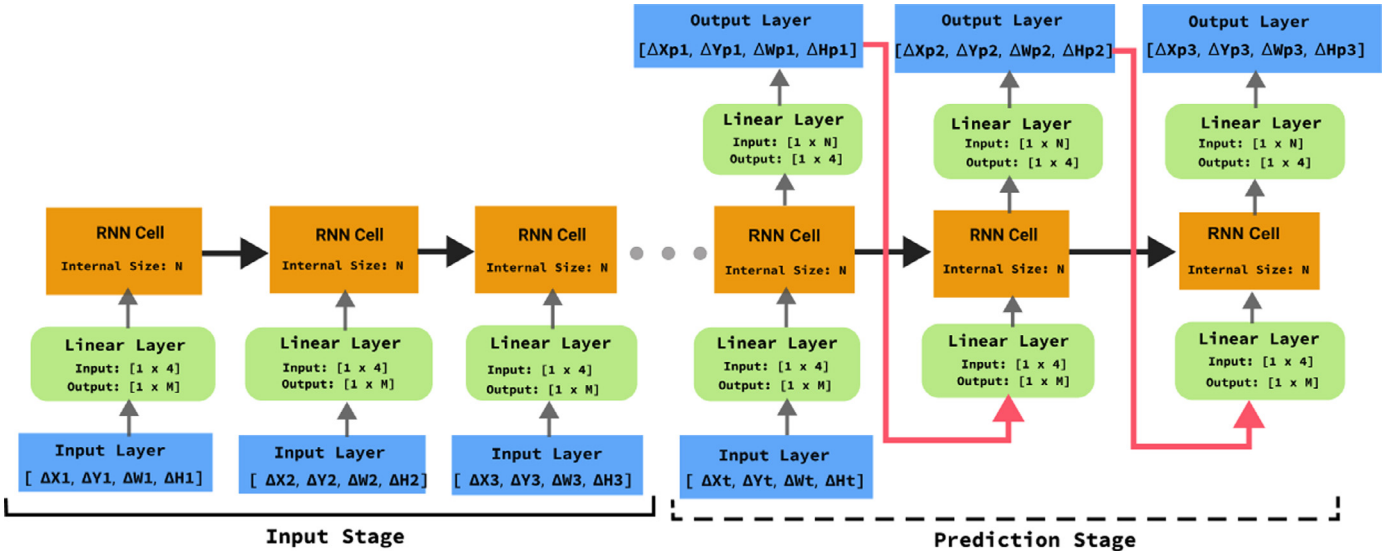

Fig. 8. The structure of the RNN for nonlinear motion prediction. At input stage, we input all the available bounding boxes into the network. At the prediction stage, we use the output of previous time step as the input of the current time step while maintaining the hidden state from the beginning.

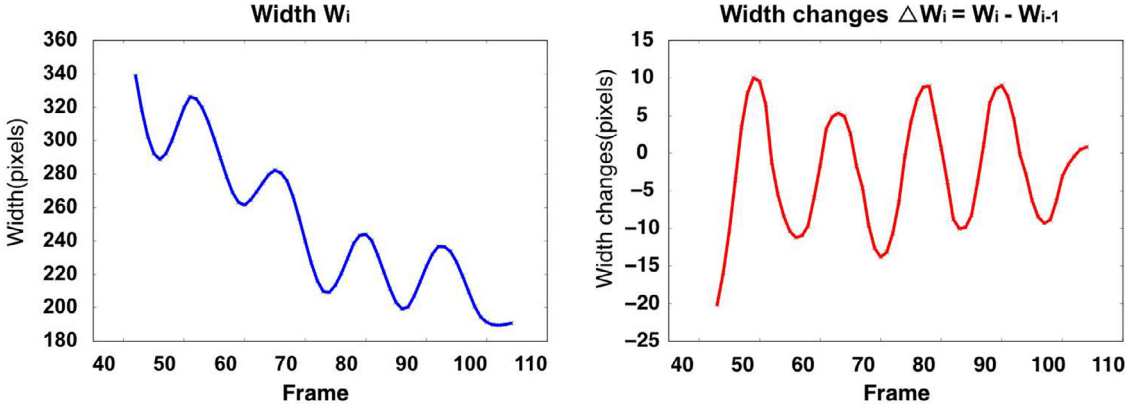

Fig. 9. Non-stationary data vs. stationary data; (Left) width W_i of the detection boxes in Fig. 7 from frame 42–105. (Right) width differences of the neighboring detection boxes $\Delta W_i = W_i - W_{i-1}$.

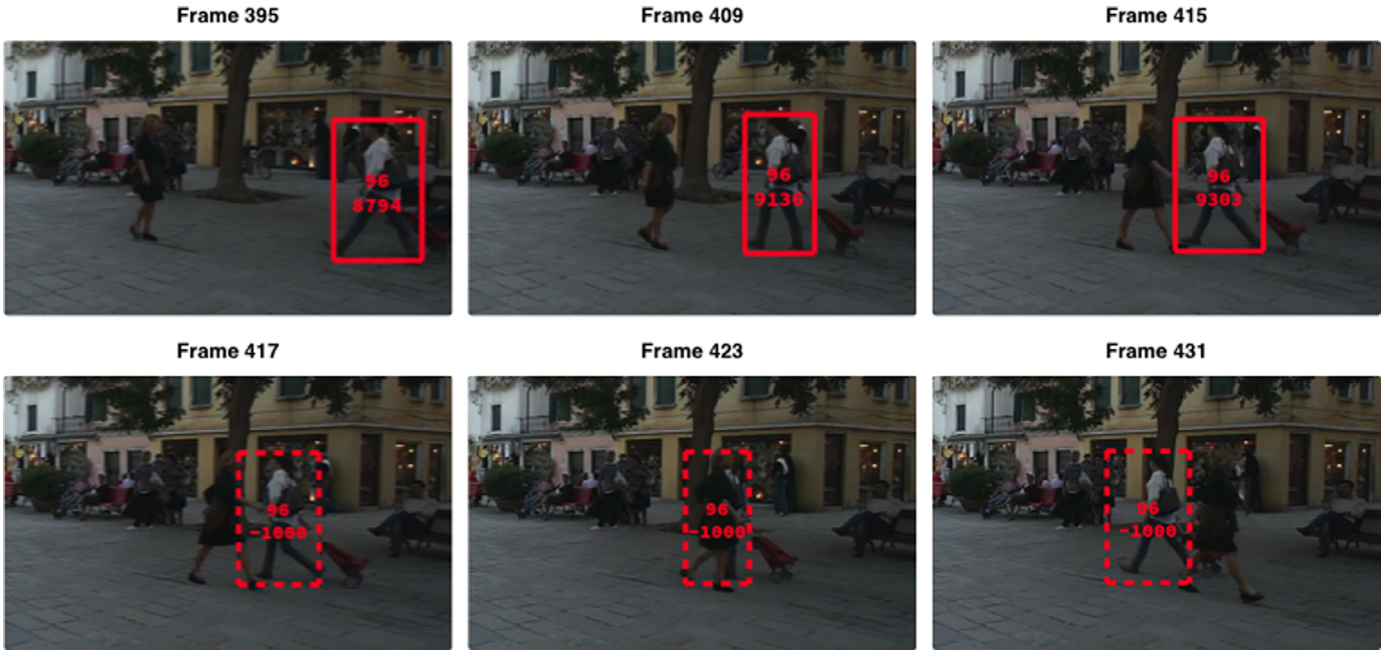

Fig. 10. Motion prediction using RNN in video sequence MOT16-01 from frame 395 to frame 431; (first row) actual track, (second row) bounding boxes with dashed line are the predicted detection bounding boxes by the RNN.

structure in the data that is dependent on time and thus the data is non-stationary. We found that non-stationary data is hard to model for recurrent neural networks and it makes the convergence speed of the training process extremely slow. Therefore, in order to make our input data stationary and remove the increasing or decreasing trends, we consider the difference in input data by subtracting each bounding box parameter at time step $(t-1)$ from the corresponding one at current time step t. The data plot of the differences is shown on the right side of Fig. 9. Our experiments show that after removing the time dependent structure from the data, the RNN has much better convergence properties, as well as much more accurate and robust prediction outputs.

At inference time, the RNN is to predict future bounding boxes in the next few steps after the end of each track(let). We split the process into two stages. At the first stage, which is the input stage, we just input all the existing bounding boxes, time step by time step, into the network. All the outputs of the first stage are ignored. The second stage, which is the prediction stage, begins from the last detection box of the track(let). At this stage, the output of the previous time step is used as the input of the current time step. Since the hidden state contains all the information about the past sequence, we keep transferring the hidden state from the beginning. The prediction stage stops after a predefined step, because as the prediction step increases, the predictions become less reliable.

We found that the position can be seen as a scale factor for predicting the width and the height. For instance, if the RNN detects some significant changes of position, this indicates the changes of the width and the height should be increased in the next time step. Therefore, adding position parameters to the RNN structure can improve the overall prediction performance. In Fig. 10, we illustrate one prediction output of the RNN from the MOT16-01 test data. On the first row, we show the actual tracklet of the target. On the second row, the bounding boxes with dashed lines are the predicted ones by the RNN. There are two numbers in the bounding boxes. The first number is the global tracking ID which should not change during the life span of the target in all video frames, and the second number is the global detection ID. Since the predicted bounding boxes do not exist in the original input detections set, we assign -1000 to the predicted bounding boxes number. From the result, it can be seen that the predictions are fairly accurate. Note that the bounding boxes in the middle on both rows are narrower than the bounding boxes on two sides.

3.5. Tracklet stitching

We use the motion prediction model to stitch the tracklets belonging to the same target. The first step in this process is to use the motion predictor (the RNN) to expand the length of the tracklets which are generated by the lifted multicut (LM) optimization. More specifically, we use the RNN to predict the future and past bounding boxes of each tracklet for some time steps. After the tracklet extrapolation step, we check every possible tracklet pair for stitching. For each tracklet pair, a stitching score is calculated, where S_{ij} denotes the score for stitching T_i and T_j together. This value indicates how likely these two tracklets are to belong to the same person. To calculate this value, the formulas (13) and (14) are applied. However, this value is not the final score. These two tracklets must satisfy two conditions, otherwise we set the score to zero. These conditions are: (1) The last time step of the first tracklet must be earlier than the first time step of the second tracklet; (2) In order to make sure the stitching score is valid, it must be above a certain threshold called the stitching threshold, which is a function of the frame gap between the two tracklets.

After calculating the stitching scores, we then go through an iterative process, in which we find the tracklet pair S_{ij} with the highest stitching score. Then these tracklets are merged to form a single tracklet. In order to fill up the frame gap between the tracklets, we interpolate the missing detections by using the predicted bounding boxes from the RNN. For the new tracklet ID, the ID of the first tracklet T_i is used. We then take all the past predictions of the first tracklet and all the future predictions of the second tracklet as the past and future bounding box predictions, respectively, for the new tracklet. We keep iterating this stage until no valid tracklet pair remained for stitching. We use the results of this process as our final tracking results.

4. Results and discussion

4.1. Datasets

We use the Multiple Objects Tracking (MOT) Challenge Benchmark 2016 [60] to test the overall performance of our algorithm. This benchmark contains 7 challenging video sequences including 5316 frames in unconstrained environments, filmed with both static and moving cameras. For all MOTChallenge test sequences, the POI detector [3], which uses a modified Fast Region-based CNN (Fast R-CNN) [61], is utilized to produce person detections. For comprehensive evaluation, we also used the MOT15 [62] and MOT17 benchmarks. MOT17 contains the same video data as MOT16, but it provides detections from different detectors. MOT15 benchmark includes video data at more divers frame rates.

4.2. Evaluation metrics

We report the performance of our tracking method in terms of CLEAR MOT metrics [63] including Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP), Mostly Tracked (MT, >80% overlap), Mostly Lost (ML, <20% overlap) and Partly Tracked trajectories (PT). MOTA incorporates three error types; the number of False Positives (FP), the number of False Negatives (FN), and the number of ID-switches (IDs). The FAF metric is the average number of false alarms per frame, and the FM metric shows the total number of times a trajectory is fragmented during tracking.

4.3. Implementation details

For training the CNN, we adopted the Adam optimizer to minimize a binary cross entropy loss function. In each epoch, we first randomly shuffled the order of the image pairs and collected 100 pairs of images as one mini-batch. Layer weights were initialized with zero mean normal distribution using the number of fan-ins as variants. The learning rate was initialized as 10^{-4} and decreased every 4 epochs by a factor of 10. To avoid over-fitting, we validated the network every 1/3 epoch. The network was trained on NVIDIA GTX Titan X. The training speed was 300–400 examples/s, and testing speed (only forward path) was 1100 examples/s. The training process converged after 10 epochs. For the evaluation of the predictions, two detections are assumed to be from the same target, if the similarity probability is larger than 0.5. Based on this calculation, the peak test accuracy stays at 95.8%.

The ground truth data of the MOT16 training set was used to generate training image pairs for the CNN. More specifically, three types of pairs are generated: (1) Detection pairs from the same person by checking the ground truth ID of detections in every next three frames. These pairs are labeled with 1 as positive pairs; (2) Detection pairs from different persons, same as the first case, by looking at the ground truth ID of the detections. This type of detection pairs are labeled with 0 as negative pairs; (3) Detection pairs from one true target and one false positive detection (labeled with 0 as negative pairs). The false positive detections are either randomly selected from the image area with given aspect ratio 14:6 from the last 10 frames that do not overlap with any ground truth detections, or some non-person objects such as cars and obstacles (note that in the MOT16 ground truth data, there are some available annotated non-person targets). For validation, we used the MOT 2015 training set (we excluded all the overlapping video sequences with MOT 2016 training set). In the end, there are 638,091 positive detection pairs and 1,304,471 negative detection pairs for training. For validation, there are 69,339 positive detection pairs and 76,237 negative pairs in total.

Table 1
Stitching score threshold as function of the frame gap in motion stitching process.

Frame gap	0–10	11–25	26–45
Stitching score threshold (static camera)	1.15	0.95	0.75
Stitching score threshold (moving camera)	1.0	0.85	0.75

The association graph is constructed by connecting detections over a certain frame gap. This is essential in order to track people in case of short occlusion. However, the pairwise similarity appearance feature becomes less reliable if the frame gap increases. Thus, we consider a maximum frame gap depending on the frame rate of the input video sequence when we construct the graph. In all experiments, we connect the detection nodes that are at most 10 frames apart for a video of frame rate less then 30 FPS. More precisely, the following relation is used to calculate the maximum frame gap g:

$$g = 10 \times \frac{fr}{30}. \tag{15}$$

The stitching score threshold and corresponding range of the frame gap used in the stitching process are presented in Table 1. We decreased the threshold as the frame gap increases because, as the frame gap becomes larger, the motion prediction becomes less reliable. We used two sets of thresholds, one for static cameras and one for moving cameras. In moving camera scenarios, the motion of the track is hard to predict, so the stitching conditions must be relaxed. The thresholds listed in Table 1 are fine-tuned on the training data of the MOT2016 benchmark. If the stitching score is below the threshold, the score is set to zero.

4.4. Benchmark evaluation

We submitted our tracking results named LM_NN_16 to MOTChallenge 2016,[1] where our results achieved third place in terms of MOTA, compared to the best performed algorithms in MOT16. In Table 2, we compare our tracking method with other algorithms tested on MOT16 benchmark. The arrows beside the evaluation metrics in the tables indicate weather higher(↑) or lower(↓) values are more desirable. In the LM_NN_16 version, we adopted both the CNN and the RNN to get our best result. While in the LM_CNN, the RNN is not used which results in lower MOTA and higher ID-switches. From this table, it is clear that our occlusion handling can successfully reduce ID-switches by about 29%, compared to the LM_CNN. It is also effective in improving MOTA and MOTP by 1.8% and 0.8%, respectively. Moreover, the number of track fragments decreases 30% by joining tracklets of the same track. Evidently, this indicates that the proposed occlusion handling (tracklet stitching) algorithm is beneficial in improving the results by a large margin. Fig. 11 shows qualitative results of our tracking algorithm on the MOT16 benchmark.

We also evaluated our method on MOT17 benchmark (see Table 5). In this challenge, the detections of three different detectors namely DPM [69], FRCNN [70] and SDP [71] are provided. By comparing to other published methods, our method is the best in terms of MOTP, FAF, FP, and Frag. The number of false positives and fragmentation is nearly 50% lower, compared to the top ranked method (FWT). Regarding the number of ID-switches, our method takes the second place.

[1] https://motchallenge.net/results/MOT16/.

Fig. 11. Visual tracking results of MOT16 testing dataset using POI detections, bounding boxes illustrated using dashed line are interpolated during motion stitching process, first number in bounding box is the global tracking ID, second number is the global detection-ID.

Table 2
Tracking results on the MOT16 benchmark. The complete table of results can be found on the MOTChallenge website. Methods marked with ∗ use detections provided by Yu et al. [3].

Tracker	MOTA (%)↑	MOTP (%)↑	FAF↓	MT (%)↑	ML (%)↓	FP↓	FN↓	IDs↓	FM↓	Hz↑
HT_SJTUZTE [56]	**71.3** ± 11.4	79.3	1.6	46.5	19.5	9238	**42,521**	617	743	29.0
LMP_p [49]∗	71.0 ± 12.9	**80.2**	1.3	**46.9**	21.9	7880	44,564	434	**587**	0.5
KDNT [3]∗	68.2 ± 12.9	79.4	1.9	41.0	19.0	11,479	45,605	933	1093	0.7
MCMOT_HDM [57]	62.4 ± 10.6	78.3	1.7	31.5	24.2	9855	57,257	1394	1318	34.9
NOMTwSDP16 [58]	62.2 ± 11.0	79.6	0.9	32.5	31.1	5119	63,352	**406**	642	3.1
DeepSORT-2 [46]∗	61.4 ± 10.6	79.1	2.2	32.8	**18.2**	12,852	56,668	781	2008	17.4
SORTwHPD16 [59]	59.8 ± 10.3	79.6	1.5	25.4	22.7	8698	63,245	1423	1835	**59.5**
EAMTT [25]	52.5 ± 11.4	78.8	**0.7**	19.0	34.9	**4407**	81,223	910	1321	12.2
LM_CNN∗	67.4 ± 12.8	79.1	1.7	38.2	19.2	10,109	48,435	931	1034	1.7
LM_NN_16∗	69.0 ± 12.8	79.9	1.0	37.2	22.0	6213	49,675	668	730	1.5

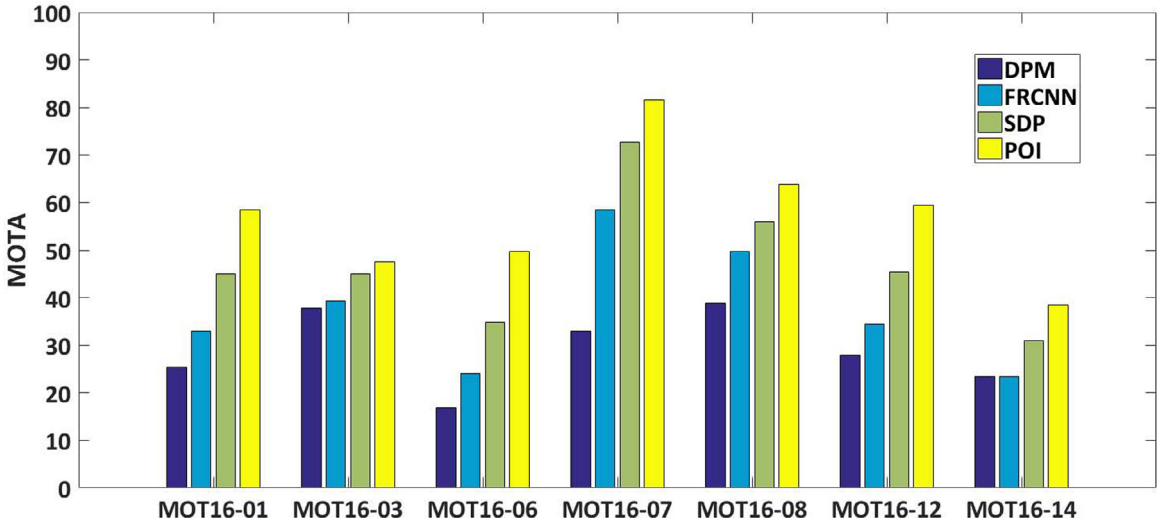

Fig. 12. The accuracy of the proposed tracking method for each sequence of MOT16 using different person detectors namely POI, SDP, FRCNN and DPM.

4.5. Ablation study

4.5.1. Analysis of detection pairwise similarity

The pairwise affinity measure in our algorithms consists of four elements; CNN score, height, position, and detection confidence score (det_score). Table 6 shows the contribution of each element. Since the MOT16 training data was used for training the CNN, we conducted this experiment on the unseen MOT15 training set. It can be seen that the CNN score is the biggest contributor to the accuracy, compared to the other elements. However, the other elements are also effective in enhancing the tracking performance.

Our CNN model, unlike a Siamese network which is a common network for human re-identification, gets the concatenated image pairs as the input. We compared our CNN with other models applied by Tang et al. [49], namely ID-Net, SiameseNet, StackNet, and StackNetPose. ID-Net is based on the VGG-16 Net [72]. After extracting the identity features from the last fully connected layer, they use Euclidean distance to decide whether the two detections belong to the same person. SiameseNet, which has a Siamese architectures, has twin CNNs followed by two fully connected layers. StackNet has the structure of normal CNN, but the image detections are stacked together along RGB channel to form the input tensor. StackNetPose is has a similar structure to StackNet, but it fuses the body part correspondences between the two images with RGB images as input. For a fair comparison, we retrained the model with the same training data they used. Table 7 shows the performance of each network. The accuracy results of the models are copied directly from the published paper. We used the same train and test dataset to evaluate these structures. In this experiment, our network outperforms the Siamese network achieving 96.1% accuracy in comparison to 84% accuracy for Siamese. This shows that joint processing of the image pairs from the first layer of the network leads to a better performance, consistent with the findings in [73].

We also evaluated the performance of our proposed method given detections from different detectors. Fig. 12 demonstrates the accuracy (MOTA) of the proposed tracking algorithm on detection inputs from different detectors, namely POI [3], FRCNN, SDP, and DPM. Since our proposed algorithm falls into the tracking-by-detection group, the final tracking results will obviously be improved by having a more accurate detection hypothesis as an input.

4.5.2. Analysis of RNN

We tested both the Gated Recurrent Unit (GRU) and the Long Short-Term Memory (LSTM) as the cell structure in the RNN shown in Fig. 8. The internal size of the cell is $N = 1024$ and the output vector size of the linear layer is $M = 100$. From Fig. 13, it can be seen that in our specific application both GRU and LSTM have an almost identical performance. After 15 training epochs, the difference of the test loss between LSTM and GRU is very low, but using GRU leads to a better convergence speed at the early stage of the training. Since the prediction performance of LSTM and GRU was identical in our bounding box prediction task, we chose GRU as the cell structure.

In order to evaluate quantitatively the performance of the motion prediction (with purpose of occlusion handling), we report the results of our tracking approach with and without using the RNN on the MOT2016 dataset. In Table 4, we show two important evaluation metrics which essentially determine the tracking

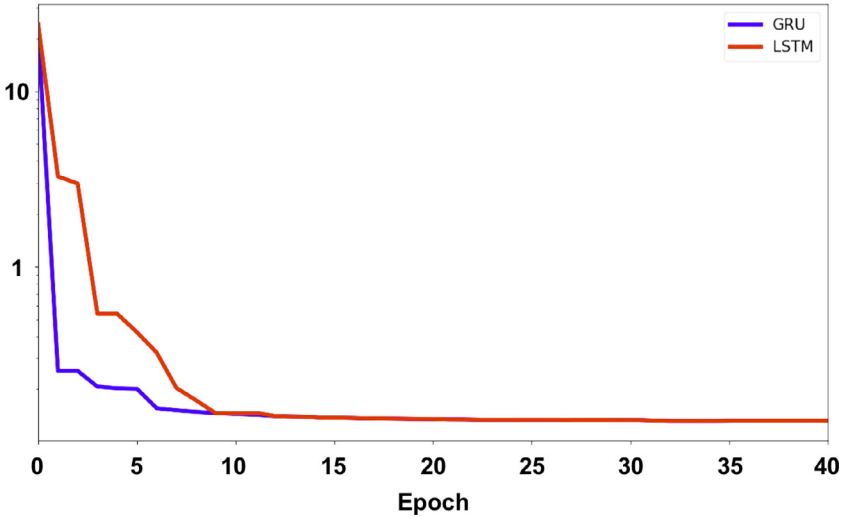

Fig. 13. Test loss of the RNN with GRU and LSTM cells.

Table 3
Detailed results on the MOT16 test sequences.

Sequence		MOTA (%)↑	MOTP (%)↑	FAF↓	MT (%)↑	ML (%)↓	FP↓	FN↓	IDS↓	FM↓
MOT16-01	LM_CNN	51.9	78.8	2.1	47.8	4.3	964	2074	38	40
	LM_NN	58.4	79.3	0.9	43.5	13.0	391	2244	24	27
MOT16-03	LM_CNN	79.7	78.9	3.1	69.6	5.4	4638	16,176	367	415
	LM_NN	81.6	79.9	1.5	65.5	6.1	2191	16,780	227	245
MOT16-06	LM_CNN	63.9	81.5	0.5	42.1	23.5	545	3526	95	116
	LM_NN	63.8	81.1	0.4	42.5	24.0	534	3543	96	128
MOT16-07	LM_CNN	59.5	80.1	1.5	40.7	5.6	748	5750	112	118
	LM_NN	59.5	80.4	1.2	37.0	5.6	599	5904	101	105
MOT16-08	LM_CNN	37.1	80.6	1.6	19.0	25.4	1025	9381	127	140
	LM_NN	38.5	81.4	1.4	23.8	33.3	856	9363	66	71
MOT16-12	LM_CNN	47.3	80.3	0.6	22.1	36.0	583	3743	44	43
	LM_NN	47.6	80.7	0.5	20.9	43.0	449	3867	32	31
MOT16-14	LM_CNN	48.4	76.8	2.1	18.3	21.3	1606	7785	148	162
	LM_NN	49.7	77.2	1.6	17.1	25.0	1193	7974	122	123

Table 4
Tracking results of the proposed method with and without occlusion handling (OH) step, comparison of the sum of false positives and false negatives and the number of ID-switches on MOT2016 train set.

	MOT16-02	MOT16-04	MOT16-05	MOT16-09	MOT16-10	MOT16-11	MOT16-13
FP+FN	11,772	14,265	3096	1669	4777	2701	4403
FP+FN **with OH**	11,543	14,244	3080	1392	4700	2581	4189
IDS	139	173	71	46	160	52	243
IDS **with OH**	81	85	45	24	100	24	152

Table 5
Tracking results on the MOT17 benchmark. The complete table of results can be found on the MOTChallenge website.

Tracker	MOTA (%)↑	MOTP (%)↑	FAF↓	MT (%)↑	ML (%)↓	FP↓	FN↓	IDs↓	FM↓	Hz↑
FWT [64]	**51.3** ± 13.1	77.0	1.4	21.4	**35.2**	24,101	247,921	2648	4279	0.2
jCC [65]	51.2 ± 14.5	75.9	1.5	20.9	37.0	25,937	247,822	**1802**	2984	1.8
MHTDAM [11]	50.7 ± 13.7	77.5	1.3	20.8	36.9	22,875	252,889	2314	2865	0.9
EDMT17 [26]	50.0 ± 13.9	77.3	1.8	**21.6**	36.3	32,279	**247,297**	2264	3260	0.6
PHD_GSDL17 [66]	48.0 ± 13.6	77.2	1.3	17.1	35.6	23,199	265,954	3998	8886	6.7
SAS_MOT17 [67]	44.2 ± 12.2	76.4	1.7	16.1	44.3	29,473	283,611	1529	2644	4.8
EAMTT [25]	42.6 ± 13.3	76.0	1.7	12.7	42.7	30,711	288,474	4488	5720	**12.0**
GMPHD_KCF [68]	39.6 ± 13.6	74.5	2.9	8.8	43.3	50,903	284,228	5811	7414	3.3
LM_NN_17	45.1 ± 13.3	**78.9**	**0.6**	14.8	46.2	**10,834**	296,451	2286	**2463**	0.9

performance. One is the sum of false positives and false negatives, and the other is the number of ID-switches.

Table 3 presents the detailed tracking results for both LM_NN and LM_CNN on each video sequence of MOT16 test set. Overall, it can be seen that utilizing motion cues improves the overall tracking performance by increasing MOTA. ID-switches reduce for almost all the sequences, with the maximum reduction of 48% for the MOT16-08. The sequences MOT16-08, MOT16-01 and MOT16-03 all have a maximum ID-switch drifting. Similarly, track fragmentation drops for these three sequences significantly. These three videos have been recorded by a static camera. Although there is improvement in moving camera scenarios, it is not as remarkable as in the static camera scenarios. In order to evaluate the performance of the RNN for motion prediction, we feed 30 frame long tracklets to the RNN and predict the detection boxes for the next 20 frames. Since we need the ground truth for comparison and

Table 6
The contribution of each element of the pairwise affinity measure in tracking results of MOT15 train dataset.

Affinity	MOTA (%)↑	MOTP (%)↑	FAF↓	MT (%)↑	ML (%)↓	FP↓	FN↓	IDs↓	FM↓
Height + position + det_score	36.8	72.6	**0.65**	11.4	48.1	**1889**	7143	265	364
CNN	51.2	73.5	0.78	28.7	34.6	2248	4820	113	233
CNN + det_score	51.1	73.2	0.79	28.8	**34.5**	2275	4798	112	235
CNN + height	51.7	73.5	0.77	29.1	35.6	2225	4764	109	224
CNN + position	53.9	73.3	0.75	31.2	34.9	2162	**4536**	90	202
CNN + height + position	53.3	73.5	0.77	31.1	35.2	2226	4539	99	204
CNN + height + position + det_score	**54.0**	**73.6**	0.74	**31.1**	35.2	2132	4543	**89**	**200**

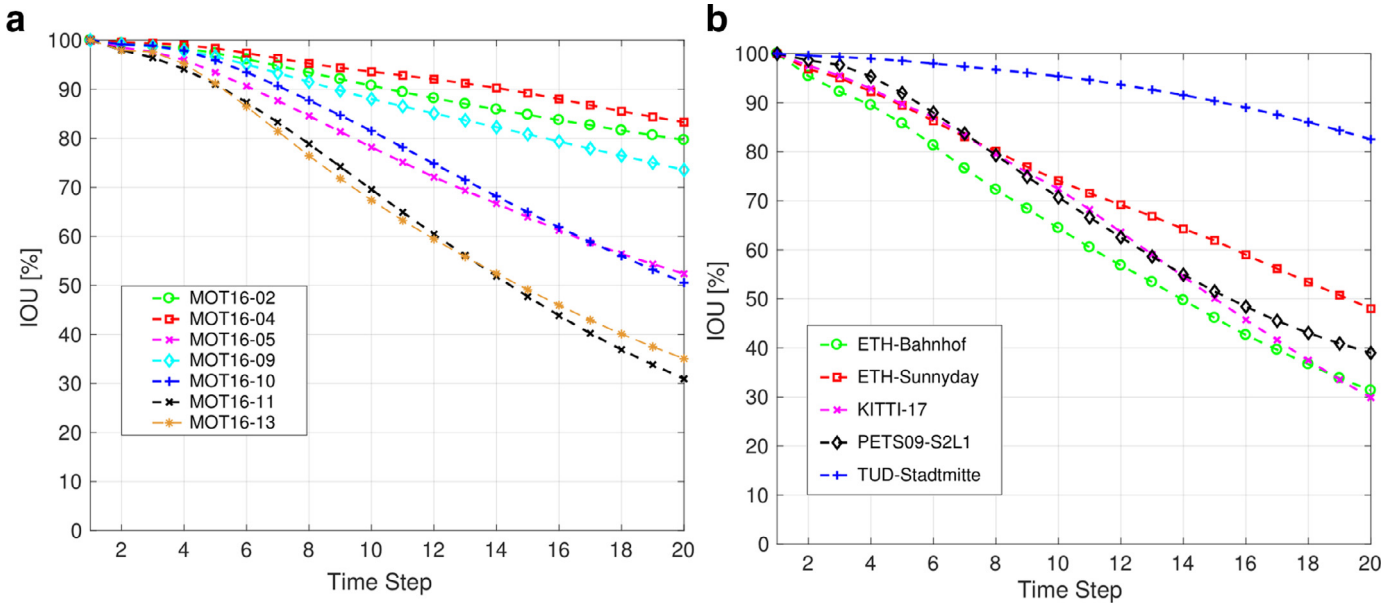

Fig. 14. The evaluation of the motion prediction by the RNN on (a) MOT16 and (b) MOT15 datasets.

Table 7
Comparison of the person re-identification models.

Model	ID-Net	SiameseNet	StackNet	StackNetPose	Ours
Acc. (%)	80.4	84.7	86.9	90.0	96.1

already used the MOT16 train data for training the RNN, the prediction performance is reported on unseen training data of the MOT15 benchmark as well. These two benchmarks have some videos in common, so we excluded them from the MOT15 data. Note that the set of generated tracklets from the MOT16 in this experiment are different from those have been used for training the RNN. Fig. 14 shows the intersection over union (IOU) average of the predicted detection boxes with the ground truth one at each time step. Evidently, as the time step increases, the predictions are less accurate due to error accumulation. It can also been seen that the predictions are more accurate for static cameras. For instance, the IOU for TUD-Stadtmitte sequence remains above 80% in all the next time steps, while the curve decreases remarkably for moving cameras like ETH-Bahnhof.

In Table 8, the results of 7 tracking algorithms with (highlighted rows) and without our occlusion handling is presented. Without any prior knowledge about these methodologies, the final tracklets of the methods released by the MOT16-Challenge were used as the input to our occlusion handling algorithm. As it can been seen, our approach reduces fragments and ID-switches, while keeping the MOTA metric relatively stable. By considering the ID-switch metric, it is clear that the proposed algorithm can decrease the ID-switch while other evaluation metrics are roughly the same. The amount of ID-switch reduction varies from one algorithm to another. The largest reduction is 988(40.86%) in the TBD algorithm [20], while the smallest reduction is 15(2.35%) in the EDMT algorithm [26]. Similarly, the number of false negatives mostly decreases by recovering the missed detection boxes in occlusion. Having the pretrained RNN, the motion prediction and tracklet stitching run at approximately 100 Hz on a CPU. This proposed post processing step is computationally efficient in reducing ID-switches of baseline tracking algorithm.

4.6. Discussion

Following the tracking-by-detection paradigm, we aim to obtain short and high confidence tracklets through a clustering graph optimization. The experiment shows that our CNN relatively performs well for assigning pair-wise detection affinity to the edges of this graph. We also verified the design structure for the CNN by testing on the MOT dataset and compared to other possible structures. However, the performance deceases gradually, as the frame gap of the two detection images increases due to illumination or pose body changes over a long occlusion. Additionally, in order to avoid having large graph for the optimization, we connect detections which are temporally close. Although, these are the reason to first create short tracklets, but still the FM and IDs values are high due to lengthy occlusion. Here, we propose to use the motion cues of tracklets before and after an occlusion in order to associate accurately those tracklets which very likely belong to the same target. However, conventional methods like Kalman-filter do not deliver acceptable results due to long occlusions, missing detections, inaccurate detection bounding boxes and the natural property of

Table 8
The effect of our occlusion handling (OH) method on the tracking results of state-of-the-art algorithms.

Tracking method	MOTA (%)↑	MOTP (%)↑	MT (%)↑	ML (%)↓	FP↓	FN↓	IDS↓	FM↓
TBD [20]	33.7	76.5	7.2	54.2	5804	112,587	2418	2252
TBD [20]+OH	34.1	76.9	7.0	56.1	4201	112,417	1430	1370
EDMT [26]	45.3	75.9	17.0	39.9	11,122	87,890	639	946
EDMT [26]+OH	45.7	75.9	17.0	39.9	11,128	87,865	624	942
CEM [5]	33.2	75.8	7.8	54.4	6837	114,322	642	731
CEM [5]+OH	33.6	75.8	7.6	54.5	6802	114,296	596	718
MHTDAM [11]	45.8	76.3	16.2	43.2	6412	91,758	590	781
MHTDAM [11]+OH	46.9	76.4	16.1	43.2	6257	91,669	549	757
QuadMOT16 [50]	44.1	76.4	14.6	44.9	6388	94,775	745	1096
QuadMOT16 [50]+OH	43.8	76.6	14.6	44.9	6994	94,758	669	1049
EAMTT_pub [25]	38.8	75.1	7.9	94.1	8114	102,452	965	1657
EAMTT_pub [25]+OH	39.5	75.2	7.9	94.1	8006	102,428	913	1606
IOU [24]	57.1	77.1	23.6	32.9	5702	70,278	2167	3028
IOU [24]+OH	58.1	77.2	23.1	33.3	4883	70,207	1624	2539

human walking which causes nonlinear changes especially for the variable 'width' of the detection box.

The experiment on the MOTchallenge dataset indicates that the proposed RNN is able to estimate detection box parameters with IOU larger than 50% till 20 frames in static camera scenarios, which is effective in handling long occlusions. Although the accuracy decreases for further predictions due to error accumulation, especially for the moving camera videos, the results show that the RNN can still reduce the number of ID-switches and hence improve the overall tracking performance. Meanwhile, the missing detections during occlusion are reconstructed using the RNN predictions. This leads to reduction in the number of false negatives, as it is evident from Table 3.

Last but not least, the proposed tracklet stitching process can be applied as a stand alone algorithm on results (as initial tracklets) of any baseline tracking method, not relying on prior knowledge of the underlying baseline method.

5. Conclusion

We proposed a method for the multiple people tracking problem. A CNN was trained to predict the likelihood of two detections belonging to the same target. Besides that, different criteria including height, location and confidence score of detections have been considered as cost edges in the graphical data association model. The primary tracklets are obtained by solving the minimum cost lifted multicut problem for the graphical model. To reduce the number of ID-switches in long occlusion, the tracklets are stitched together using motion information predicted by an RNN. As future work, one could extend this work by adding other constraints such as body shape or head pose to the graphical model.

Declaration of Competing Interest

All authors have participated in (a) conception and design, or analysis and interpretation of the data; (b) drafting the article or revising it critically for important intellectual content; and (c) approval of the final version.

References

[1] S. Tang, B. Andres, M. Andriluka, B. Schiele, Multi-person tracking by multicut and deep matching, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 100–111.

[2] A. Milan, L. Leal-Taixé, K. Schindler, I. Reid, Joint tracking and segmentation of multiple targets, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5397–5406.

[3] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, J. Yan, POI: multiple object tracking with high performance detection and appearance feature, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 36–42.

[4] C. Park, T.J. Woehl, J.E. Evans, N.D. Browning, Minimum cost multi-way data association for optimizing multitarget tracking of interacting objects, IEEE Trans. Pattern Anal. Mach. Intell. 37 (3) (2015) 611–624.

[5] A. Milan, S. Roth, K. Schindler, Continuous energy minimization for multitarget tracking, IEEE Trans. Pattern Anal. Mach. Intell. 36 (1) (2014) 58–72.

[6] R. Henschel, L. Leal-Taixé, D. Cremers, 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, 2018, pp. 1428–1437.

[7] M. Betke, Z. Wu, Data association for multi-object visual tracking, Synth. Lect. Comput. Vis. 6 (2) (2016) 1–120.

[8] Z. Wu, M. Betke, Global optimization for coupled detection and data association in multiple object tracking, Comput. Vis. Image Underst. 143 (2016) 25–37.

[9] T.E. Fortmann, Y. Bar-Shalom, M. Scheffe, Multi-target tracking using joint probabilistic data association, in: Proceedings of the 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes, IEEE, 1980, pp. 807–812.

[10] M. Hofmann, D. Wolf, G. Rigoll, Hypergraphs for joint multi-view reconstruction and multi-object tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 3650–3657.

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

[12] B. Leibe, K. Schindler, L. Van Gool, Coupled detection and trajectory estimation for multi-object tracking, in: Proceedings of the IEEE 11th International Conference on Computer Vision, IEEE, 2007, pp. 1–8.

[13] A. Andriyenko, K. Schindler, S. Roth, Discrete-continuous optimization for multi-target tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 1926–1933.

[14] A. Dehghan, S. Modiri Assari, M. Shah, GMMCP tracker: globally optimal generalized maximum multi clique problem for multiple object tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4091–4099.

[15] S. Tang, M. Andriluka, B. Schiele, Detection and tracking of occluded people, Int. J. Comput. Vis. 110 (1) (2014) 58–69.

[16] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: Proceedings of the 2012 Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.

[17] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.

[18] M. Keuper, E. Levinkov, N. Bonneel, G. Lavoué, T. Brox, B. Andres, Efficient decomposition of image and mesh graphs by lifted multicuts, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1751–1759.

[19] E. Levinkov, J. Uhrig, S. Tang, M. Omran, E. Insafutdinov, A. Kirillov, C. Rother, T. Brox, B. Schiele, B. Andres, Joint graph decomposition & node labeling: Problem, algorithms, applications, in: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[20] A. Geiger, M. Lauer, C. Wojek, C. Stiller, R. Urtasun, 3D traffic scene understanding from movable platforms, IEEE Trans. Pattern Anal. Mach. Intell. 36 (5) (2014) 1012–1025.

[21] M. Babaee, Y. You, G. Rigoll, Combined segmentation, reconstruction, and tracking of multiple targets in multi-view video sequences, Comput. Vis. Image Underst. 154 (2017) 166–181.

[22] P. Dollár, R. Appel, S. Belongie, P. Perona, Fast feature pyramids for object detection, IEEE Trans. Pattern Anal. Mach. Intell. 36 (8) (2014) 1532–1545.

[23] J. Gall, A. Yao, N. Razavi, L. Van Gool, V. Lempitsky, Hough forests for object detection, tracking, and action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2188–2202.

[24] E. Bochinski, V. Eiselein, T. Sikora, High-speed tracking-by-detection without using image information, in: Proceedings of the 2017 Advanced Video and Signal Based Surveillance (AVSS), IEEE, 2017, pp. 1–6.

[25] R. Sanchez-Matilla, F. Poiesi, A. Cavallaro, Online multi-target tracking with strong and weak detections, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 84–99.

[26] J. Chen, H. Sheng, Y. Zhang, Z. Xiong, Enhancing detection model for multiple hypothesis tracking, in: Proceedings of the 2017 Computer Vision and Pattern Recognition Workshops, 2017, pp. 18–27.
[27] A. Sadeghian, A. Alahi, S. Savarese, Tracking the untrackable: learning to track multiple cues with long-term dependencies, in: Proceedings of the IEEE International Conference on Computer Vision, 2017.
[28] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, S.Z. Li, Multiple target tracking based on undirected hierarchical relation hypergraph, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1282–1289.
[29] G. Wang, Y. Wang, H. Zhang, R. Gu, J.-N. Hwang, Exploit the Connectivity: Multi-Object Tracking with TrackletNet, arXiv:1811.07258 (2018).
[30] H. Shen, L. Huang, C. Huang, W. Xu, Tracklet association tracker: an end-to-end learning-based association approach for multi-object tracking, CoRR (2018). arXiv preprint arXiv: 1808.01562
[31] H. Sheng, J. Chen, Y. Zhang, W. Ke, Z. Xiong, J. Yu, Iterative multiple hypothesis tracking with tracklet-level association, IEEE Trans. Circuits Syst. Video Technol. PP (2018) 1-1.
[32] B. Yang, 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.
[33] L. Zhang, Y. Li, R. Nevatia, Global data association for multi-object tracking using network flows, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.
[34] H. Pirsiavash, D. Ramanan, C.C. Fowlkes, Globally-optimal greedy algorithms for tracking a variable number of objects, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2011, pp. 1201–1208.
[35] S. Tang, B. Andres, M. Andriluka, B. Schiele, Subgraph decomposition for multi-target tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5033–5041.
[36] S. Khan, M. Shah, A multiview approach to tracking people in crowded scenes using a planar homography constraint, in: Proceedings of the European Conference on Computer Vision, Springer, 2006, pp. 133–146.
[37] T. Kroeger, R. Dragon, L. Van Gool, Multi-view tracking of multiple targets with dynamic cameras, in: Proceedings of the German Conference on Pattern Recognition, Springer, 2014, pp. 653–665.
[38] E. Horbert, K. Rematas, B. Leibe, Level-set person segmentation and tracking with multi-region appearance models and top-down shape information, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2011, pp. 1871–1878.
[39] R. Henschel, Y. Zouand, B. Rosenhahn, Multiple people tracking using body and joint detections, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2019.
[40] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, M.S. Lew, Deep learning for visual understanding: a review, Neurocomputing 187 (2016) 27–48.
[41] M. Ehsani, M. Babaee, Recognition of farsi handwritten cheque values using neural networks, in: Proceedings of the 3rd International IEEE Conference Intelligent Systems, IEEE, 2006, pp. 656–660.
[42] H. Noh, S. Hong, B. Han, Learning deconvolution network for semantic segmentation, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1520–1528.
[43] W. Ouyang, X. Wang, Joint deep learning for pedestrian detection, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 2056–2063.
[44] N. Wang, S. Li, A. Gupta, D.-Y. Yeung, Transferring Rich Feature Hierarchies for Robust Visual Tracking, arXiv:1501.04587 (2015).
[45] H. Li, Y. Li, F. Porikli, DeepTrack: learning discriminative feature representations online for robust visual tracking, IEEE Trans. Image Process. 25 (4) (2016) 1834–1848.
[46] N. Wojke, A. Bewley, D. Paulus, Simple Online and Realtime Tracking With a Deep Association Metric, arXiv:1703.07402 (2017).
[47] J. Fan, W. Xu, Y. Wu, Y. Gong, Human tracking using convolutional neural networks, IEEE Trans. Neural Netw. 21 (10) (2010) 1610–1623.
[48] L. Leal-Taixé, C. Canton-Ferrer, K. Schindler, Learning by tracking: Siamese CNN for robust target association, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2016, pp. 33–40.
[49] S. Tang, M. Andriluka, B. Andres, B. Schiele, Multiple people tracking with lifted multicut and person re-identification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
[50] J. Son, M. Baek, M. Cho, B. Han, Multi-object tracking with quadruplet convolutional neural networks, in: Proceedings of the Computer Vision and Pattern Recognition, 2017, pp. 5620–5629.
[51] A. Milan, S.H. Rezatofighi, A. Dick, I. Reid, K. Schindler, Online multi-target tracking using recurrent neural networks, in: Proceedings of the 2016 AAAI, 2016.
[52] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: Proceedings of the IEEE International Conference on Image Processing (ICIP), IEEE, 2016, pp. 3464–3468.
[53] L. Pishchulin, E. Insafutdinov, S. Tang, B. Andres, M. Andriluka, P.V. Gehler, B. Schiele, DeepCut: joint subset partition and labeling for multi person pose estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4929–4937.
[54] P. Weinzaepfel, J. Revaud, Z. Harchaoui, C. Schmid, DeepFlow: large displacement optical flow with deep matching, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 1385–1392.
[55] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291–307.
[56] W. Lin, J. Peng, S. Deng, M. Liu, X.H. Jia, Real-Time Multi-Object Tracking With Hyper-Plane Matching (V2), Technical Report, Shanghai Jiao Tong University, 2017.
[57] B. Lee, E. Erdenee, S. Jin, M.Y. Nam, Y.G. Jung, P.K. Rhee, Multi-class multi-object tracking using changing point detection, in: Proceedings of the European Conference on Computer Vision, Springer, 2016, pp. 68–83.
[58] W. Choi, Near-online multi-target tracking with aggregated local flow descriptor, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3029–3037.
[59] A. Bewley, Z. Ge, L. Ott, F. Ramos, B. Upcroft, Simple online and realtime tracking, in: Proceedings of the IEEE International Conference on Image Processing (ICIP), IEEE, 2016, pp. 3464–3468.
[60] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, K. Schindler, MOT16: A Benchmark for Multi-Object Tracking, arXiv:1603.00831 (2016).
[61] R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 1440–1448.
[62] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, K. Schindler, MotChallenge 2015: Towards a Benchmark for Multi-Target Tracking, arXiv:1504.01942 (2015).
[63] K. Bernardin, R. Stiefelhagen, Evaluating multiple object tracking performance: the clear MOT metrics, EURASIP J. Image Video Process. 2008 (1) (2008) 1–10.
[64] R. Henschel, L. Leal-Taixé, D. Cremers, B. Rosenhahn, A novel multi-detector fusion framework for multi-object tracking, in: Proceedings of the Computer Vision and Pattern Recognition Workshops, 2018.
[65] M. Keuper, S. Tang, Y. Zhongjie, B. Andres, T. Brox, B. Schiele, A Multi-Cut Formulation for Joint Segmentation and Tracking of Multiple Objects, arXiv:1607.06317 (2016).
[66] Z. Fu, P. Feng, F. Angelini, J. Chambers, S. Naqvi, Particle PHD filter based multiple human tracking using online group-structured dictionary learning, IEEE Access 6 (2018) 14764–14778.
[67] A. Maksai, P. Fua, Eliminating exposure bias and metric mismatch in multiple object tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 4639–4648.
[68] T. Kutschbach, E. Bochinski, V. Eiselein, T. Sikora, Sequential sensor fusion combining probability hypothesis density and kernelized correlation filters for multi-object tracking in video data, in: Proceedings of the 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), 2017, pp. 1–5.
[69] P.F. Felzenszwalb, R.B. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, IEEE Trans. Pattern Anal. Mach. Intell. 32 (9) (2010) 1627–1645.
[70] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2015, pp. 91–99.
[71] F. Yang, W. Choi, 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, 2016, pp. 2129–2137.
[72] K. Simonyan, A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, arXiv:1409.1556 (2014).
[73] S. Zagoruyko, N. Komodakis, Learning to compare image patches via convolutional neural networks, in: Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2015, pp. 4353–4361.

Maryam Babaee received her B.S. degree in computer engineering from the University of Tehran, Tehran, Iran, in 2011, and the M.S. degree in Artificial Intelligence from the University of Isfahan, Isfahan, Iran, in 2014. She is currently pursuing Ph.D. degree in electrical and computer engineering at the Technical University of Munich (TUM), Munich, Germany. Her research interests include computer vision, image processing, and machine learning.

Zimu Li received his master degree in electrical and computer engineering at Technical University of Munich (TUM), Munich, Germany. He received the bachelor degree in mechatronics with honor from Tongji University, Shanghai, China, in 2015. His current research is centered on machine learning, computer vision, and artificial neural networks.

Gerhard Rigoll The research of Prof. Rigoll (b. 1958) deals with all aspects of pattern recognition for multimodal human-machine interaction. Subfields include speech processing, audiovisual information processing, handwriting recognition, gesture and emotion recognition, face detection and recognition, object tracking and interactive graphical systems. He is the author or co-author of over 500 publications and has been member of many different program committees. He has been involved in numerous expert panels in Germany and internationally. After studying technical cybernetics in Stuttgart, he became research assistant at Fraunhofer Institute (IAO) in Stuttgart. He obtained his doctoral degree in 1986 with a thesis on automatic speech recognition. After that, he was a postdoctoral fellow at IBM Thomas Watson Research Center in Yorktown Heights/USA until 1988. After qualifying as a lecturer in Stuttgart from 1991 to 1993, he was a visiting scientist at the NTT Human Interface Laboratory in Tokyo. From 1993 to 2001, he was professor of computer engineering at Gerhard Mercator University in Duisburg prior to accepting his current position at TUM in 2002.