Applied Intelligence (2022) 52:1268–1288
https://doi.org/10.1007/s10489-021-02457-5

Online multi-object tracking using multi-function integration and tracking simulation training

Jieming Yang[1,2] · Hongwei Ge[1,2] · Jinlong Yang[1,2] · Yubing Tong[3] · Shuzhi Su[4]

Accepted: 20 April 2021 / Published online: 19 May 2021

Abstract
Recently, with the development of deep-learning, the performance of multi-object tracking algorithms based on deep neural networks has been greatly improved. However, most methods separate different functional modules into multiple networks and train them independently on specific tasks. When these network modules are used directly, they are not compatible with each other effectively, nor can they be better adapted to the multi-object tracking task, which leads to a poor tracking effect. Therefore, a network structure is designed to aggregate the regression of objects between frames and the extraction of appearance features into one model to improve the harmony between various functional modules of multi-object tracking. To improve the support for the multi-object tracking task, an end-to-end training method is also proposed to simulate the multi-object tracking process during the training and expand the training data by using the historical position of the target combined with the prediction of the motion model. A metric loss that can take advantage of the historical appearance features of the target is also used to train the extraction module of appearance features to improve the temporal correlation of extracted appearance features. Evaluation results on the MOTChallenge benchmark datasets show that the proposed approach achieves state-of-the-art performance.

Keywords Deep learning · Neural network · Computer vision · Multi-object tracking · Object detection

1 Introduction

Multiple Object Tracking (MOT) is a computer vision task that aims to analyze video to identify and track targets belonging to one or more categories, such as pedestrians, cars, animals, and inanimate objects, without any prior knowledge about the appearance and the number of targets. Unlike the detection task of outputting the bounding box set determined by the coordinates, height, and width, the multi-object tracking algorithm will assign an identity to each bounding box to distinguish these inter-class targets. MOT tasks play an important role in computer vision: from video surveillance to autonomous vehicles, from motion recognition to crowd behavior analysis, many of these problems will benefit from high-quality tracking algorithms.

According to the way of processing data, multi-object tracking algorithms can be divided into two categories. One type is online tracking, which only processes the current data frame. It is suitable for real-time tasks such as autonomous driving. The other is offline tracking, which uses the data of the entire video frame. Although offline methods show better performance than online tracking algorithms, they are generally not suitable for time-critical applications. Due to the adoption of the global optimization process, the calculation overhead of the offline algorithm is relatively large. Besides, multi-hypothesis tracking [1] has also been widely adopted as a semi-online tracking framework. The main interest of this paper is online multi-object tracking, so the following part only focuses on the online tracking framework.

With the development of target detectors, tracking-by-detection [2–5] is regarded as the most popular tracking paradigm. E.g., the literature [2] utilizes the online-provided detection to generate tracklets, then obtain the tracklet confidence to solve multiple object tracking problems. The literature [3] introduces a dynamic version of the successive

✉ Hongwei Ge
ghw8601@163.com

[1] School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, Jiangsu, China

[2] Key Laboratory of Advanced Process Control for Light Industry (Jiangnan University), Ministry of Education, Wuxi 214122, China

[3] Medical Image Processing Group, Department of Radiology, University of Pennsylvania, Philadelphia, PA 19104, USA

[4] School of Computer Science and Engineering, Anhui University of Science & Technology, Huainan 232001, Anhui, China

shortest-path algorithm, which solves the data association problem between detection and candidates optimally. In the literature [4], object detection and data association are expressed by a single objective function. A coupling formulation is used to avoid the problem of error propagation. To calculate the affinity between detections and tracks, STRN [5] present a unified framework for similarity measurement by integrating multiple cues in an end-to-end manner through extending the object-object relation network from the spatial domain to the spatial-temporal domain. However, these methods rely heavily on detection results. False detection and missed detection will cause false positives, false negatives and identity switches. Some methods introduce a single object tracker to track a target frame by frame as a supplement to solve missed detection. For example, the literature [6] adopts a CNN-based single object tracker to regress the target of the previous frame to the current frame and utilizes a spatio-temporal attention mechanism to associate targets in different frames. The literature [7] employs ECO [8] as the baseline tracker combined with a dual matching attention mechanism to perform multi-object tracking. The literature [9] utilizes SiamRPN [10] to provide short-term clues and a ReID sub-network to provide long-term clues. Based on these two clues, a switcher-aware classifier is utilized to calculate the affinity between targets across frames. IAT [11] propose an instance-aware tracker to integrate SOT techniques for MOT by encoding awareness both within and between target models. However, the algorithm combined with single object tracking is prone to tracking drift when processing occluded targets, which leads to an increase in the number of false positives. MOTDT [12] handle unreliable detection by collecting candidates from outputs of both detection and tracking. In response to false detection, the literature [13] propose the historical appearance matching method and joint-input siamese network which was trained by a 2-step process to improve the uncertain target state caused by it. Based on the literature [13], the literature [14] introduces a data-driven association method, which successfully associates concatenated templates of target and observation and directly derives the possibility of the target-detection pair, further alleviating the impact of false detection.

Different from the literature [13, 14], Tracktor [15] utilizes the regression network of the faster-RCNN to refine the predicted position of the motion model and the public detection to provide more accurate target positions, strengthen the robustness of the algorithm to false detection. In response to missed detection, Tracktor utilizes a ReID network to extract the appearance features of the target and save it as a historical appearance feature, and then restore its identity by matching the appearance feature when the target reappears. By solving false detection and missed detection, Tracktor obtained state-of-the-art tracking results on the MOTChallenge benchmark website.

However, whether Tracktor or the other methods above, its network model consists of several independent modules with different functions, such as the single-object tracking module, regression module, and appearance feature extraction module. Each module carries out targeted training on different tasks. When these separately trained modules are directly applied to a multi-object tracking task, they may not be effectively compatible and adapted to this task and ultimately lead to a decline in the tracking ability. First, the bounding box samples during the training of the ReID network is the ground truth position of the target provided by the label. Still, the appearance feature during tracking comes from the bounding boxes refined by the faster-RCNN, which leads to the conflict of them and causes incompatibility between the two modules. Second, Tracktor directly utilizes RPN to provide proposal regions to train the refinement ability of the regression network. In inference, when this regression network receives a bounding box from motion prediction that is not in the distribution of the proposal regions provided by the RPN, it cannot be refined to fit the target location accurately more. Therefore, based on Tracktor, we designed a new network structure that aggregates multiple functional modules and proposed an end-to-end training method for multi-object tracking tasks to train each module in the network simultaneously to make them compatible with each other and more suitable for the tracking task, thereby improving the performance of multi-object tracking of the whole model.

Our contributions in this paper are as follows:

- We designed a new network structure that includes a classification module, regression module, and ReID module that share a backbone, which can refine the location of the target, realize the regression of tracks across frames, filter out non-pedestrian objects, and provide the appearance features of the target. For training these three modules simultaneously, we also introduce a metric loss to train the ReID head. First, compared to Tracktor [15], this network structure has fewer parameters and takes up less storage space. Second, it is also possible to directly extract the appearance features of the target during inference, without inputting the refined bounding box to the additional ReID network, thereby improving the computational efficiency. Finally, because the three modules share a backbone, they have a consistent middle-level feature representation. After a synchronous training process, the regression of the bounding box and the extraction of appearance features can promote each other to obtain the bounding box that can more accurately fit the target and the appearance features that can better reflect the appearance information of the target, thereby improving the tracking performance of the model.
- We proposed an end-to-end training method, which simulates the multi-object tracking process during training and utilizes the historical position of the target combined with the motion model to predict the position of the target

in the current frame. The predicted position and ground-truth position are used to generate extended bounding boxes that approximately meet the distribution of motion prediction as a supplementary sample to participate in the training of the regression network, which improves the adaptability of the regression network to the predicted position.

- To predict the location of the target more accurately, we adopted a motion model that combines the Kalman filter [16] and Enhanced Correlation Coefficient (ECC) [17]. Through ECC, the position change of the target caused by rigid motion such as camera lens movement is processed first, and then the Kalman filter combined with the motion intensity given by the ECC is used to predict the position of the target after its own motion, which improves the accuracy of motion prediction.
- We evaluated the proposed method on four benchmark datasets provided by MOTChallenge and compared it with the published state-of-the-art works, which confirmed that the proposed method has superior multi-object tracking performance. We also performed an ablation experiment to illustrate the contribution of each part of the proposed method to the tracking effect.

2 Related work

2.1 Object detection

Multi-object tracking without frame-to-frame identity prediction is a sub-problem usually refers to as video object detection. In recent years, the target detection method based on deep learning has achieved excellent detection performance. The two-stage algorithm developed from work [18] to work [19] and then to work [20], achieving increasingly superior detection accuracy. However, the one-stage algorithm has a faster detection speed. One-stage algorithm [21] utilizes a single neural network to predict bounding boxes and class probabilities directly from full images in one evaluation, which achieves a faster speed. While ensuring high speed, algorithm [22] has expanded the detection category of A to more than 9000. Based on [22], algorithm [23] introduces multi-scale feature mapping and obtains higher detection accuracy. There are also methods to consider both object detection and tracking jointly. The literature [24] proposes a network that processes two consecutive frames and uses ground-truth data to improve the detection regression, thereby generating a two-frame trajectory. Through subsequent offline methods, these trajectories are merged into multi-frame trajectories. The literature [25] provides an RNN with detection services through an SSD [26], thus realizing the combination of tracking and detection. Like Tracktor [15], to make a fair comparison with other algorithms, we did not use a private detector to provide detection results but only evaluate the proposed algorithm on the public detection set provided by MOTChallenge.

2.2 Motion prediction

Multi-object tracking algorithms usually adopt a motion model to predict the position of the target based on the tracking historical trajectory and then perform data association with the target detection in the current frame. For example, the literature [7] applies a constant velocity model to predict the position of the target in the current frame. The literature [27, 28] both exploit the Kalman filter as a motion model to predict the position of the target and introduce the Hungarian algorithm for data association, which has achieved a breakneck processing speed. Based on the literature [27], the literature [28] further introduces appearance features. Tracktor [15] adds an enhanced correlation coefficient (ECC) based on the constant motion model as the compensation of camera motion. The literature [29] utilizes the proposed motion intensity measure to integrate the motion of pedestrians and cameras to overcome their mutual influence. In the literature [30], the recurrent neural network, RNNs, which can process timing information, is used to predict and update the state of the target. However, without other optimizations, the algorithm does not achieve outstanding tracking results and needs further improvement. Built upon standard RNN components, the RAN [31] learns to extract long-term sequence history in an internal memory represented by the recurrent hidden layer to estimates the probability distribution of new detections in an autoregressive manner. The proposed algorithm also combines the Kalman filter with ECC as the motion model, and the excellent performance of this method is proved in ablation experiments.

2.3 Appearance model

In complex scenes with frequent occlusion, it is often not sufficient to use only position information to determine the association of targets in two consecutive frames, which can easily cause the identity switch of the target, leading to an increase in the number of IDsw. One solution is to add the appearance information of the target in the affinity calculation and utilize the two information of position and appearance to determine the similarity between targets in different frames. The simplest appearance model is the color histogram. E.g., in the literature [32], RGB color histogram and correlogram are exploited as the color cues and texture properties are represented by local binary patterns. Still, the performance is not improved much. Therefore, some methods using manual features, such as histogram of gradient (HOG) [33] and optical flow [34], have been adopted. Recently, feature extraction methods based on deep learning have been proposed to obtain stronger discrimination capabilities. The literature [1] first

utilizes a pre-trained CNN to extract the appearance features of the target from the detection and then applies PCA for dimensionality reduction. The literature [35] exploits a modified version of GoogLeNet [36] trained on the ReID dataset to extract the appearance features of the target. The literature [37] combines the visual features extracted by CNN as well as dynamic and location features and utilizes a Hungarian algorithm to solve the association problem. Tracktor [15] utilizes the pre-trained network in [38] to extract the appearance information of the target and re-identity the track that reappears after being lost. However, the appearance feature extraction module in the above method is independent of other functional modules, and the training process does not interfere with each other. When using together, it will cause poor compatibility, ambiguity and affect the tracking effect. And we add a ReID branch to the network structure and introduce a metric loss to train it, which improves the compatibility of each functional module and obtain more effective appearance features.

2.4 End-to-end training

Most of the multi-object tracking methods based on deep learning train different modules in the model on corresponding applications. For example, the methods combined with single-object tracking [6, 7] will train a single-target tracker based on the SOT task on the corresponding datasets. The algorithm [35] that adopts appearance features also trains the feature extraction network on the ReID dataset. Although Tracktor [15] utilizes the public detection set for tracking, it trains the network model on the tracking dataset based on the detection task. These methods do not train the network end-to-end for multi-object tracking tasks. When these network models are directly applied to the tracking task, each tracking module cannot adapt to the task better, resulting in poor tracking performance. To solve this problem, in recent years, some multi-object tracking works have adopted end-to-end training methods. For example, in the literature [39], the author utilizes multi-scale appearance features to construct an affinity matrix, and then during training, by padding rows or columns on the calculated soft allocation matrix, and then performing column-wise or row-wise softmax to simulate data associate process to achieve end-to-end training. The literature [40] proposes a deep Hungarian network to simulate data association based on the Hungarian algorithm and then calculate the differentiable *dMOTA* and *dMOTP* as the metric loss to train the network. FAMnet [41] integrates feature extraction, affinity calculation, and differentiable data association [42] into a network to achieve end-to-end training. The literature [43] exploits the graph network and the frames before and after the current frame to train the affinity scores between targets and then performs data association based on the trained affinity scores during inference. The tracking results are all at the forefront of multiple tracking channels of the MOTchallenge benchmark. The proposed algorithm simulates the tracking process during training. It used the tracking trajectory and the motion model to predict the position to expand the training data, thereby achieving end-to-end training.

2.5 Car tracking

Although the proposed method is mainly aimed at pedestrian targets in the scene, in order to prove that this method is compatible with different types of targets, the proposed method is tested on the car category of the Kitti dataset. There are also many methods to tracking the car in the scene. In leterature [44], A temporal window is constituted by three consecutive frames. Tracklets of the detected car objects in the temporal window are generated by solving min-cost max flow problem of a sparse affinity network with a limited number of edges, which are created if a strong detection affinity exists. CIWT [45] propose to use image- and world-space information jointly to track car in urban street scenes. SCEA [46] propose a new data association method that effectively exploits structural motion constraints in the presence of large camera motion and a novel event aggregation approach is developed to integrate structural constraints in assignment costs for online MOT. LP-SSVM [47] describe an end-to-end framework for learning parameters of min-cost flow multi-target tracking problem with quadratic trajectory interactions including suppression of overlapping tracks and contextual cues about co-occurrence of different objects. In extraCK [48], multiple-object vehicle tracking system by affinity matching using min-cost linear cost assignment is proposed.

3 The proposed method

3.1 Motion prediction

In some cases, the IOU (intersection-over-union) based data association can outperform many general methods. This becomes possible due to the high-quality detection and high frame rates. However, if there is a large pedestrian motion, camera motion, or low frame rates, we must consider it. Generally, the position change of the target between video frames caused by pedestrian movement can be regarded as non-rigid motion, and the Kalman Filter is used to predict the position of the target after movement. The deviation of the target and the entire background in the video frame caused by camera motion can be regarded as rigid motion, and we exploit the Enhanced Correlation Coefficient (ECC) method to calculate the offset affine matrix to align the continuous frames. Considering the space consistency, the pedestrian motion model needs to be processed before the camera motion model. In detail, each position of the target needs to be predicted by the Kalman Filter firstly and aligned by the ECC

model, which is named as Kalman+ECC, and it can be established as below:

$$\begin{aligned} s_{t+1} &= warp(Fs_t) \\ P_{t+1} &= FP_tF^T + Q \end{aligned} \tag{1}$$

Where, s represents the 8-dimensional motion state $[c_x, c_y, a, h, v_{c_x}, v_{c_y}, v_a, v_h]$, including the target centre coordinate $[c_x, c_y]$, the target length-width ratio a, the target height h, and the change rate $[v_{c_x}, v_{c_y}, v_a, v_h]$ of the first four variables. Q stands for motion covariance, P for prior covariance, *warp* for ECC model.

However, the independent motion processing of the Kalman+ECC solution will raise a compatible problem. Therefore, referring to the method [29], we mix the camera motion and pedestrian motion using the affine matrix to adjust the integrated motion model, which is named as Kalman&ECC. In this way, the integrated motion model can adapt to various motion scenes without pre-defined parameters. First, we define the intension of camera motion as Eq.2:

$$I_c = 1 - \frac{W \times R}{\|W\|_2 \times \|R\|_2}, R = [I; 0] \tag{2}$$

Where the I_c denotes the intension of camera motion, W denotes the vectorization of the affine matrix. The R means the affine matrix of static frames. I is the identity matrix, and O is the all-zero matrix. With the intension defined above, we can adjust the Kalman Filter by changing the state transition matrix:

$$\begin{aligned} S_{t+1} &= warp(F_c s_t) \\ P_{t+1} &= aF_cP_tF_c^T + Q \end{aligned}, F_c = \begin{bmatrix} I(d_t + I_c) & I \\ 0 & I \end{bmatrix} \tag{3}$$

Where the F_c denotes the adjusted state-transition matrix and the d_t means the original time step of the Kalman Filter. After the state s_{t+1} of the target in the t + 1 frame is obtained by the above formula, we can extract the position part vector $[c_x, c_y, a, h]$ and calculate the prediction bounding box $[x_1, y_1, x_2, y_2]$ of the target, where, x_1, y_1, x_2, and y_2 are the coordinates of the upper left corner and the lower right corner of the bounding box, respectively.

3.2 Network structure

Based on the Faster-RCNN [20], we design a network structure with three functions. As shown in Fig. 1, we adopt Resnet50 [49] +FPN [50] as the backbone. Three branch networks composed of the Fully Connected Layer (FC Layer) are designed, which are Classification head that output the probability of the object being target or background, Regression head that output the regression coefficient of the refined bounding box, and ReID head that output the appearance features of the target. It is equivalent to adding a ReID head to the network structure of faster-RCNN. Where Resnet50 consists of five stages, the first stage is composed of a single convolutional layer, BatchNorm [51] layer, Relu activation function, and Max Pool, while the other stages are all composed of a Conv Block [49] and two ID blocks [49]. We will also use the Region Proposal Network (RPN) proposed in Faster-RCNN to provide the proposal regions when training networks.

Figure 1 also shows the network data processing flow after a given image frame. First, the image frame is inputted into Resnet50 to obtain feature maps of different scales, and they are used to build feature pyramids. Then, according to the size of the bounding box of the target, we select the feature mapping of the corresponding scale to extract the appropriate feature mapping of the target, ROI pooling is conducted to obtain the ROI features of the target, which is finally inputted into three branch network Regression head, Classification head, and ReID head to obtain the bounding box regression coefficient, classification probability and appearance features of the target.

3.3 Training strategy

In Tracktor [15], the author trains the network model on the tracking datasets based on detection tasks. The model obtained by such a training method is directly applied to the multi-object tracking task, which will cause incompatibility problems. The proposal regions during training are generated from the Region Proposal Network (RPN). With the training, the proposal regions that are more accurate and closer to the actual target position can be generated by RPN, that is, the dispersion degree of the distribution of the training samples for regression and classification branch is reduced. However, different from the training, the proposal regions during inference came from the prediction of the motion model. In some video sequences with complex scenes, frequent occlusion, and nonlinear motion trajectory of the target, these predicted positions may be far from the actual position of the target and are not in the training sample distribution. After these low-quality proposal regions are inputted into the regression network, they cannot be refined to the actual target location, which affects the follow-up tracking and leads to the degradation of tracking performance.

Therefore, we simulate the multi-object tracking process in training, record the actual position of the target as the tracking trajectory, estimated the possible position of each target in the current frame by using the motion model, and add it into the training of the regression network as a supplement to the proposal regions. The whole training process is shown in Fig. 2:

Consecutive N frames of images are randomly selected from the training dataset. The ground-truth label of the first frame is used to initialize the tracking state of the target, which includes the position state and historical ROI features. Instead of saving the appearance features of the target, we save the ROI features of the target since the weight of the ReID head needs to be updated during training. When calculating

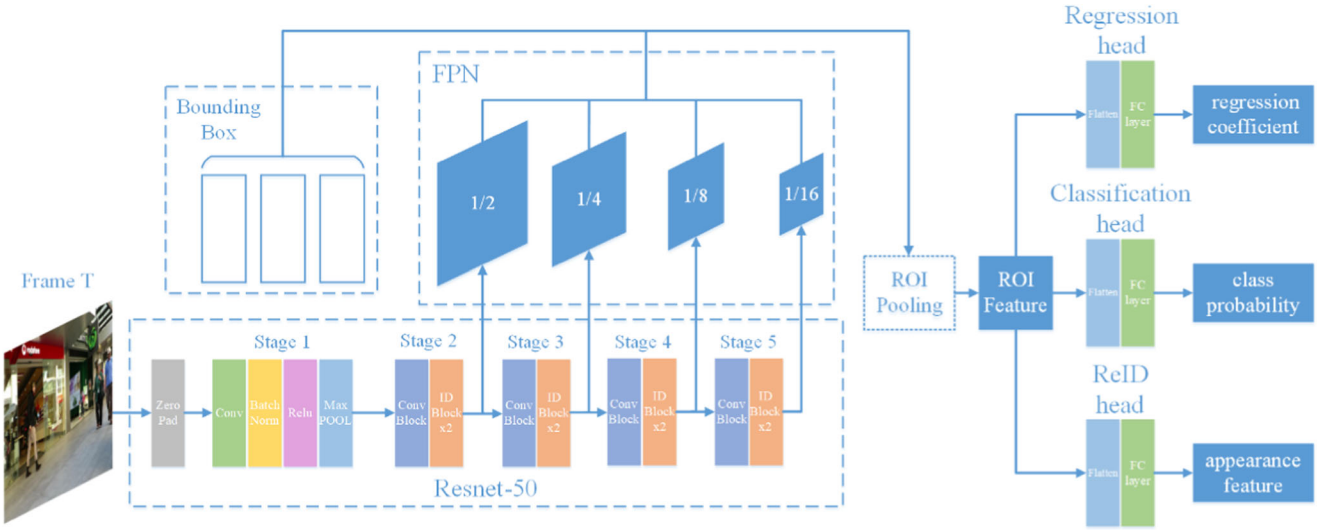


Fig. 1 The structure of the designed network model. The network consists of 5 modules, namely Resnet-50, FPN, Regression head, Classification head, and ReID head

appearance-based affinity, the historical ROI features can be input into the ReID head to obtain the historical appearance features of the targets. When the tracking state of the target is initialized, the bounding box provided in the ground-truth label is converted to the Kalman state, and the ROI feature of the target is added to the historical ROI features.

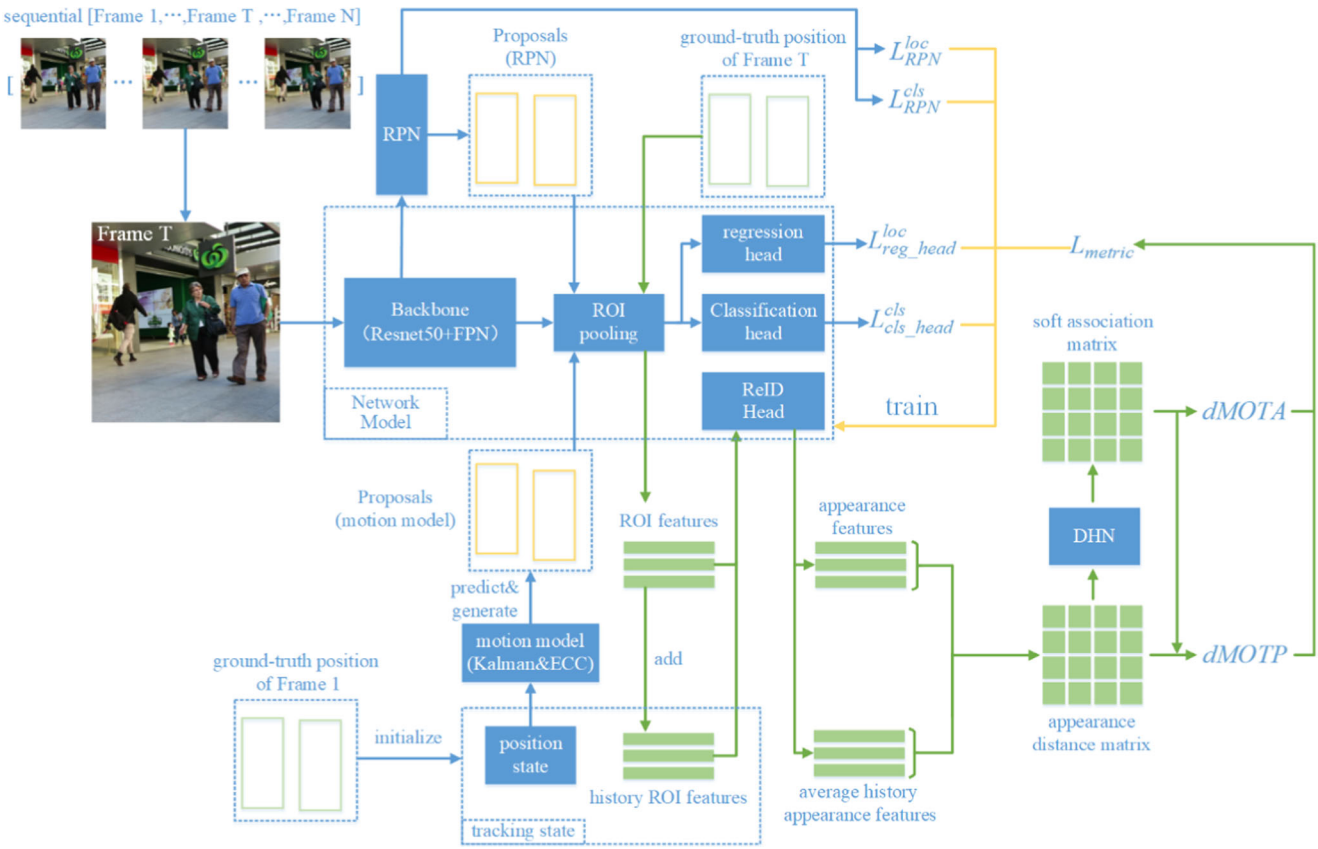


Fig. 2 The proposed training strategy for network model. The method provided by Faster-RCNN [20] is used to calculate the loss L^{loc}_{RPN} and L^{cls}_{RPN} of RPN. The proposal regions generated by combining the position information and the motion model and the proposal regions provided by the RPN are used as the training samples to calculate the loss $L^{loc}_{reg\_head}$ and $L^{cls}_{cls\_head}$. The saved history ROI features and the appearance features from ground-truth positions are exploited to calculate the metric loss L_{metric}

When it comes to the next frame, the ground-truth label is used to calculate the losses of three parts, namely, the losses L_{RPN}^{loc} and L_{RPN}^{cls} from regional proposal Network (RPN), the losses $L_{reg\_head}^{loc}$ and $L_{cls\_head}^{cls}$ from regression network, as well as the metric loss L_{metric} used to train the appearance extraction module. The objective function of the network model is as follows:

$$Loss = \lambda_1\left(L_{RPN}^{\text{loc}} + L_{RPN}^{\text{cls}}\right) + \lambda_2\left(L_{reg\_head}^{loc} + L_{cls\_head}^{cls}\right) + \lambda_3 L_{metric} \quad (4)$$

Where λ_1, λ_2, and λ_3 are the scale factor that controls the influence of different sub-losses on the objective function. The calculation process of each sub-loss will be introduced in the following sub-sections. After the loss calculated by the formula (4) is used to train the network model and update the weight, the ground-truth label is used to update the tracking state of the targets. The Kalman state of the target is updated with the ground-truth position in the current frame, and the ROI feature of the target is added to the historical ROI features. After the training of continuous N frames of images, the tracking state of all targets is cleared, and then the continuous N images are randomly selected for the above training.

3.3.1 The losses of RPN

The method provided by Faster-RCNN [20] is used to calculate the loss of RPN. We assign a positive label to two kinds of anchors: (i) the anchor/anchors with the highest Intersection-over-Union (IoU) overlap with a ground-truth box, or (ii) an anchor that has an IoU overlap higher than 0.7 with any ground-truth box. We assign a negative label to a non-positive anchor if its IoU ratio is lower than 0.3 for all ground-truth boxes. Anchors that are neither positive nor negative do not contribute to the training objective. The loss functions are calculated as follows:

$$L_{RPN}^{cls} = \frac{1}{N}\sum_i L_{cls}\left(P_i, P_i^*\right) \quad (5)$$

$$L_{RPN}^{loc} = \frac{1}{N_{reg}}\sum_i p_i^* L_{reg}\left(t_i, t_i^*\right) \quad (6)$$

Where i is the index of anchor, and P_i is the prediction probability that anchor points contain a target, which comes from the classification head in RPN. The corresponding label P_i^* of the positive anchor is 1, and 0 for negative. t_i is a vector representing the 4 parameterized coordinates of the predicted bounding box, and t_i^* is that of the ground-truth box associated with a positive anchor. The classification loss L_{cls} is log loss over two classes (object versus not object). For the regression loss, we use $L_{reg}\left(t_i, t_i^*\right) = R\left(t_i - t_i^*\right)$ where R is the robust loss function (smooth L1) defined in [19]. The term $P_i^* L_{reg}$ means the regression loss is activated only for positive anchors ($P_i^* = 1$) and is disabled otherwise ($P_i^* = 0$). These two items are normalized by batch-size N and the number of anchors N_{reg}.

3.3.2 The losses of regression head and classification head

In this section, we focus on the generation of the proposal regions. As mentioned before, the proposal regions for training are derived from two modules, namely RPN and motion prediction. As for the generation mode of proposal regions in RPN, we refer to paper [20] for more details. The number of proposal regions derived from motion prediction depends on the number of targets. In some datasets with fewer targets, the number of training data is also relatively small, and these proposal regions are usually regarded as positive samples. The lack of negative samples will lead to training imbalance and the lack of generalization ability of the model. Therefore, the method in Fig. 3 is used to generate more positive samples and negative samples as a supplement, and meanwhile, the generative positive samples can be close to the distribution of the motion prediction of the target.

As shown in Fig.3, supplementary samples are generated through four steps, namely boundary box interpolation, negative sample generation, random scaling and bias, and sample filtering. First, for a given ground-truth bounding box and the predicted bounding box of a target, N_{inter} equal parts interpolation is carried out between them to generate some interpolation bounding boxes. These interpolating bounding boxes, ground-truth bounding boxes, and predicted bounding boxes are considered as potential positive samples. Secondly, some negative sample bounding boxes are generated around each potential positive sample bounding box. Specifically, for a potentially positive sample bounding box, we make multiple copies of it. Each copy is randomly offset on the horizontal and vertical axes with a larger scale to make sure that the IoU between it and the potential positive sample bounding box is less than the threshold τ_{pos}, then perform a random length and width scaling. A smaller scale offset to obtain multiple potential negative sample bounding boxes. All the potential positive sample bounding boxes are similarly offset and scaled randomly with a small range, ensuring that the IoU between most potential positive samples and the ground-truth bounding boxes is bigger than the threshold τ_{pos}. Finally, The potential positive samples whose IoU between the ground-truth bounding box and it is larger than the threshold τ_{pos} are marked as the positive samples corresponding to the target, otherwise marked as negative samples. Same as faster-RCNN, after obtaining positive and negative samples of all targets, we randomly select a certain number of positive and negative samples meeting a certain proportion as training samples.

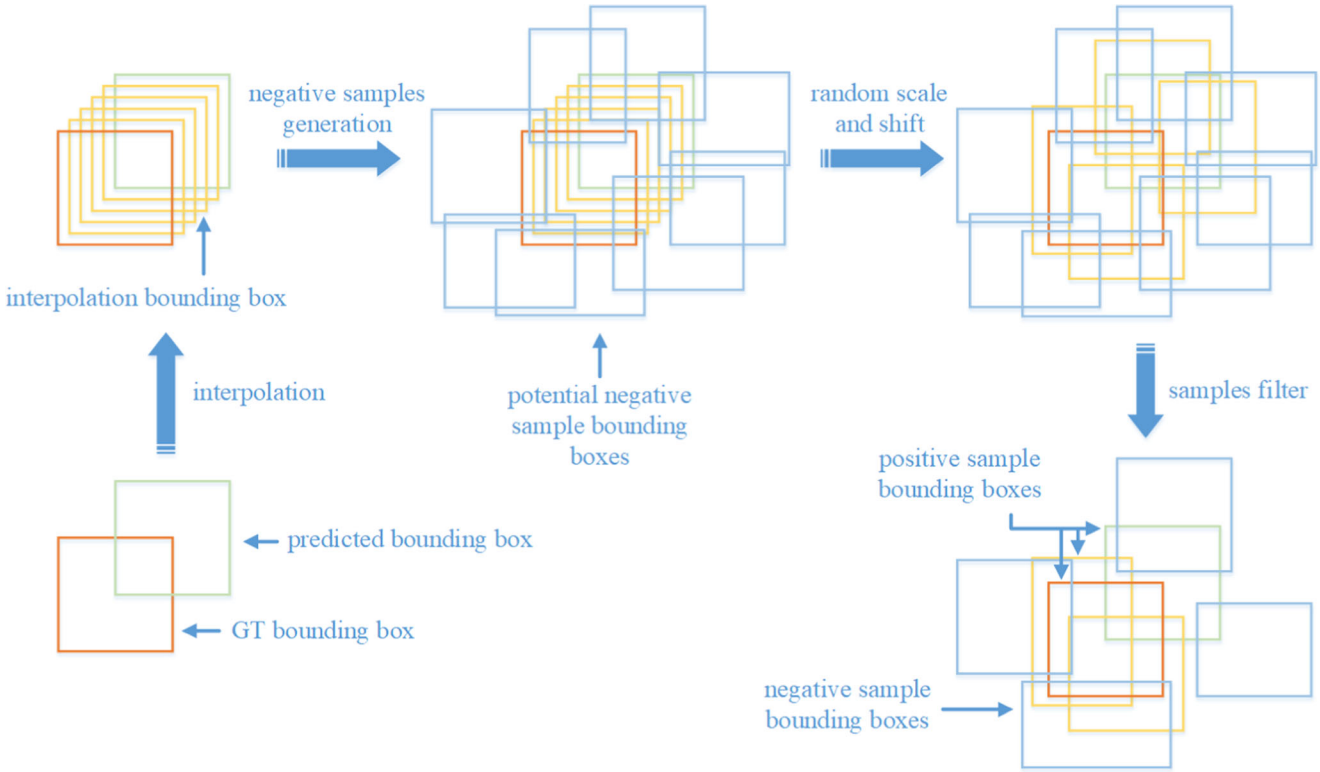


Fig. 3 The generation process of supplementary samples. Supplementary samples are generated through four steps, namely boundary box interpolation, negative sample generation, random scaling and bias, and sample filtering

After the training samples generated from both RPN and the proposal regions of motion prediction are obtained, the loss is calculated using the following formula:

$$L_{cls\_head}^{cls} = \frac{1}{N_{sam}} \sum_i L_{cls}\left(P_i, P_i^*\right) \tag{7}$$

$$L_{reg\_head}^{loc} = \frac{1}{N_{pos\_sam}} \sum_i P_i^* L_{reg}\left(t_i, t_i^*\right) \tag{8}$$

These losses are similar to those from the RPN part. Some differences are that P_i represents the probability that the category of the target in the sample boundary box is pedestrian or background, and these two terms are normalized by the sample number N_{sam} and positive sample number N_{pos_sam}, respectively.

3.3.3 The metric loss of ReID head

Tracktor [15] adopts the ReID module independent of the tracking network and conducts training with Triple Loss [38]. The training data for Triple Loss comes from random sampling in a batch. When an insufficient selection of samples for the same target is made, the temporal correlation of extracted appearance features will be affected. The same target in different frames will be distinguished into different identities to extract the appearance feature vectors with large differences. Identity switch is easy to occur in inference, which leads to the decline of the identity preservation ability of the model. During the training, we simulated the tracking process and used the ground-truth position of the target to obtain its appearance features in each frame through the ReID Head. These continuous appearance features can be used to improve the similarity of the appearance features of the same target in different frames, that is, to improve the temporal correlation of the appearance features of the target. Therefore, instead of the triplet loss based on random sampling, we adopt the metric loss that could utilize the historical appearance features of the target to train the ReID Head, which was proposed in DeepMOT [40].

As shown in Fig. 2, we will extract the appearance features f_{app} of all the targets in the current frame through the ground-truth bounding box. Then, the saved historical ROI features of each target are used to calculate the historical appearance features and the average historical appearance features f_{his_app} of these targets through the ReID Head. The distance between the appearance feature of the target in the current frame and the average historical appearance features of the saved target is calculated by the following formula:

$$dist_{app} = \frac{1}{2}\left(1 - cosine\left(f_{app}, f_{his\_app}\right)\right) \tag{9}$$

Where cosine (A, B) is the cosine distance between A and B. According to the appearance feature distance between target pairs, we get an appearance distance matrix D_{dis_app} with

the size of $N_{tar} \times N_{tar}$, where N_{tar} is the number of ground-truth targets in the current frame. The target identity on the horizontal axis and vertical axis in D_{dis_app} is also one-to-one correspondence, that is, the value of row i and column i of the matrix is the distance between the appearance feature and the historical average appearance features of the target i. Note that when calculating the appearance feature distance, we do not consider a saved target that does not exist in the current frame or a new target that appears in the current frame. Then, D_{dis_app} is input into the Deep Hungarian Network (DHN) as shown in Fig. 4 to obtain a soft assignment matrix $\widetilde{A}_{dis\_app}$.

As shown in Fig. 4, The row-wise flattened distance matrix D_{dis_app} is input to a first Bi-RNN that outputs the first-stage hidden representation of size $N_{tar} \times 2 * hidden\_unit \times N_{tar}$, where $hidden\_unit$ is the size of the Bi-RNN hidden layers. Intuitively the first-stage hidden representations encode the row-wise intermediate assignments. We then flatten the first-stage hidden representation column-wise to input to a second (different) Bi-RNN that produces the second-stage hidden representation of size $N_{tar} \times 2 * hidden\_unit \times N_{tar}$. The two Bi-RNNs have the same hidden size, but they do not share weights. To obtain the final assignments, we feed the second-stage hidden representation through three fully-connected layers (along the $2 * hidden\_unit$ dimension, i.e., independently for each element of the original D_{dis_app}). Finally, a sigmoid activation produces the optimal $N_{tar} \times N_{tar}$ soft-assignment matrix $\widetilde{A}_{dis\_app}$.

The metric loss is composed of the differentiable multi-objective tracking metric *dMOTA* and *dMOTP*, which can be calculated as follows:

$$dMOTA = 1 - \frac{\widetilde{\mathrm{FP}} + \widetilde{\mathrm{FN}} + \gamma_{metric}\widetilde{\mathrm{IDS}}}{M} \tag{10}$$

$$dMOTP = 1 - \frac{\left\| D_{dis\_app} \odot B^{TP} \right\|_1}{\left\| B^{TP} \right\|_0} \tag{11}$$

Where $\widetilde{\mathrm{FP}}$, $\widetilde{\mathrm{FN}}$, and $\widetilde{\mathrm{IDS}}$ are the soft representation of false positive, false negative, and identity switches. M represents the matched target, B^{TP} represents the binary assignment matrix corresponding to the distance matrix D_{dis_app}, and γ_{metric} represents a weight factor controlling the proportion of $\widetilde{\mathrm{IDS}}$. The calculation process of the above variables is shown in Fig. 5.

we first construct a matrix C^r by appending a column to $\widetilde{A}_{dis\_app}$, filled with a threshold value δ, and perform row-wise softmax. Analogously, we construct C^c by appending a row to $\widetilde{A}_{dis\_app}$ and perform column-wise softmax. Then, we can express a soft approximation of the number of false positives and false negatives as:

$$\widetilde{\mathrm{FP}} = \sum_{N_{\mathrm{tar}}} C^r_{N_{\mathrm{tar}}, N_{\mathrm{tar}}+1}, \widetilde{\mathrm{FN}} = \sum_{N_{\mathrm{tar}}} C^c_{N_{\mathrm{tar}}, N_{\mathrm{tar}}+1} \tag{12}$$

To calculate $\widetilde{\mathrm{IDS}}$, DeepMOT [40] constructed two additional binary matrices B^{TP} and B^{TP}_{-1}, in which non-0 elements represent the matching of the current frame and the previous frame, respectively. Then, the matching difference of the two frames is used to calculate the $\widetilde{\mathrm{IDS}}$. When calculating the distance matrix D_{dis_app}, we only consider the targets that exist in the current frame and the previous frames at the same time, and the order of the targets in D_{dis_app} and $\widetilde{A}_{dis\_app}$ corresponds to each other, that is, we already know the

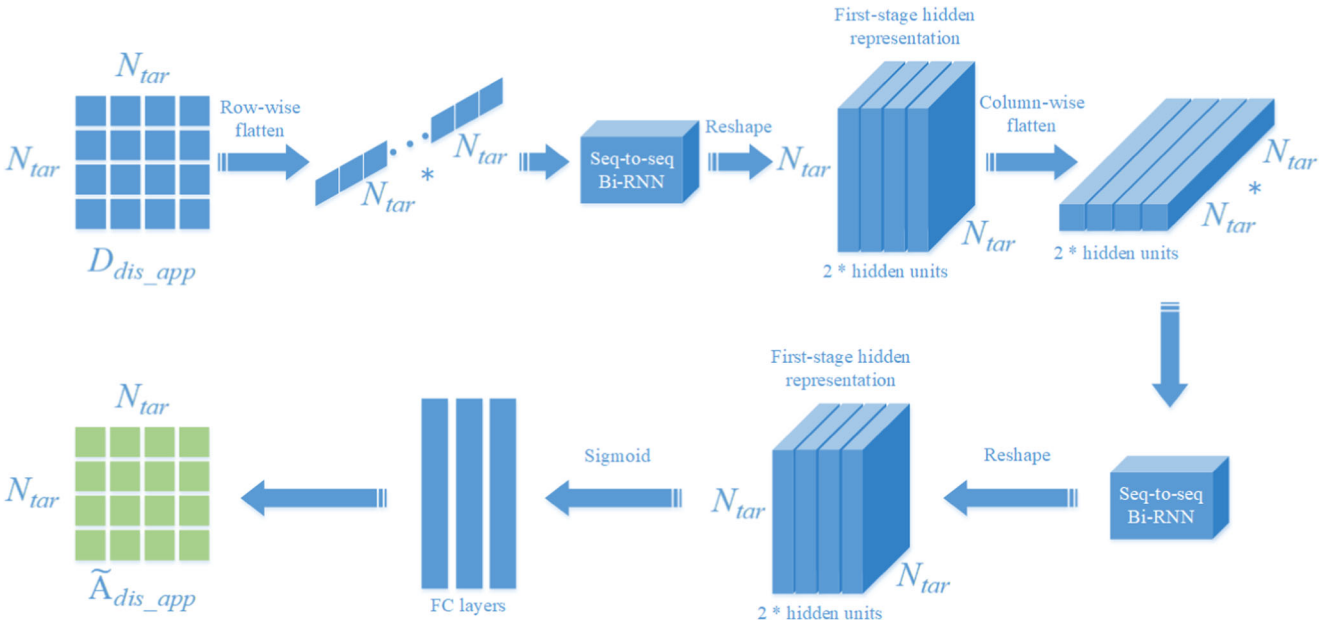


Fig. 4 Structure of the adopted Deep Hungarian Network and the forward propagation process

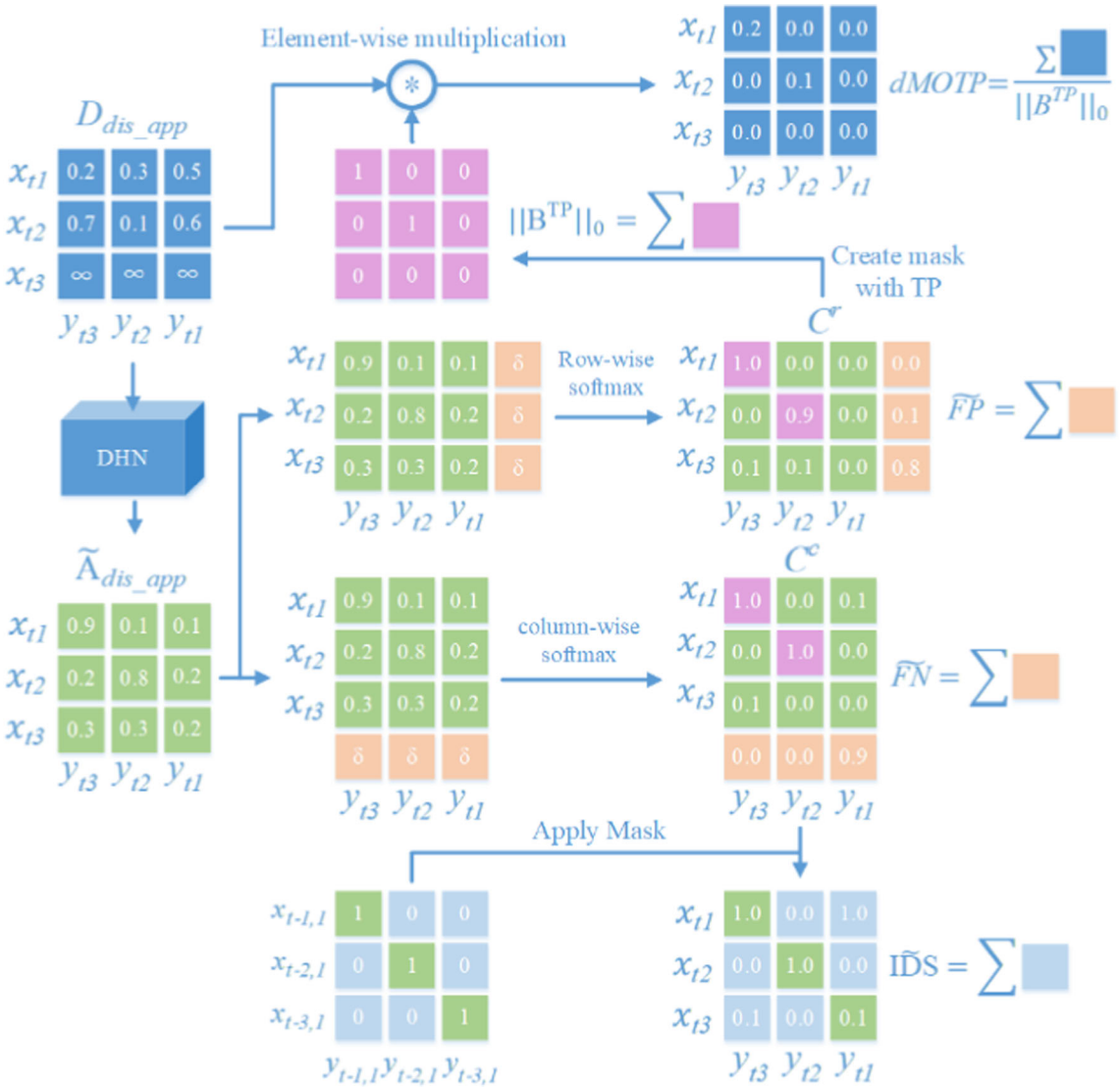


Fig. 5 The computation of differentiable *dMOTA* and *dMOTP*. *dMOTP* is computed as the average distance of matched tracks and *dMOTA* is composed with $\widetilde{FP}$, $\widetilde{FN}$, and $\widetilde{IDS}$

matching status in the previous frame. Therefore, we adopt a different method from DeepMOT [40], instead of saving the binary assignment matrix of each frame, we directly use the following formula to simplify the calculation of $\widetilde{IDS}$:

$$\widetilde{IDS} = \left\| C^{c}_{1:N_{tar},1:N_{tar}} \odot \bar{I}_{N_{tar}} \right\|_1 \tag{13}$$

Where $\|\bullet\|_1$ represents the L_1 normalization of the flattened matrix, $\bar{I}_{N_{tar}}$ represents a matrix with size $N_{tar} \times N_{tar}$, diagonal elements 0, and other elements 1. After calculating *dMOTA* and *dMOTP* according to formula (10)(11), we use the following formula to calculate the metric loss L_{metric}:

$$L_{metric} = (1-dMOTA) + \lambda_{\text{metric}}(1-dMOTP) \tag{14}$$

Where λ_{metric} represents a weighting factor that controls the proportion of sub-loss.

3.4 Re-identification

Based on the appearance features extracted from the ReID head, we design a short-term pedestrian re-identification module to keep the tracker running online. We store deactivated tracks for a fixed number of F_{ReID} frames. We then compare the distance in the appearance embedding space of the deactivated with the newly detected tracks through Eq. (9) and re-identify via a threshold τ_{ReID}. To minimize the risk of false ReID, we only consider pairs of deactivated and new detection with a sufficiently large IoU λ_{ReID}. The motion model is continuously applied to the deactivated tracks.

3.5 Track management

The tracking management includes the following 5 steps:

①Track initialization: First, we select the first frame. The filtered detection results are regarded as the tracking positions $p_t = [x, y, w, h]$ of the targets, and then they are inputted into the ReID head of the model to obtain the appearance featuref_t of the target. The tracking position p_t^{track} and appearance feature f_t are used to initialize the tracking state $s_t = \{h_{p_t}, h_{f_t}\}$ of the target, which including historical positions h_{p_t} and historical appearance features h_{f_t}. finally, a tracking set $S = \{s_t\}$, $t = 1, 2, 3 \ldots T$ is obtained, where T represents the number of targets.
②Motion prediction: the historical position h_{p_t} of each target in the tracking state is inputted into the motion model to predict the position p_t^{pre} in the current frame.
③Position refinement: the predicted position p_t^{pre} is inputted into the network model to obtain the regression coefficient $[t_x, t_y, t_w, t_h]$ from the regression head. Then, the regression coefficient is applied to the predicted position to get the refined position p_t^{refine}. This process is equivalent to single target tracking for each target.
④Data association: IOUs between the detection D and the refined position p_t^{refine} of all targets are calculated, and then a simple greedy matching is used to associate the detection and targets. The refined position p_t^{refine} of the matched target is regarded as its tracking position p_t^{track} in the current frame and added to the historical position h_{p_t}. Then, the refined position p_t^{refine} is used to extract the appearance feature of the target from the ReID head and add it to its historical appearance features. A target that does not match any detection is marked as a lost target and participates in the ReID operation described in Section 3.4.
⑤New track addition: the process same as the ① step is used to initialize a new track for the detection that does not match any target in the ④ step.

Finally, we select the next frame and repeat steps ②-⑤ until all frames are tracked.

4 Experiments

4.1 Datasets

First, the proposed method was evaluated on datasets 2DMOT2015 [52], MOT16 [53], MOT17, and MOT20 from MOTChallenge, a multi-object tracking benchmark website. These datasets contain multiple challenging pedestrian tracking sequences with frequent occlusion, crowded scenes, varying perspectives and camera movements, and varying target sizes and frame rates. Where 2DMOT2015 consist of 11 training video sequences and 11 test video sequences. MOT16 and MOT17 contain the same 7 training video sequences and 7 test video sequences. However, MOT16 only provides detection results generated from DPM [54]. MOT17 provides additional detection results based on Faster-RCNN [20] and SDP [55]. MOT20 contains 4 training video sequences and 4 test video sequences. The pedestrian density in each video sequence is much higher than the previous three dataset, which is more difficult to process. To make a fair comparison with other multi-object tracking algorithms, we only use the public detection set for tracking.

In order to illustrate the compatibility of the proposed method for different types of targets, we also carried out a tracking evaluation in the Kitti dataset. The Kitti dataset contains 21 training sequences and 29 test sequences. Like the MOTchallenge dataset, the label of the test set is not public, so the tracking results can only be uploaded to the corresponding evaluation website for evaluation. The scene of the Kitti dataset contains multiple categories of targets, and its training labels provide ground-truth bounding boxes for pedestrians, cyclists, cars, and vans. However, since the official website only provides evaluations for pedestrians and cars, we only trained the proposed model on these two categories. Unlike the dataset on the MOTchallenge, the Kitti dataset has a small number of pedestrians and many cars, so the tracking results of different categories in the same scene will also have large differences. We exploited the trained faster-RCNN model to provide detection results for tracking.

4.2 Evaluation metric

The metrics used by multi-object tracking benchmarks [56] were considered to evaluate tracking performance on datasets from the MOTchallenge, which includes multi-object tracking accuracy (MOTA) and multi-object tracking precision (MOTP). The ratio of correctly identified detection over the average number of ground-truth and computed detection (IDF1), the ratio of mostly tracked targets (MT), the ratio of mostly lost targets (ML), the number of false negatives (FN), the number of false positives (FP), the number of ID switches (IDsw) and the number of fragments (Frag).

In addition to the above indicators, the official website provides HOTA [57] metric that can better measure tracking performance, so we additionally used HOTA as the main metric for evaluation on the Kitti dataset.

4.3 Experiment detail

The proposed method used Python and Pytorch for implementation. The Faster-RCNN was pre-trained on MS COCO [58] to initialize the regression network. The weights of Classification head and ReID head were initialized by a normal distribution with mean = 0 and variance = 0.01. The weights of DHN pre-trained on MOT17 was provided by the

author of DeepMOT [40]. The scaling factors for each sub-loss during training were $\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 10$, and $\lambda_{metric} = 5$. These above scaling factors were chosen to ensure their approximate size. The threshold value for the generation of supplementary samples was $\tau_{pos} = 0.5$, which was the same as the threshold for filtering positive and negative samples in Faster-RCNN. The Adam optimizer with a learning rate of r = 1e-6 was used to train 60 epochs on a system with i9-9900 × 3.5Ghz CPU and NVIDIA GeForce GTX 2080 Ti GPU. To make the proposal regions of the RPN more dispersed and approximate the distribution of motion prediction, the RPN was stopped from training in the later 30 epochs and only the proposed network model was trained. To prevent overfitting, the noise was added to the ground-truth bounding boxes, scaled with a coefficient of 0.8 to 1.2, and randomly moved horizontally and vertically with a coefficient of 0 to 0.25. To expand the data, the image was enhanced with a probability of 0.25 from 0.8 to 1 for illumination, chroma, contrast, and sharpness, respectively. Finally, the model trained on the dataset MOT17 was used to evaluate datasets 2DMOT2015, MOT16 and MOT17, and the model trained on the dataset MOT20 was used to evaluate dataset MOT20. For kitti dataset, the model trained on the training set was used to evaluate the test set.

In inference, we chose the partial hyper-parameters used in Tracktor [15], namely, σ_{active}=0.3, $\lambda_{active} = 0.5$, and $\lambda_{new} = 0.6$. Other parameters $\tau_{ReID} = 0.12$ and $\lambda_{ReID} = 0.2$ were selected through comparison experiments to obtain a best tracking performance.

4.4 Experiment results

The tracking results of the proposed method on datasets 2DMOT2015, MOT16, MOT17, and MOT20 were evaluated on the MOT Challenge benchmark website and compared with other state-of-the-art methods. Tables 1, 2, 3 and 4 show the quantitative comparison results on 2DMOT2015, MOT16, MOT17, and MOT20, respectively (bold font represents the best result, ↑ means higher is better, ↓ means lower is better, PD indicates the publication date).

The above table shows that compared with other online algorithms or even offline algorithms that can exploit the entire video sequence, the proposed method can achieve the highest MOTA on the four benchmark datasets, confirming that the method has the most excellent synthesis performance. Meanwhile, compared with other online methods, the proposed method also achieves the highest MT and lowest ML, indicating that this method has excellent track connection ability and can improve the tracking continuity. Besides, the number of FNs in the tracking results of the proposed method is also the smallest among all tracking algorithms, which proves its excellent ability to deal with missed detection. Secondly, the proposed method also achieved the highest IDF1 score in the above four data sets and the least number of IDSWs in MOT20, showing that the proposed method has superior identity preservation capabilities. The proposed method achieves the lowest number of Frags on MOT16 and MOT20, which shows its robustness to occlusion. For other metrics, the proposed method is still better than most tracking algorithms.

Tables 5 and 6 respectively show the tracking results of the proposed method on the car and pedestrian categories on the kitti test set and the comparison with other methods.

From Table 5, we can see that the proposed method obtains the best HOTA, MT, FP, Idsw, and frag in the car category of kitti. Table 6 shows that the proposed method gets the best HOTA, MOTP, MT, ML, and FP in the pedestrian category of kitti. By comparison, it is proved that the proposed method has superior compatibility for different categories of targets.

4.5 Ablation analysis

To verify the contribution of each part of the proposed algorithm, an ablation study was set up. The experiment in the ablation study was conducted based on the MOT16 benchmark dataset. The first 50% of each video sequence was used as the training set, and the last 25% as the test set. The training lasted for 30 epochs. Other than that, the other settings for the experiment were the same as in Section 4.3.

Three baseline methods were designed to conduct the ablation analysis. The first one, Instead of the proposed fusion

Table 1 Comparison on 2DMOT2015

Mode	Method	PD	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓
Online	RAN [31]	2018	35.1	70.9	45.4	13.0	42.3	6771	32,717	**381**	1523
	STRN [5]	2019	38.1	72.1	46.6	11.5	33.4	5451	31,571	1033	2665
	DAN [39]	2019	38.3	71.1	45.60	17.6	41.2	**1290**	27,000	1648	1516
	IAT [11]	2019	38.9	70.6	–	16.6	31.5	7321	29,501	720	1440
	Tracktor [15]	2019	44.1	–	46.7	18.0	26.2	6477	26,577	1318	–
	Tracktor_v2 [15]	2019	46.6	**76.4**	47.6	18.2	27.9	4624	26,896	1290	1702
	Proposed	–	**49.2**	75.4	**52.4**	**29.1**	**24.4**	8706	**21,594**	910	**1400**

Table 2 Comparison on MOT16

Mode	Method	PD	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓
online	RAN [31]	2018	45.9	74.8	48.8	13.2	41.9	6871	91,173	648	1992
	STAM [6]	2017	46.0	74.9	50.0	14.6	43.6	6895	91,117	**473**	1422
	DMMOT [7]	2018	46.1	73.8	54.8	17.4	42.7	7909	89,874	532	1616
	D_TAMA [14]	2019	46.2	75.4	49.4	14.1	44.0	5126	92,367	598	–
	MOTDT [12]	2018	47.6	74.8	50.9	15.2	38.3	9253	85,431	792	–
	STRN [5]	2019	48.5	73.7	53.9	17.0	34.9	9038	84,178	747	2919
	IAT [11]	2019	48.8	75.7	47.2	15.8	38.1	5875	86,567	906	1116
	LSSTO [9]	2019	49.2	74.0	56.5	–	–	7187	84,875	606	–
	Tracktor [15]	2019	54.4	78.2	52.5	19.0	36.9	3280	79,149	682	–
	DeepMOT [40]	2020	54.8	77.5	53.4	19.1	37.0	2955	78,765	645	–
	Tracktor_v2 [15]	2019	56.2	**79.2**	54.9	20.7	35.8	**2394**	76,844	617	1068
	Proposed	–	**59.8**	79.0	**58.7**	**24.1**	**30.8**	3669	**68,920**	617	**1052**

motion model, ECC was used to compensate the camera motion, followed by the Kalman filter to predict the motion. Then the network model was trained based on the training method of Faster-RCNN. The second one, the proposed fusion model of ECC and the Kalman filter was adopted as the motion model, and the training method was the same as the first method. The third one, based on the second method, the proposed end-to-end training method was adopted. The fourth one, that is, the proposed method, based on the third method, added the ReID head and utilized the metric loss to train the total network model. The comparison of the results of the different methods is shown in Table 7.

As can be seen from Table 7, when the fusion model of Kalman filter and ECC was adopted, although FP was slightly increased, the number of FN and IDsw decreased more, which improved MOTA and IDF1 to a certain extent, and proved the excellent prediction performance of this fusion model. After the end-to-end training method was adopted, both MOTA and IDF1 were greatly improved, indicating that The fitting of the proposal regions during training to the distribution of motion prediction can effectively improve the refinement ability of the regression network. After adopting ReID head and adding metric loss, although it only brought slight fluctuations to the number of FPs and FNs, the number of IDsw was reduced by almost 50%. It proved that the added ReID head and the newly adopted training method could effectively solve the incompatibility between the refined bounding box and the extracted appearance features, and obtain more distinguishable appearance features, thereby improving the identity preservation ability of the model.

4.6 Compatibility analysis

To further prove that adding the ReID head to the network structure can improve the compatibility between bounding box regression and appearance feature extraction, we designed

Table 3 Comparison on MOT17

Mode	Method	PD	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓
Online	DMMOT [7]	2018	48.2	75.7	55.7	19.3	38.3	26,218	263,608	2194	5378
	D_TAMA [14]	2019	50.3	76.7	53.5	19.2	37.5	25,479	252,996	2192	–
	MOTDT [12]	2018	50.9	76.6	52.7	17.5	35.7	24,069	250,768	2474	–
	STRN [5]	2019	50.9	75.6	56.5	20.1	37.0	27,530	246,924	2593	9622
	FAMNet [41]	2019	52.0	76.5	48.7	19.1	33.4	14,138	253,616	3072	–
	DAN [39]	2019	52.4	76.9	49.4	21.4	30.7	25,423	234,592	8431	14,797
	MCSAC [9]	2019	52.7	76.2	57.9	–	–	22,512	241,936	2167	–
	Tracktor [15]	2019	53.5	78.0	52.3	19.5	36.6	12,201	248,047	2072	–
	DeepMOT [40]	2020	53.7	77.2	53.8	19.4	36.6	11,731	247,447	**1947**	–
	Tracktor_v2 [15]	2019	56.3	**78.8**	55.1	21.1	35.3	**8866**	235,449	1987	**3763**
	Proposed	–	**60.1**	78.7	**58.8**	**26.0**	**29.7**	13,523	**209,472**	2066	3830

Table 4 Comparison on MOT20

Mode	Method	PD	MOTA↑	MOTP↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDsw↓	Frag↓
online	SORT20 [27]	2016	42.7	**78.5**	45.1	16.7	26.2	27,521	264,694	4470	17,798
	D_TAMA [14]	2019	47.6	77.6	48.7	27.2	23.6	38,194	252,934	2437	3887
	Tracktor [15]	2019	51.3	76.7	47.6	24.9	26.0	**16,263**	253,680	2584	4824
	Proposed	–	**59.2**	76.8	**59.1**	**41.1**	**17.3**	36,402	**172,785**	**1914**	**3745**

Table 5 Comparison on car category of kitti dataset

Mode	Method	HOTA↑	MOTA↑	MOTP↑	MT↑	ML↓	FN↓	FP↓	Idsw↑	Frag↑
online	SASN_MCF [44]	52.24%	69.82%	**82.37%**	57.38%	8.00%	2403	7294	683	703
	FAMNet [41]	52.56%	75.92%	78.78%	52.46%	9.69%	762	7000	521	614
	CIWT [45]	54.90%	74.44%	79.32%	50.46%	11.23%	955	7346	491	552
	SCEA [46]	56.09%	74.92%	79.46%	53.69%	12.31%	1310	6993	324	317
	LP-SSVM [47]	56.62%	76.82%	78.00%	57.69%	9.23%	1247	6401	325	466
	extraCK [48]	59.76%	**79.29%**	82.06%	62.31%	**5.85%**	**675**	5929	520	750
	Proposed	**66.72%**	78.10%	81.27%	**63.38%**	6.00%	1453	**5843**	**235**	**202**

Table 6 Comparison on pedestrian category of kitti dataset

Mode	Method	HOTA↑	MOTA↑	MOTP↑	MT↑	ML↓	FN↓	FP↓	Idsw↑	Frag↑
online	LP-SSVM [47]	33.74%	**43.42%**	70.23%	21.99%	35.05%	2083	10,730	284	736
	CIWT [45]	33.93%	42.10%	71.15%	14.09%	35.05%	1149	11,821	433	851
	SCEA [46]	34.66%	43.26%	71.57%	17.87%	43.64%	**1144**	11,752	**239**	**491**
	Proposed	**41.38%**	43.26%	**71.91%**	**25.77%**	**23.71%**	2575	**10,145**	416	658

two models to compare the differences between the appearance features extracted during the tracking process. One is the proposed model that integrates classification, regression, and extraction of appearance features, which is labeled ReID head. The other uses a separate ReID network to extract appearance features, which is trained separately through triplet loss [38] and marked as ReID net. Similar to the previous ablation experiment, we train these two models on the first 50% of MOT16 and perform the tracking task on the last 25% of MOT16. Then the appearance features of the target in the tracking process are saved.

First, we calculated the average similarity between the appearance features of targets and their mean appearance features as the intra-class similarity of the sequence and then calculated the average of the similarity between mean appearance features of the target as the inter-class similarity of the sequence. The comparison of the intra-class similarity, the inter-class similarity, and the difference in similarity between the two models are shown in the figure below.

As shown in sub-figure(a) of Fig. 6, in all the sequences of MOT16, the in-class similarity of the appearance features extracted from the model ReID head is slightly higher than that

Table 7 Comparison of results from ablation analysis

method	Kalman&ECC	end-to-end training	ReID head + metric loss	MOTA↑	IDF1↑	FP↓	FN ↓	IDSW↓
baseline				44.60%	57.30%	1627	12,836	207
	√			45.40%	58.50%	1644	12,692	137
	√	√		51.80%	62.40%	711	11,940	122
	√	√	√	52.10%	63.50%	668	11,965	65

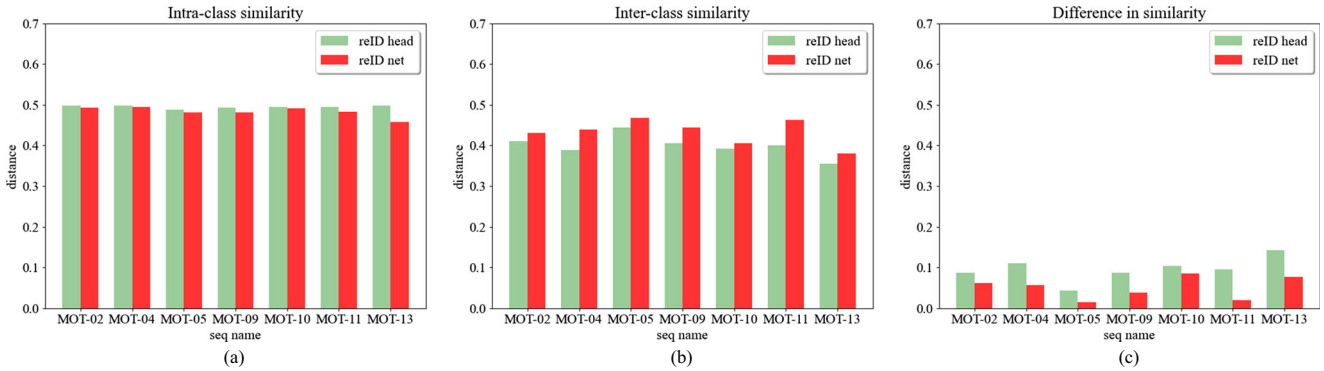


Fig. 6 The comparison of the intra-class similarity, the inter-class similarity, and the difference in similarity between the two models. **a** The intra-class similarity; **b** The inter-class similarity; **c** The difference in similarity

of the model ReID net. This proves that the former can extract appearance features that better represent the same target. Sub-figure (b) shows that the inter-class similarity of the appearance features extracted from the model ReID head is also higher than that of the model ReID net. This demonstrates that the model ReID head can provide more distinguishing target appearance features. Sub-figure (c) shows the comparison of the difference in similarity between the two models. Obviously,

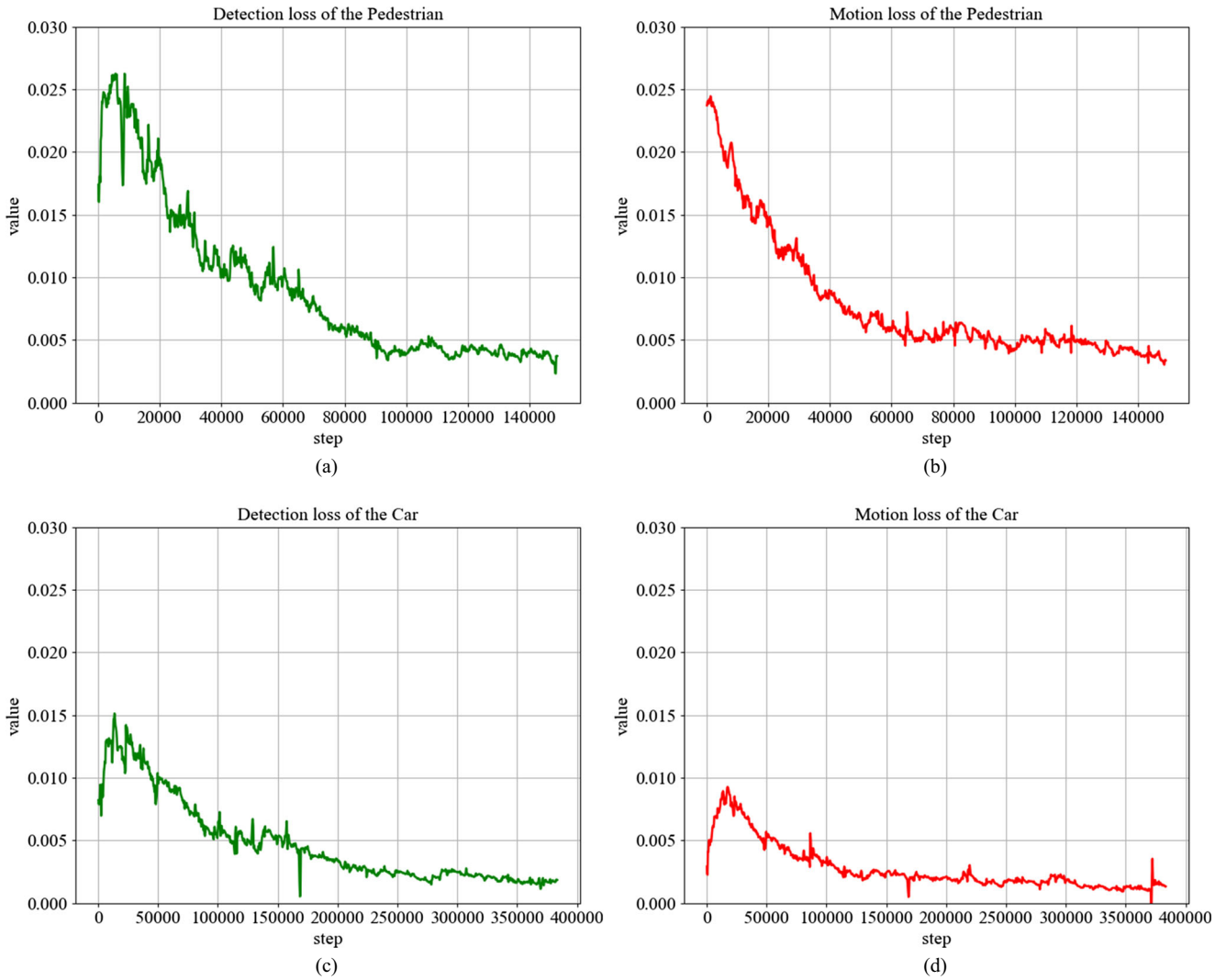


Fig. 7 The convergence process of the loss of the proposed model when it is trained on the kitti training set. **a** The detection loss of the pedestrian; **b** The motion loss of the pedestrian; **c** The detection loss of the car; **d** The motion loss of the car

the model ReID head has a higher similarity difference. Through the comparison of the above three sub-graphs, it can be shown that adding ReID head to the network and training together can improve the compatibility between the modules, and get more effective, that is, the appearance features with higher intra-class similarity and lower inter-class similarity.

4.7 Training loss analysis

In order to analyze the difference between different types of data during training and understand whether the training samples from the motion prediction during the tracking simulation training conflict with those provided in the RPN, we recorded the loss during the training process and provided the convergence process of it in the figure below. The loss calculated on the training samples provided by the RPN in the detection model is marked as detection loss, and the loss calculated on the training samples from the motion prediction is marked as motion loss.

By comparing sub-figure (a) and (b), (c) and (d) in Fig. 7, we find that the losses from different training samples in the same category follow almost the same convergence trend, which shows that the two types of training samples do not conflict, and proves that the proposed tracking simulation training is feasible. By comparing sub-figure a and c, b and d, we find that there are certain differences in training losses in different categories, that is, the loss of cars is smaller than that of pedestrians. This is firstly caused by the difference in the target category itself. For example, Intuitively, cars have a larger volume than pedestrians and are therefore easier to identify from the background. The second is that the number of vehicles in the dataset largely exceeds the pedestrians, resulting in more training samples of cars to update the network model.

4.8 Visualization results

Figures 8, 9, 10, and 11 provide the visual tracking results of the proposed method on different datasets, and qualitatively illustrate the excellent performance of the proposed method.

Figure 8 shows the tracking results of the proposed method on the video sequence ADL-Rundle-6 of the 2DMOT2015 dataset. The pedestrian with identity 4 is occluded by other pedestrians at frame 45 and was invisible, and his identity did

Fig. 8 The tracking result on video sequence ADL-Rundle-6 of 2DMOT2015

Fig. 9 The tracking result on video sequence MOT16–10 of MOT16

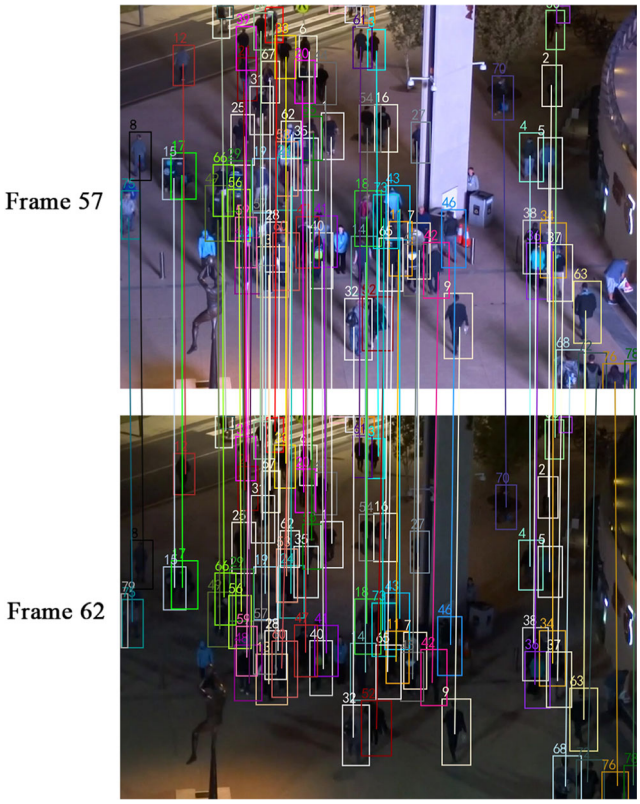


Fig. 10 The tracking result on video sequence MOT20–03 of MOT20

not change when he reappeared at frame 60. It is proved that the proposed method has high robustness to occlusion, thanks to the appearance extraction branch fused into the main network can accurately extract the appearance features with more inter-class differences and intra-class similarity from the refined bounding box.

Figure 9 shows the tracking result of the proposed method on the video sequence MOT16–10 of the MOT16 dataset, There is a large amount of camera motion and severe motion blur in this video sequence. Pedestrian with identity 6 gets increasingly close to the camera with the movement from frame 29 to frame 89, and its size in the image is also getting increasingly bigger. The proposed method can still accurately locate its position. This also proves the robustness of the proposed method for pedestrian motion, camera motion, and fuzzy image, which is benefited from the preprocessing combining Kalman Filter and ECC motion model and bounding box refinement.

Figure 10 shows the tracking result of the proposed method on the video sequence MOT20–03 of the MOT20 dataset. Compared with the previous two datasets, the scene of this video sequence is denser in terms of the number of pedestrians, and the lighting changes greatly in the two provided frames. However, as can be seen from Fig. 8, the proposed method can still accurately track targets in the scene and maintain accurate identity information, which proves the superior tracking performance of the proposed method in complex scenes and its robustness to light changes.

Figure 11 shows the tracking result of the proposed method on the video sequence 0022 of the kitti test set. From the right column of images, that is, the visual results of pedestrian tracking, it can be seen that both pedestrians 2 and 3 with small displacements and pedestrians 9 and 14 with large displacements can be accurately tracked, indicating that the proposed method can be effective Deal with the different target moving speeds. From the left column of images, that is, in the visual results of car tracking, car 6 can still be tracked effectively even if it is blocked by pedestrians, which shows the

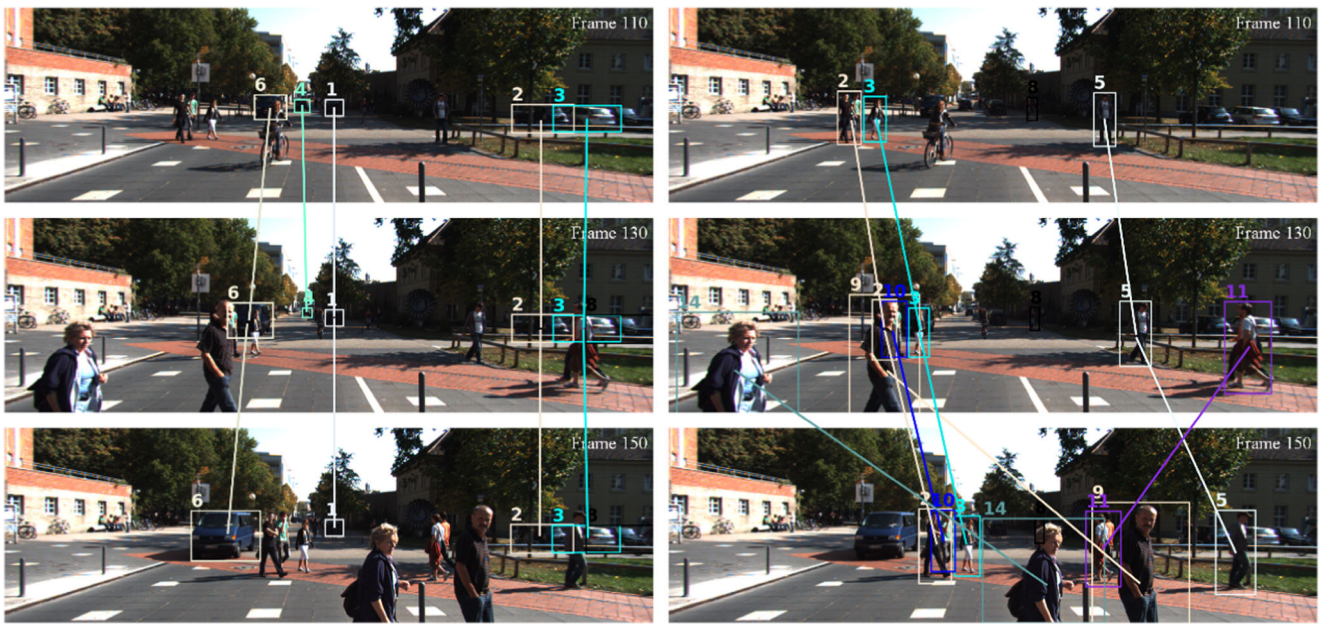


Fig. 11 The tracking result on video sequence 0022 of kitti test set

robustness of the proposed method to occlusion. In the same scenario, different types of targets can be accurately tracked, which shows that the proposed method has good compatibility for different types of targets.

5 Discussion

It can be seen from the experimental results in the above section that the proposed method has the following advantages: ①The designed network model can perform classification, regression, and appearance feature extraction at the same time. In addition to the classification loss and regression loss, the indicator loss is used to achieve Simultaneous training of the three modules. First, it saves the space required by the extra appearance feature extraction network. Secondly, sharing a backbone and simultaneous training makes the classification and regression and the appearance feature extraction module have higher compatibility so that the whole algorithm has better tracking performance. ②The proposed method uses Kalman filter & ECC as the motion model, and simultaneously solves the position difference caused by the target motion and camera motion, improves the prediction accuracy, and improves the tracking performance. ③The proposed algorithm exploits tracking simulation training. That is, the training samples are expanded by motion prediction during the training process to solve the problem of the mismatch of proposal bounding boxes selected during training and test, thereby further improving the tracking ability of the algorithm.

In addition to the above advantages, the method proposed in this paper also has the following shortcomings to be solved:①Kalman filter&ECC can effectively improve the tracking performance. However, it is still a linear prediction model, and its improvement effect is limited in the case of multi-frame loss.②The network model has more functional modules, and more loss functions and training samples are used during training, so it may not have a high training efficiency.

In addition to the baseline, there are several methods similar to the proposed method. For example, SORT [27] and DeepSORT [28] both exploit the Kalman filter as a motion model, and DeepSORT also applies additional networks to provide appearance features to participate in matching. However, the proposed method further utilizes the Kalman filter&ECC as the motion model to simultaneously solve the target movement and camera motion. In addition, the appearance features in the proposed method come from a network branch, which can be trained with other modules through metric loss to obtain effective appearance features. The comparison of the tracking results with the proposed method on MOT20 proves that the latter has better performance than the above two methods. DAN [39] also applies an end-to-end training method, but it only combines the association matching during training to optimize the calculation of affinity score. There is no appropriate processing when the target is occluded, resulting in the number of FPs in its tracking results far more than the proposed method. DeepMOT [40] also employs metric loss based on Tracktor [15]. Still, it does not apply a better motion model for position prediction. No fit candidate samples in the tracking process during training are collected, resulting in poorer tracking performance on MOT16 and MOT17 than the proposed method.

6 Conclusion

In this paper, we propose an algorithm based on a deep neural network for online multi-object tracking. Our contribution is mainly reflected in three aspects: ①In order to improve the prediction accuracy in the tracking process, we adopted the Kalman filter&ECC as the motion model and exploited ablation experiments to prove the effectiveness of this improvement. ②In order to solve the compatibility problem between classification, regression and appearance feature extraction, we designed a network model that aggregates multiple functions and added metric loss to realize the simultaneous training of the appearance extraction module and other modules. We have proved that this design can improve tracking performance and get more effective appearance characteristics in ablation analysis and compatibility analysis. ③In order to solve the problem of the mismatch between the proposal bounding boxes during training and test, we designed a tracking simulation training method. By using the motion model to predict the position and expand the training samples during training, the training data is fitted to the actual tracking data, thereby improving tracking performance. We also proved the feasibility and effectiveness of this strategy in training loss analysis and ablation analysis. In order to prove the compatibility of the proposed method for different types of targets, we also conducted experiments on different types of targets in the Kitti data set and achieved excellent tracking results.

Although the proposed method has achieved superior tracking performance, there are still some shortcomings worthy of improvement. Therefore, we provide some future development trends and research directions. First, the current MOT algorithm can only track a single type of target, so tracking multiple objects of multiple categories in the same scene simultaneously maybe the main trend of future development. Secondly, most of the current MOT algorithms are based on 2D bounding boxes. In some applications that need to know the target depth information, it can not provide more information. Therefore, 3D object tracking is also an interesting research direction. Third, multi-target tracking for multiple data sources such as radar information, point cloud

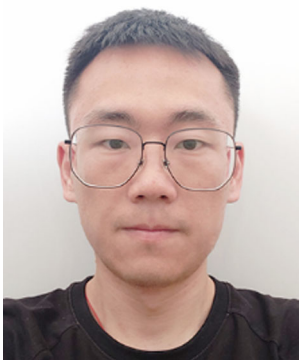
information, or multiple camera information is also a direction worth studying.

Acknowledgements This paper was supported by the Graduate Innovation Foundation of Jiangsu Province [grant No. KYLX16_0781]; the Natural Science Foundation of Jiangsu Province [grants No. BK20181340];the 111 Project [grants No. B12018]; PAPD of Jiangsu Higher Education Institutions; National Natural Science Foundation of China [grants No. 61806006]; China Postdoctoral Science Foundation [Grant No. 2019 M660149].

References

1. Kim C, Li F, Ciptadi A, Rehg JM (2015) Multiple hypothesis tracking revisited, in: Proceedings of the IEEE international conference on computer vision, pp. 4696–4704
2. Bae S-H, Yoon K-J (2014) Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1218–1225
3. Lenz P, Geiger A, Urtasun R (2015) Followme: Efficient online min-cost flow tracking with bounded memory and computation, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 4364–4372
4. Wu Z, Thangali A, Sclaroff S, Betke M (2012) Coupling detection and data association for multiple object tracking, in: proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1948-1955
5. Xu J, Cao Y, Zhang Z, Hu H (2019) Spatial-temporal relation networks for multi-object tracking, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 3988–3998
6. Chu Q, Ouyang W, Li H, Wang X, Liu B, Yu N (2017) Online multi-object tracking using CNN-based single object tracker with spatial-temporal attention mechanism, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 4836–4845
7. Zhu J, Yang H, Liu N, Kim M, Zhang W, Yang M-H (2018) Online multi-object tracking with dual matching attention networks, in: Proceedings of the European Conference on Computer Vision, pp. 366–382
8. Danelljan M, Bhat G, Shahbaz Khan F, Felsberg M (2017) Eco: Efficient convolution operators for tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6638–6646
9. Feng W, Hu Z, Wu W, Yan J, Ouyang W (2019) Multi-object tracking with multiple cues and switcher-aware classification, arXiv:1901.06129
10. Li B, Yan J, Wu W, Zhu Z, Hu X (2018) High performance visual tracking with siamese region proposal network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8971–8980
11. Chu P, Fan H, Tan CC, Ling H (2019) Online multi-object tracking with instance-aware tracker and dynamic model refreshment, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 161–170
12. Chen L, Ai H, Zhuang Z, Shang C (2018) Real-Time Multiple People Tracking with Deeply Learned Candidate Selection and Person Re-Identification, in: Proceedings of the IEEE International Conference on Multimedia and Expo (ICME), 1–6
13. Yoon Y-C, Boragule A, Song Y-M, Yoon K, Jeon M (2018) Online multi-object tracking with historical appearance matching and scene adaptive detection filtering, in: Proceedings of the IEEE International conference on advanced video and signal based surveillance, pp. 1–6
14. Yoon Y-C, Kim DY, Yoon K, Song Y-m, Jeon M (2019) Online multiple pedestrian tracking using deep temporal appearance matching association, arXiv:1907.00831
15. Bergmann P, Meinhardt T, Leal-Taixe L (2019) Tracking without bells and whistles, in: Proceedings of the IEEE international conference on computer vision, pp. 941–951
16. Kalman RE (1960) A new approach to linear filtering and prediction problems. ASME J Basic Eng March 82(1):35–45
17. Evangelidis GD, Psarakis EZ (2008) Parametric image alignment using enhanced correlation coefficient maximization. IEEE Trans Pattern Anal Mach Intell 30(10):1858–1865
18. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587
19. Girshick R (2015) Fast r-cnn, in: Proceedings of the IEEE international conference on computer vision, pp. 1440–1448
20. Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks, in: Proceedings of the Advances in neural information processing systems, pp. 91–99
21. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788
22. Redmon J, Farhadi A (2017) YOLO9000: better, faster, stronger, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 7263–7271
23. Redmon J, Farhadi A (2018) Yolov3: An incremental improvement, arXiv:1804.02767
24. Feichtenhofer C, Pinz A, Zisserman A (2017) Detect to track and track to detect, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 3038–3046
25. Kieritz H, Hubner W, Arens M (2018) Joint detection and online multi-object tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1459–1467
26. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC (2016) Ssd: Single shot multibox detector, in: Proceedings of the European Conference on Computer Vision, pp. 21–37
27. Bewley A, Ge Z, Ott L, Ramos F, Upcroft B (2016) Simple online and realtime tracking, in: Proceedings of the IEEE International Conference on Image Processing, pp. 3464–3468
28. Wojke N, Bewley A, Paulus D (2017) Simple online and realtime tracking with a deep association metric, in: Proceedings of the IEEE International Conference on Image Processing, pp. 3645–3649
29. Huang P, Han S, Zhao J, Liu D, Wang H, Yu E, Kot AC (2020) Refinements in motion and appearance for online multi-object tracking, arXiv:2003.07177
30. Milan A, Rezatofighi SH, Dick A, ReID I, Schindler K (2016) Online multi-object tracking using recurrent neural networks, arXiv:1604.03635
31. Fang K, Xiang Y, Li X, Savarese S (2018) Recurrent autoregressive networks for online multi-object tracking, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 466–475
32. Takala V, Pietikainen M (2007) Multi-object tracking using color, texture and motion, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–7
33. Yang M, Jia YJCV, Understanding I (2016) Temporal dynamic appearance modeling for online multi-person tracking. Comput Vis Image Underst 153:16–28
34. Wang L, Xu L, Kim MY, Rigazico L, Yang M-H (2017) Online multiple object tracking via flow and convolutional features, in:

Proceedings of the IEEE International Conference on Image Processing, pp. 3630–3634
35. Yu F, Li W, Li Q, Liu Y, Shi X, Yan J (2016) Poi: Multiple object tracking with high performance detection and appearance feature, in: Proceedings of the European Conference on Computer Vision, pp. 36–42
36. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1–9
37. Mahmoudi N, Ahadi SM, Rahmati M (2019) Multi-object tracking using CNN-based features: CNNMTT. Multimed. Tools Appl 78(6):7077–7096
38. Hermans A, Beyer L, Leibe B.J.a.p.a. (2017) In defense of the triplet loss for person re-identification, arXiv:1703.07737
39. Sun S, Akhtar N, Song H, Mian AS, Shah M (2019) Deep affinity network for multiple object tracking. IEEE Trans Pattern Anal Mach Intell:1
40. Xu Y, Osep A, Ban Y, Horaud R, Leal-Taixé L, Alameda-Pineda X (2020) How to train your deep multi-object tracker, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6787–6796
41. Chu P, Ling H (2019) Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 6172–6181
42. Shi X, Ling H, Pang Y, Hu W, Chu P, Xing J (2019) Rank-1 tensor approximation for high-order association in multi-object tracking. Int J Comput Vis 127(8):1063–1083
43. G. Brasó, L. Leal-Taixé (2020) Learning a neural solver for multiple object tracking, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6247–6257
44. Gündüz G, Acarman T (2019) Efficient multi-object tracking by strong associations on temporal window. IEEE Transactions on Intelligent Vehicles 4(3):447–455
45. Osep A, Mehner W, Mathias M, Leibe B (2017) Combined image- and world-space tracking in traffic scenes. In 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 1988–1995
46. Yoon JH, Lee CR, Yang MH, Yoon KJ (2016) Online multi-object tracking via structural constraint event aggregation. In Proceedings of the IEEE Conference on computer vision and pattern recognition, pp. 1392–1400
47. Wang S, Fowlkes CC (2017) Learning optimal parameters for multi-target tracking with contextual interactions. International journal of computer vision 122(3):484–501
48. Gündüz G, Acarman T (2018) A lightweight online multiple object vehicle tracking method. In 2018 IEEE Intelligent Vehicles Symposium (IV), pp. 427–432
49. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778
50. Lin T-Y, Dollár P, Girshick R, He K, Hariharan B, Belongie S (2017) Feature pyramid networks for object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2117–2125
51. Ioffe S, Szegedy C (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv: 1502.03167
52. Leal-Taixé L, Milan A, ReID I, Roth S, Schindler K (2015) Motchallenge 2015: Towards a benchmark for multi-object tracking, arXiv:1504.01942
53. Milan A, Leal-Taixé L, ReID I, Roth S, Schindler K (2016) MOT16: A benchmark for multi-object tracking, arXiv: 1603.00831
54. Felzenszwalb PF, Girshick RB, McAllester D, Ramanan D (2009) Object detection with discriminatively trained part-based models. IEEE Trans Pattern Anal Mach Intell 32(9):1627–1645
55. Yang F, Choi W, Lin Y (2016) 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, pp. 2129–2137
56. Bernardin K, Stiefelhagen R (2008) Evaluating multiple object tracking performance: the CLEAR MOT metrics. EURASIP Journal on Image and Video Processing 2008:1–10
57. Luiten J, Osep A, Dendorfer P, Torr P, Geiger A, Leal-Taixé L, Leibe B (2020) HOTA: a higher order metric for evaluating multi-object tracking. Int J Comput Vis:1–31
58. Lin TY, Maire M, Belongie S, James P, Perona P, Ramanan D, Piotr D, Zitnick CL (2014). Microsoft Coco: Common Objects in Context. in: Proceedings of the European Conference on Computer Vision, pp. 740–755



Jieming Yang was born in 1995. He received the B.S. degree in automation fromJiangnan University, China, in 2017 and He is now an MD-PhD based in controlscience and engineering from Jiangnan University, China. His research interestsinclude artificial intelligence, machine learning, image process, visual objecttracking and their applications.

Ge Hongwei was born in 1967. Hereceived the M.S. degree in computer science from Nanjing University ofAeronautics and Astronautics, China, in 1992 and the Ph.D. degreein control engineering from Jiangnan University, China, in 2008. Currently, heis a professor and Ph.D. supervisor in the School of Internet of Things ofJiangnan University. His research interests include artificial intelligence,machine learning, pattern recognition and their applications.

Jinlong Yang received the M.S.degree inCircuit and System from Northwest Normal University, China in 2009, and the Ph.D. degree inPattern Recognition andIntelligent System from Xidian University, China in 2012. He was a visiting scholar in theDepartment of Electrical and Computer Engineering, University of Wisconsin-Madisonfrom December 2016 to December 2017. He is currently an associate professor ofthe School of Internet of Things Engineering in Jiangnan University. Hisresearch interests include visual object tracking, pattern recognition andsignal processing.

Yubing Tong received the Ph.D.degree in Beihang University, Beijing, China. He currently works with Prof.Udupa, JK at Medical Image Processing Group (MIPG) at the University ofPennsylvania. His research interests include Medical Image Processing &Analysis, Visual perception and image retrieval with machine learning techniques, Videocompress and embedded system application & development.

Shuzhi Su is an associateprofessor with the School of Computer Science and Engineering, Anhui Universityof Science & Technology, China. He received the Ph.D. degree in the Schoolof Internet of Things Engineering, Jiangnan University. His research interestsinclude multimodal pattern recognition, information fusion, feature learning,and image processing.