Online Layered Multiple Object Tracking Using Residual-Residual Networks

Bo-Cheng Jiang and Chung-Nan Lee*
National Sun Yat-sen University, Taiwan
*E-mail: cnlee@mail.cse.nsysu.edu.tw Tel: + 886-7-5252000 ext. 4313

***Abstract*—When dealing with multiple object tracking in the real world, it faces several challenges: (a) The number of targets to be tracked will change over time, (b) The data association of the target at different times will be affected by occlusion, (c) The problem of estimating the continuous state of all targets and deciding whether the targets leave the screen and then stop tracking. In this paper, a novel multiple object tracking method that consists of a residual-residual network and a four-layer data association scheme. The residual-residual network combines a deep residual classification network and a deep residual feature network. The deep residual classification network is used to remove unwanted background noise from the frame and corrects the target position of the missing ones. It can accurately track the position of multiple targets, link their positions in each time period, combine layered target data association method, and stepwise pair the trajectories and target candidates according the features generated from the deep residual feature network. The experiments using MOT16 that is a multi-object tracking database, show that the proposed method leads most existing researches in several evaluation criteria including the accuracy, speed and false positive.**

Keywords: Object Detection, Multiple Object Tracking, Residual Network, Person Re-identification, Data Association

I. INTRODUCTION

In an unconstrained environment, tracking multiple targets is a challenging issue, and even after decades of research, it is still far less accurate than human visual judgment. The definition of multi-object tracking consists of two points, (a) locating all the objects of interest in the film or image, and marking each target, (b) keeping the target's tagged message as time progresses. Different viewing angles, different brightness or occlusion levels and an unfixed number of targets can make tracking difficult. The main problem with tracking algorithms is how to overcome these difficulties and establish a general tracking model for each different video scenario, different background and different targets.

Multi-object tracking has been studied for a long time and has been applied in many fields, such as human behavior analysis, automatic car driving, video surveillance and so on. However, despite significant progress in recent years, multi-object tracking methods still have serious occlusion and noise interference in complex scenes. And we use tracking-by-detection, one of the most effective strategies, to filter out unwanted messages in the screen, leaving only the target information of interest. The candidates at different times are linked to the corresponding targets, and data associations are generated for each target at different times. In addition, because of the complex and unknown number of targets, the generation of false alarm, the start of tracking of new targets entering the scene, and the termination of tracking of targets leaving the scene, these factors must be taken into account.

Now deep learning technology has solved various problems in the field of computer vision, such as image classification [9, 20], semantic segmentation [14, 16] and single object tracking [5, 15]. Nevertheless, the multi-object tracking [1, 6, 21] using deep learning is still affected by the following reasons: Firstly, the training data for multi-object tracking is not enough. It is necessary to train a neural network with a large number of parameters. However, there are only a few training databases at present; in addition, when using the pre-trained model learned by the object classification network, the tracking will lose the ability to distinguish different targets with only nuances, and it is easy to lose the action feature of the target. Therefore, we would like to measure the targets and predict the trajectory through the double neural networks to achieve effective tracking results.

The contribution of the paper as follows:

1. Use the double neural networks structure to compare the probability weights and deep features of the target to be tracked, and the two networks assist each other to perform accurate target tracking.
2. The classification network and tracklets confidence are used to enhance tracklet prediction and the Kalman filter is used to predict the target position and determine the stability of the tracklet based on the tracklet confidence.
3. The deep residual convolutional layer as the basenet is adopted to improve the validity of the feature, and simultaneously classify multiple targets in the classification network by ROI pooling to minimize the computational time caused by the deep network.
4. A four-layer data association scheme is used to pair tracking targets by their priorities. The scheme conducts the data association layer by layer to make the pairing more accurate.

II. RELATED WORK

With the rapid advancement of deep learning, image processing has become more accurate and faster It has been applied in many computer vision fields. In this section, we review some related researches on Basenet, Object Detection and Multi-Object Tracking.

A. Basenet

Deep neural networks is indispensable for the design and evolution of the basenet. It is of paramount importance to design a highly efficient neural network structure. GoogLeNet [10] increases the depth of the network and improves the structure of the network. It uses a structure called the Inception Module, which makes the structure complexity much lower than the simple convolution layer stacking. SENet [17] introduced a SE unit (Squeeze-and-Excitation unit) to improve network efficiency and further reduce the cost of computing parameters.

SqueezeNet [13] uses the Fire Module to change each convolution layer to a squeeze layer plus an expand layer, which can significantly reduce the number of parameters and operations while maintaining accuracy. In Resnet [20] it indicates that residual networks are easier to optimize and have better accuracy from considerably increased depth. Since the combination of multiple nonlinear layers in the network approximates a complex function, it can also be assumed that the residual of the hidden layer approximates a complex function, so that it is better to obtain the residual expression of the hidden layer.

B. Detection

Currently most of accurate object detection methods use two-stage with proposal driven design. The concept is extended from the structure of R-CNN [25]. First, it generates a sparse candidate location set, and then uses a convolutional neural network to distinguish each candidate location into a foreground or background and finally determines the foreground category. Despite the continuous technological evolution in recent years, the two-stage approach is still a popular one. Though the success of the two-stage detection it encounters some problems. For example, with the assumption of a large number of candidate regions, the amount of computation in the second stage is very large.

Recently a variety of single-stage methods are proposed, they do not rely on the assumptions of candidate regions to solve the problem. For example, SSD [31] is a fast single-stage detection model and uses different resolutions of the feature map in the neural network's forward process, it can detect targets that are not limited to a specific range of size, and its performance is even faster than Faster-RCNN, and achieves good detection results in accuracy.

YOLO [18, 19] is another fast single-stage detection method that has also achieved outstanding results. It uses grid cells to divide and detect the corresponding target position. Each square is used for dense surrounding detection according to different scales and sizes, and determines the position and category of the object. With the progress of single-phase detection methods such as YOLO and SSD, the faster single-stage method is more vigorous than the two-stage detection.

C. Multi-Object Tracking

The early multi-object tracking algorithm, when dealing with the data association problem, uses the assumed object pairing for the target of detection between two consecutive pictures, and measures the similarity by appearance, position, size, etc. [8, 28, 32]. However, when one only uses the data association of regional information the tracking algorithm is easily affected by occlusion or noise to produce a broken, occasionally interrupted trajectory. So, in contrast, some multi-object tracking methods generate better target trajectories by obtaining global information, that is, the entire movie, or using multiple images in a short time [2, 3, 32].

Now with the development of deep learning, several multi-object tracking methods based on convolutional neural networks and recurrent neural networks have emerged. Compared with artificial features, deep neural networks have considerable advantages. Leal-Taixe et al. used Siamese CNN [22] to learn features, then RGB images and optical flow diagrams as multi-model inputs, and used Gradient Boosting combined with image local features and context feature extracted by CNN.

Wang et al. [6] re-derived and improved Siamese CNN that mixes joint learning and temporal constraint metric to obtain the appearance of the affinity model. In addition, the use of long short-term memory models [1] for instant multi-object tracking, is the first complete end-to-end learning network based on deep learning, but its accuracy has not yet reached a good performance. Kim et al. [11] used the deep features pre-trained by large datasets as the appearance of multi-object tracking.

The use of tracking-by-detection is also a trend for multi-object tracking. These methods define multi-object tracking as a data association problem. The main function is to connect the detection result to the tracking target [4, 26, 32]. Breitenstein et al. [24] used the continuous confidence of detection results and the appearance model of online training to conduct multi-person tracking with resistance. This concept of using online training appearance model to deal with mutual occlusion between targets has also come in recent years and been more popular [7, 30].

But most of these tracking methods are time-consuming. Moreover, in the scene with noise and obstacles, they are also easily interfered. Our proposed method is not only faster than most of them, but also good for resisting noise. The experiment in MOT16 showed the false positives of the proposed method are obviously much lower than others.

III. PROPOSED METHOD

This section describes the proposed multi-object tracking method. It uses tracking-by-detection strategy to take the video through the detector and take out the trajectory information of all target bounding boxes, and then uses another independent tracker structure for the data association to make the structure more flexible. In general, the detection and the tracking processes are separated to make the tracking result more significant.

The proposed multi-object tracking method is shown in Figure 1. It consists of a candidate location prediction unit, a

classification unit, a feature generation unit and a data association unit. When the detector sends in to detect the position of the pedestrian, all region of interests(ROIs) will be generated. The value of confidence is passed to the tracker, which is judged by the candidate location prediction unit. Based on the past ROI information, Kalman filter predicts the most likely location of the target, including some targets that are missed by the detector. Then, the non-target noise is then filtered by the classification unit to generate candidates. Finally, the object deep feature is generated for each candidate by feature generation unit and the four-layer data association is performed using the features of tracklets and the candidates.

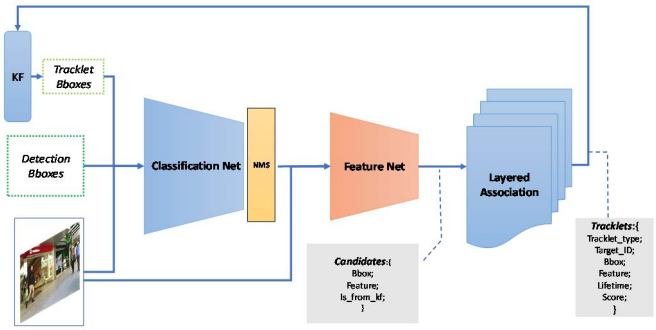

Figure 1. Architecture of the proposed multi-object tracking method

A. Tracklet Prediction

When entering a new frame, we will predict the current position of each candidate. This prediction method is based on the Kalman filter, which uses the inertia of the target's movement to process the trajectory changes during the interlace and predicts their position. We predict the positions by the Kalman filter, so that in the crowded scene, the tracklets can resist the occlusion in a short time. But this is not suitable for long-term tracking, and the accuracy will gradually decrease or even be seriously misjudged with time. Therefore, the position of the candidates sent by the detector is used to correct the prediction from the Kalman filter, and determine which position is correct will be based on the IOU and the feature distance of the predicted position and the actual position.

The stability of each trajectory changes continuously with time. After correlating adjacent ROIs over a period of time and generating candidate positions for successive frames, we describe them as trajectories. Each track may be suddenly interrupted and the target is lost. At this time, we will mark the segment of the trajectory as lost, keep trying to connect to the newly generated tracklet, and initialize a new Kalman filter from the lost state, and then use the latest tracklet to determine the confidence.

Since the trajectory predicted by the Kalman filter is only based on the moving path of the target, we need to eliminate the background trajectory of the predicted trajectory by the residual classification network. In addition to the stability of the predicted target trajectory, it is also terminated when the predicted trajectory deviates too much from the target, and the erroneous prediction is gradually expanded.

B. Residual Classification Network

Since the candidate position of the detector and the prediction of the trajectory by the Kalman filter will result in too many candidate positions, we use the classifier to filter and remove the part of the noise in these candidate positions, and generate a score for the tracklet. The process will be done using the residual classification network.

We use the Resnet-50 convolutional layer in the classification network, which is a powerful deep network structure in object detection based on Faster RCNN object detection network [27]. However, we remove the RPN network structure during actual tracking and reduce the amount of computation of the network. After the entire network is routed by multiple layers of convolution, the feature map is subjected to ROI pooling, and after two layers of fully-connected for each feature map ROI block, the classification probability of each block is outputted. As shown in Figure 2, the classification network contains 50 residual convolution layers, and outputs the probability of the target category after ROI pooling and two fully-connected layers.

We use a deep network structure such as ResNet-50n to construct the dual neural network for classification and generate target features to enhance the effect. The tens of layers of convolution layers allow features to be subdivided layer by layer, and let the neural network produce a more accurate classification of the connotation of objects, which is of great help for us to find every pedestrian in a complex environment where there may be dozen kinds of objects. However, due to the use of deep networks, each layer will subdivide the features again, which also makes the higher layers of the convolution layer lose more details, forming a feature that is rich of semantic features but lacks appearance information. Therefore, in order to solve this problem, we applied the feature graph pyramid structure [34], so that high-level features are referred and low-level feature information is preserved. And we applied the concept of ROI pooling [27] to obtain the global feature map by multi-layer convolution in the neural network, and then cut out the corresponding area of each ROI in the feature maps, reduced the dimension to 7x7 and finally pass the features through the fully-connected layer to do classification. When training the residual classification network, the loss function is defined as (1):

$$L_{obj}(p_i, p_i^*) = -\log[p_i^* p_i + (1 - p_i^*)(1 + p_i)] \qquad (1)$$

where $\boldsymbol{p_i^*}$ represents the ground truth of the object $\boldsymbol{i}$, and $\boldsymbol{p_i}$ represents the probability of the category predicted by the object through the classification network. We use the form of cross entropy as the calculation formula of the loss function.

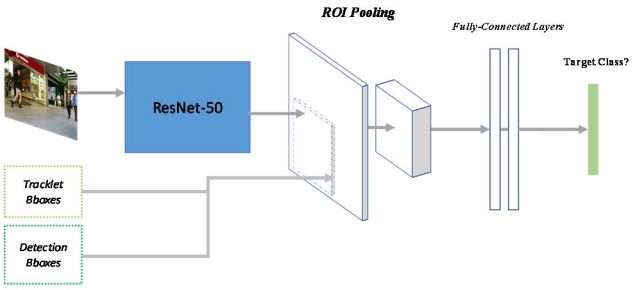

Figure 2. Architecture of the Residual Classification Network

C. Residual Feature Network

How to determine the similarity between candidate locations and trajectories is the key to data association. To produce the similarity of the appearance of the object, the current mainstream is to generate depth features by deep learning, which is superior to the hand-craft features generated by traditional methods, such as Histogram of Oriented Gradient (HOG) and color histogram maps. We calculate the feature distance of the candidate pairs by inputting the RGB images of targets and extracting the feature. Linking the most similar tracklets and distinguishing between two overlapping staggered candidates can reduce the loss of trajectory and maintain tracking stability in crowded scenes.

A feature network is trained so that the features of different targets have the largest distance, while the features of the same target have the smallest distance. By using the model of this appearance feature and recording the historical feature of the tracklets, it is possible to effectively distinguish between different targets. ResNet-50 is used as the basenet for the convolutional layer for deep features. It helps to convert the candidate ROI images filtered by the residual classification network into the residual feature network and convert them into 256-dimensional features. As shown in Figure 3, the original candidate region RGB image is inputted into the 50-layer convolution, and then subjected to two fully-connected layers to reduce the dimension to a unique feature vector. The entire residual feature network is trained using the triplet loss method.

But the deep residual network makes training difficult. The network is superimposed from layers, and the adjacent layers have high influence on each other. However, the deeper hidden layers of the network is, the more likely it is to cause a gradient diffusion problem. Therefore, according to [35], we introduce group normalization to standardize the values of the channel dimensions of the feature map, divide all channels into groups of equal divisions, and normalize the values in groups.

The residual feature network can generate unique features from the different targets, and the features must have high similarity under the same target but different images. Therefore, the triple loss function [37] is used to generate the target features to determine the similarity. We randomly select a sample from the training samples and use it as the core of this round of training to compare the calculated distance; Second, from the training samples, we randomly select another sample which is the same category as anchor, called it positive; Third, we select any sample of the different categories with anchor from the training samples, call it negative. Triplet loss has three requirements: (a) The shorter the feature distance between anchor and positive, the better, (b) The longer the distance between anchor and negative, the better, (c) The distance between the two groups, positive and negative groups, needs to have a minimum interval. The training objective function of residual feature network is given as follows.

$$L_{triplet} = \frac{1}{N} \sum_{(s_a, s_p, s_n)} \max(d_{ap} - d_{an} + g, 0) \qquad (2)$$

we want to minimize (2), where $\boldsymbol{N}$ is the number of samples used in the training round, $\boldsymbol{s_a}$ represents the features of any random sample extracted during training through the network embedding, $\boldsymbol{s_p}$ represents the feature of positive sample from the $\boldsymbol{s_a}$ category, and $\boldsymbol{s_n}$ represents the features of a sample from the negative categories other than $\boldsymbol{s_a}$. We can calculate the distance between features by Euclidean distance, $\boldsymbol{d_{ap}}$ represents the Euclidean distance of $\boldsymbol{s_a}$ and $\boldsymbol{s_p}$, and $\boldsymbol{d_{an}}$ represents the Euclidean distance of $\boldsymbol{s_a}$ and $\boldsymbol{s_n}$. The difference between the two groups must be greater than a threshold $\boldsymbol{g}$, it is set to one here. The purpose of our training is to make $\boldsymbol{d_{ap}}$ as small as possible and $\boldsymbol{d_{an}}$ as large as possible.

After training with the triplet loss, each candidate image is mapped to the feature dimension through the residual feature network, and we can use the distance of the features to judge its uniqueness and similarity. The overall structure is shown in Figure 3. Again ResNet-50 is used as the convolutional layer to reduce the dimension of two fully-connected layers into a feature vector. After that, the feature distance can be used to distinguish whether the input target is the same person. In order to enhance the versatility of network and prevent the network from over fitting, we add the dropout layer between the two fully-connected layers. In this way, the performance of residual feature network mapping image to feature dimensions is more general.

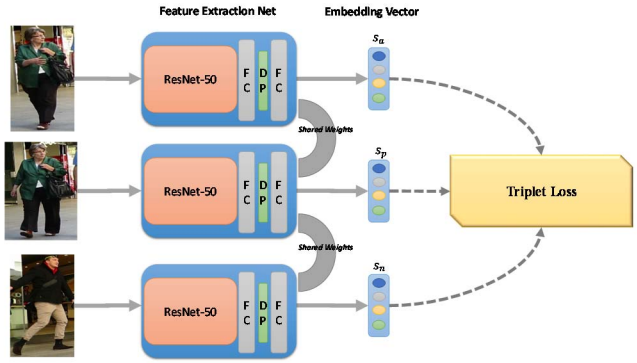

Figure 3. Training of Residual Feature Network

D. Four Layer Data Association

The residual classification network and the residual feature network is combined to associate the same target at different points in time, and each target has its own action, resulting in different tracklets. When dealing with the prediction of trajectory tracking, it may predict noise such as the background of the tree, thus causing the tracking target position to start deviating, and making these noise backgrounds affect the appearance features. In order to avoid this situation, we will correlate the alive tracklets, the missing tracklets and the newborn tracklets layer by layer according to the features of different candidates. In this way, we can reduce the vulnerability of the Kalman filter in long-term tracking.

With the position input from the detector, the Kalman filter is used to predict the trajectory of targets. This has a high degree of accuracy over a short period of time, but once the time interval is stretched, the interference that may be experienced during the period may cause the tracklet to deviate or even be interrupted. Therefore, a reference value is used for the possible position to assist the stability of the tracklet. The trajectory can be seen as a relationship in which the target positions of the respective time periods are connected in series by the time line, and is composed of short tracks that are interrupted one by one. We will re-associate the old and new tracks, and after retrieving the trajectory, we will reset the confidence of this tracklet, that is, use only the last uninterrupted and continuously tracklet to assist the Kalman filter. This representation of confidence is given by (3):

$$V_{conf} = \min(\tau_{up}, \alpha + \log(1 + \lambda \cdot L_{trk}) \cdot \mathbb{I}(L_{det} \geq 2)) \quad (3)$$

$\boldsymbol{V_{conf}}$ represents the confidence of the tracklet, $\boldsymbol{\alpha}$ represents the initial value of the confidence and $\boldsymbol{\tau_{up}}$ represents the upper bound of the maximum value of the confidence. $\boldsymbol{\lambda}$ is the growth coefficient of the confidence, and we set its value to 0.05 in practice. $\boldsymbol{L_{trk}}$ represents the lifetime of tracklet and $\boldsymbol{L_{det}}$ represents how long the tracjectory takes after obtaining the target position from the detector. $\mathbb{I}(\cdot)$ means that if the judgment in the bracket is true, then it returns 1, otherwise it returns 0. For the design of tracklet confidence, if the target exists in the short tracklet for a long time, we can infer that the tracklet is more likely to be a stable moving path, so it will give it a higher confidence and will not be interrupted to make up the instability of the Kalman filter for long-term trajectories.

We can integrate the input from the detector and the confidence of the target predicted by the Kalman filter to a score, and represent it by the probability value of the residual classification network. The score is given in (4).

$$s = p(\mathrm{y}|\mathrm{z},\mathrm{x}) \cdot [\mathbb{I}(\mathrm{x} \in B_{det}) + V_{conf} \cdot \mathbb{I}(\mathrm{x} \in B_{trk})] \quad (4)$$

$\boldsymbol{s}$ represents the stability score for each candidate bounding box, p represents the classification probability of the candidate, B_{det} represents the candidate from the detector, and B_{trk} represents the candidate from the Kalman filter prediction. After the score of all the candidates are obtained, we will do non-maximum suppression (NMS), and eliminate the overlapping candidate regions according to this score.

Next, a four-layer data association scheme is conducted based on the features of targets and candidates. In order to avoid the phenomenon that the trajectory appears intermittent, if the features are paired in a simple and rough way, a layered associated matching method is used. We found that in addition to the better results, it also reduces the number of ID switching. The four-layer data association scheme is shown in Figure 4. First, the Euclidean distance of features to match the candidate targets sent by the detector with the tracklets that is calculated using the following formula:

$$M_d = \|(\mathcal{F}_{\mathcal{T}} - \mathcal{F}_B\|_2 \quad (5)$$

$$\mathbb{P} = \mathrm{HA}(M_d) \quad (6)$$

where $\boldsymbol{\mathcal{F}_{\mathcal{T}}}$ represents the features of the tracklets and $\boldsymbol{\mathcal{F}_B}$ represents the features of the candidates. The Euclidean distance is calculated from all combinations and paired them by the Hungarian assignment method. $\mathbb{P}$ represents the pairs matched from the candidates with tracklets. The maximum distance of the feature pairs is set to a threshold $\boldsymbol{\tau_d}$. If the feature distance exceeds $\boldsymbol{\tau_d}$, it is ignored. If the tracklet in the tracking is not matched to any candidate, it is assumed that the tracklet is possibly missing. Second, the remaining candidates and the lost tracklets are matched by the Hungarian assignment again; Third, those candidates that are not paired and the candidates predicted by the Kalman filter are combined, and then are matched to the tracklets that are not successfully paired in the first step. Fourth, the newborn tracklets from the previous frame are matched to the remaining candidates of the third step. Finally the remaining candidates are assumed to be the new targets. After the four-layer data association scheme, we can match all candidates with the old tracklets and get the newborn tracklets.

Algorithm 1 The Proposed Four-layer Data Association Scheme.

/* $\mathcal{T}$ represents the tracklets */
/* S represents the stability scores of the tracklets */
/* B represents the bounding box locations */
for each $frame$ f_k in v **do**
  $\mathcal{T}_{trk} = \mathcal{T}_{alive} \cup \mathcal{T}_{lost}$
  Estimate S_{trk} by tracklet scoring function
  Set all $S_{det} = 1$
  $S_{cand} = S_{trk} \cup S_{det}$
  for each t in $\mathcal{T}_{trk}$ **do**
    Predict t's new bbox B_t by Kalman Filter
    $B_{trk} = B_{trk} \cup B_t$
  end for
  $B_{cand} = B_{det} \cup B_{trk}$
  Gecnerate P_{cand} by Classification Net
  $S_{cand} = S_{cand} \times P_{cand}$
  $B_{cand}, S_{cand} = NMS(B_{cand}, S_{cand}, \tau_{iou})$
  $B_{cand}, S_{cand} = Greater(B_{cand}, S_{cand}, \tau_s)$
  for each b in B_{cand} **do**
    /* C represents the cropped images */
    $C_b = Crop(b, f_k)$
    $C_{cand} = C_{cand} \cup C_b$
  end for
  Generate feature for all B_{cand} by passing C_{cand} through Feature Net
  /* Four Steps Data Association */
  $\mathcal{T}_{match1}, \mathcal{T}_{rest1}, B_{rest1} = First\_Match(\mathcal{T}_{alive}, B_{det})$
  $\mathcal{T}_{match2}, B_{rest2} = Second\_Match(\mathcal{T}_{lost}, B_{rest1})$
  $B_{third} = B_{rest2} \cup B_{trk}$
  $\mathcal{T}_{match3}, B_{rest3} = Third\_Match(\mathcal{T}_{rest1}, B_{third})$
  $\mathcal{T}_{match4}, B_{rest4} = Fourth\_Match(\mathcal{T}_{unconf}, B_{rest3})$
  $\mathcal{T}_{new} = Init\_Track(B_{rest4})$
  $\mathcal{T}_{alive} = \mathcal{T}_{match1} \cup \mathcal{T}_{match2} \cup \mathcal{T}_{match3} \cup \mathcal{T}_{match4} \cup \mathcal{T}_{new}$
end for

Figure 4. The four-layer data association scheme

IV. EXPERIMENTAL RESULTS

This section presents the design and results of the experiment. The experimental environment is shown in Table 1.

Table 1. Experimental environment

Experimental Environment
• Operating System: Ubuntu 16.04 64bit • CPU: Intel® Core™ i7-7700 Processor • Main Memory: 16.0 GB • GPU: NVIDIA TITAN Xp 12GB • Development Software: Tensorflow • Graphics API: OpenCV

In order to train the residual classification network, the Microsoft COCO 2017 Dataset [38] is used for training. The training set of COCO2017 provides more than 120,000 images and 80 categories of objects. During training, we used the ImageNet to pretrain Resnet-50 convolutional layers and then migrated to the classification network.

We used two re-identified datasets, DukeMTMC [39] and Market1501 [40] to train the residual feature network. DukeMTMC provides 36,411 images containing 1,404 people, while the Market1501 provides 32,668 images containing 1,501 people. Integrating these two datasets, we train the residual feature network with nearly 3,000 targets to extract the deep feature from each target.

In order to test the effectiveness of the proposed method, we used the MOT16 tracking database [41] to conduct the experiments. It is a test database widely used in multi-object tracking, and it is also the most popular evaluation benchmark. We used seven films from the testing set as inputs and experimented with Deformable Part Models (DPM) [42] provided in the database as the detector. The evaluation metric are as follows:

1. **MOTA** (Multiple Object Tracking Accuracy): the proportion of the correct predicted trajectory segment in ground truth trajectories.
2. **IDF1** (Identification F-Score): the proportion of the correct predicted ID in ground truth IDs.
3. **FP** (False Positive number): the number of wrong objects as the target.
4. **IDsw** (Identification Switch number): the number of conversions of the target ID in all trajectories.
5. **Hz**: the number of frames that can be processed in one second.

To effectively tracking multi-object, the proposed method uses the residual classification network and the residual feature network, and applies a four-layer data association scheme. And in order to prove the validity of the proposed different structure, we will show the effect of the tracking method improvement after each structure is added. As listed in Table 2, ***base*** represents the basic tracking method consisting only of the detection combined with the Kalman filter, ***C*** represents the residual classification network, and ***F*** represents the residual feature network, ***A*** represents the association scheme, which is the structure consisting of adding four-layer data association and the tracklet confidence.

The proposed method is also compared with other current multi-object tracking methods. These testing are based on MOT16 testing dataset for experiments, and the results are shown in Table 3. Compared with several competing methods, the proposed Layered Residual-Residual tracking method leads most of the current multi-object tracking methods. Due to the use of the classification network to filter and correct the predicted position of the Kalman filter, the false positive number of our tracker has a significant reduction and reduce the influence caused by background noise or target frame deviation. The experimental result shows that combining such a double network with layered association structure has a significant effect on accuracy, which is only a little bit lower than FWT, but our method is 7 times faster than FWT and most of existing methods. Moreover, the false positive reduction of our tracker is better than all of them.

Table 2. Performance by adding each structure

Method	MOTA(↑)	IDF1(↑)	FP(↓)	FN(↓)	IDsw(↓)
Base	35.6%	39.5%	8,991	114,875	1021
Base + C	40.1%	42.2%	**2,746**	106,040	1171
Base + C + F	43.6%	44.6%	2,837	101,869	**809**
Base + F + A	40.3%	40.1%	7,503	100,246	1099
Base + C + F + A	**46.8%**	**47.7%**	2,855	**93,334**	780

Table 3. Comparison of the proposed methods and other methods in MOT2016

Mode	Method	MOTA(↑)	IDF1(↑)	FP(↓)	FN(↓)	IDsw(↓)	Hz(↑)
Offline	LINF1 [43]	41.0%	45.7%	7,896	99,224	430	4.2
	MHT_DAM [44]	45.8%	46.1%	6,412	91,758	590	0.8
	JMC [45]	46.3%	46.3%	6,373	90,914	657	0.8
	NOMT [46]	46.4%	53.3%	9,753	87,565	**359**	0.9
Online	EAMTT [47]	38.8%	42.4%	8,114	102,452	965	**11.8**
	STAM [48]	46.0%	50.0%	6,895	91,117	473	0.2
	DMMOT [49]	46.1%	**54.8%**	7,909	89,874	532	0.3
	RAR16pub [50]	45.9%	48.8%	6,871	91,173	648	0.9
	FWT [51]	**47.8%**	44.3%	8,886	**85,487**	852	0.6
	Our method	46.8%	47.7%	**2,855**	93,334	780	8.5

V. CONCLUSIONS

We proposed the residual-residual network tracking structure and the layered data association scheme. The residual classification network is used to predict and filter the target location, and the residual feature network generating target feature that is used to judge the similarity, the four-layer data association scheme is used to match the tracklets layer by layer. Experimental results prove that using such a double network combined with layered association structure is effective.

REFERENCES

[1] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid and K. Schindler, "Online Multi-object Tracking Using Recurrent Neural Networks," in *AAAI*, 2016.

[2] A. Andriyenko and K. Schindler, "Multi-object Tracking by Continuous Energy Minimization," in *CVPR*, 2011.

[3] A. Butt and R. Collins, "Multi-object Tracking by Lagrangian Relaxation to Min-cost Network Flow," in *CVPR*, 2013.

[4] A. Bewley, Z. Ge, L. Ott, F. Ramos and B. Upcroft, "Simple Online and Realtime Tracking," in *ICIP*, 2016.

[5] B. Han, J. Sim and H. Adam. Branchout, "Regularization for Online Ensemble Tracking with Convolutional Neural Networks," in *CVPR*, 2017.

[6] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. Luk Chan and G. Wang, "Joint Learning of Convolutional Neural Networks and Temporally Constrained Metrics for Tracklet Association," in *CVPRW*, 2016.

[7] B. Wang, G. Wang, K. Luk Chan and L. Wang, "Tracklet Association with Online Target-specific Metric Learning," in *CVPR*, 2014.

[8] B. Wu and R. Nevatia, "Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet Based Part Detectors," in *IJCV*, 2007.

[9] B. Yang and R. Nevatia, "An Online Learned CRF Model for Multi-object Tracking," in *CVPR*, 2012.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going Deeper with Convolutions," in *CVPR*, 2015.

[11] C. Kim, F. Li, A. Ciptadiand J. M. Rehg, "Multiple Hypothesis Tracking Revisited," in *ICCV*, 2015.

[12] C.-H. Kuo and R. Nevatia, "How Does Person Identity

Recognition Help Multi-person Tracking," in *CVPR*, 2011.
[13] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally and K. Keutzer, "SqueezeNet: AlexNet-level Accuracy with 50x Fewer Parameters and <0.5MB Model Size," *arXiv preprint arXiv:1602.07360*, 2016.
[14] H. Noh, S. Hong and B. Han, "Learning Deconvolution Network for Semantic Segmentation," in *ICCV*, 2015.
[15] H. Nam and B. Han, "Learning Multi-Domain Convolutional Neural Networks for Visual Tracking," in *CVPR*, 2016.
[16] J. Long, E. Shelhamer and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," in *CVPR*, 2015.
[17] J. Hu, L. Shen and G. Sun, "Squeeze-and-Excitation Networks," *arXiv preprint arXiv:1709.01507*, 2017.
[18] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-time Object Detection," in *CVPR*, 2016.
[19] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *CVPR*, 2017.
[20] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *CVPR*, 2016.
[21] K. He and J. Sun, "Convolutional Neural Networks at Constrained Time Cost," in *CVPR*, 2015.
[22] L. Leal-Taix´e, C. Canton-Ferrer, and K. Schindler, "Learning by Tracking: Siamese CNN for Robust Target Association," in *CVPRW*, 2016.
[23] L. Leal-Taix´e, M. Fenzi, A. Kuznetsova, B. Rosenhahn and S. Savarese, "Learning an Image-based Motion Context for Multiple People Tracking," in *CVPR*, 2014.
[24] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier and L. Van Gool, "Online Multiperson Tracking-by-Detection from a Single, Uncalibrated Camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 9, pp. 1820–1833, 2011.
[25] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in *CVPR*, 2014.
[26] R. Sanchez-Matilla, F. Poiesi and A. Cavallaro, "Online Multi-object Tracking with Strong and Weak Detections," in *ECCV*, 2016.
[27] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks," in *NIPS*, 2015.
[28] S. Kim, S. Kwak, J. Feyereisl and B. Han, "Online Multi-target Tracking by Large Margin Structured Learning," in *ACCV*, 2012.
[29] S. Wang and C. Fowlkes, "Learning Optimal Parameters for Multi-object Tracking," in *BMVC*, 2015.
[30] S.-H. Bae and K.-J. Yoon, "Robust Online Multi-object Tracking Based on Tracklet Confidence and Online Discriminative Appearance Learning," in *CVPR*, 2014.
[31] W. Liu, D. Anguelov, D. Erhan, C. Szegedy and S. Reed, "SSD: Single Shot Multibox Detector," in *ECCV*, 2016.
[32] X. Song, J. Cui, H. Zha and H. Zhao, "Vision-based Multiple Interacting Targets Tracking via Online Supervised Learning," in *ECCV*, 2008.
[33] Y. Xiang, A. Alahi and S. Savarese, "Learning to Track: Online Multi-object Tracking by Decision Making," in *ICCV*, 2015.
[34] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," in *CVPR*, 2017.
[35] Y. Wu and K. He, "Group Normalization," in *ECCV*, 2018.
[36] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv preprint arXiv: 1502.03167*, 2015.
[37] A. Hermans, L. Beyer and B. Leibe, "In Defense of the Triplet Loss for Person Re-Identification," *arXiv preprint arXiv:1703.07737*, 2017.
[38] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick and P. Dollár, "Microsoft COCO: Common Objects in Context," *arXiv preprint arXiv:1405.0312*, 2015.
[39] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara and C. Tomasi, "Performance Measures and a Data Set for Multi-object, Multi-Camera Tracking," in *ECCVW*, 2016.
[40] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang and Q. Tian, "Scalable Person Re-identification: A Benchmark," in *ICCV*, 2015.
[41] A. Milan, L. Leal-Taixe, I. Reid, S. Roth and K. Schindler, "MOT16: A Benchmark for Multi-Object Tracking," *arXiv preprint arXiv:1603.00831* , 2016.
[42] R. Girshick, F. Iandola, T. Darrell and J. Malik, "Deformable Part Models are Convolutional Neural Networks," *arXiv preprint arXiv:1409.5403*, 2014.
[43] L. Fagot-Bouquet, R. Audigier, Y. Dhome and F. Lerasle, "Improving Multi-Frame Data Association with Sparse Representations for Robust Near-Online Multi-Object Tracking," in *ECCV*, 2016.
[44] C. Kim, F. Li, A. Ciptadi and J. Rehg, "Multiple Hypothesis Tracking Revisited," in *ICCV*, 2015.
[45] S. Tang, B. Andres, M. Andriluka and B. Schiele, "Multi-Person Tracking by Multicuts and Deep Matching," in *BMTT*, 2016.
[46] W. Choi, "Near-Online Multi-object Tracking with Aggregated Local Flow Descriptor," in *ICCV*, 2015.
[47] R. Sanchez-Matilla, F. Poiesi and A. Cavallaro, "Multi-object tracking with strong and weak detections," in *BMTT*, 2016.
[48] Q. Chu, W. Ouyang, H. Li, X. Wang, B. Liu and N. Yu, "Online Multi-Object Tracking Using CNN-based Single Object Tracker with Spatial-Temporal Attention Mechanism," in *ICCV*, 2017.
[49] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang and M. Yang, "Online Multi-Object Tracking with Dual Matching Attention Networks," in *ECCV*, 2018.
[50] K. Fang, Y. Xiang, X. Li and S. Savarese, "Recurrent Autoregressive Networks for Online Multi-Object Tracking," in *WACV*, 2018.
[51] R. Henschel, L. Leal-Taixé, D. Cremers and B. Rosenhahn, "Fusion of Head and Full-Body Detectors for Multi-Object Tracking," in *CVPRW*, 2018.