FGAGT: Flow-Guided Adaptive Graph Tracking

Chaobing Shan\(^1,2\), Chunbo Wei\(^2\), Bing Deng\(^2\), Jianqiang Huang\(^2\), Xian-Sheng Hua\(^2\)
Xiaoliang Cheng\(^1\), Kewei Liang\(^1,\)*

\(^1\)School of Mathematical Sciences, Zhejiang University \quad \(^2\)DAMO Academy, Alibaba Group

chaobing_s@zju.edu.cn, chunbo.wcb@alibaba-inc.com, dengbing.db@alibaba-inc.com
jianqiang.hjq@alibaba-inc.com, huaxiansheng@gmail.com, xiaoliangcheng@zju.edu.cn, matlkw@zju.edu.cn

Abstract

Most previous tracking methods usually use the optical flow method to estimate the position of the historical object in the current frame and then use the linear combination of feature similarity and IOU (Intersection over Union) to perform association matching near the position. However, the features used in these methods are not aligned, i.e., the features of the historical objects are extracted from the historical feature maps, not from the current frame, even the same object may undergo posture, angle, etc. changes during the movement, and even light intensity changes. In addition, most methods only use the appearance information when extracting the feature vector, not the position relationship, nor the feature information of the historical object, so the information is not fully utilized. In order to solve the above problems, we proposed the FGAGT tracker, which uses the optical flow method to predict the center position of the historical object in the current frame and extract the feature vector, so that the feature of the historical object can be aligned with the feature of the object in the current frame. Then these features are input into the graph neural network, and the global Spatio-temporal position and appearance information are integrated to update the feature vectors of all objects. In the training phase, we propose the Balanced MSE LOSS to balance the sample distribution for data association. Experiments show that our method reaches the level of state-of-the-art, where the MOTA index exceeds FairMOT by 2.5 points, and CenterTrack by 8.4 points on the MOT17 dataset, exceeds FairMOT by 1.6 points on the MOT16 dataset. Code will be available.

1. Introduction

Object tracking has always been a very important research direction in computer vision, and it has great applications in autonomous driving, video object behavior prediction, traffic management, and accident prevention\cite{18,17}. These objects can be arbitrary, such as pedestrians, vehicles, athletes, various animals, and so on. Compared with single-object tracking, multi-object tracking is more complicated. It needs to match multiple objects that have been detected in the current frame with the objects on the historical trajectories. The same object is given the same id (identity), then a new trajectory is formed. The new objects that never appear before need to be given new ids and the objects that disappear need to be removed from the trajectories and cannot be matched.

As we all know, many videos cover different scenes, such as clarity from clear to blurry, light intensity from day to night, viewing angle from high to low, and camera lens from static to moving. Many objects are self-occlusion, mutual occlusion in the video. Besides, objects are reflected in the mirror or window and only visible at the edge of the camera. And even the appearance of objects is very similar, objects are too small, deformed, and very dense, and so on, which brings great challenges to multi-object tracking\cite{4}.

Previous methods such as \cite{21,22,23,24,25,26,27} used the Kalman filter\cite{40} to predict the position of the historical objects in the current frame, and used the Hungarian algorithm\cite{20} for matching, or \cite{1,28,29,30,31,32,34,33} use CNN or variational methods to predict the optical flow of the historical objects from previous frames to the current frame, and use a simple greedy algorithm to complete the matching. \cite{2,35,36} add a re-id branch to extract finer feature vectors and use the linear combination of feature similarity and IOU to form a similarity matrix, then use the Hungarian algorithm to complete the matching. Although these methods have made great progress in the field of multi-object tracking, they still have the following shortcomings:

1. The features used in these methods are not aligned, i.e., the features of the historical objects are extracted from the historical feature maps, not from the current frame. This is a problem, even the same object may undergo posture, angle, etc. changes during the move-
ment, and even light intensity changes, so the features of the same object extracted from different frames may be very dissimilar, then the tracking accuracy will be reduced during association matching.

2. Most methods only use the appearance information when extracting the feature vector, not the position relationship, nor the feature information of the historical object, so the information is not fully utilized.

3. One object from the previous frame can only match one object in the current (continuous positive examples), and the rest does not match it (continuous negative examples). What’s more, there are few new objects and disappeared objects after a long time. Therefore, the number of these samples is very uneven.

Optical flow estimation can be divided into sparse optical flow estimation and dense optical flow estimation, here we should use the sparse optical flow method, because we only need to estimate the motion of the object’s center point. In some cases, the speed of the object is very large, so we must choose the optical flow method suitable for large motion estimation. The pyramid LK algorithm is a good choice. As shown in the figure[1], even if the object has a large motion between the previous and current frames, using the pyramid LK[7] algorithm can still align the historical bounding box (bbox) with the current bbox well.

The graph neural network[15] is very suitable for keeping the global spatiotemporal position, appearance information, and historical information of each object, so it can make full use of this information to update the feature vectors of all objects. Here we have slightly improved the graph network to allow the graph network to learn adaptive weights to fuse different information, so we call it adaptive graph neural network.

In this article, we propose the FGAGT tracker. At first, we use the pyramid LK algorithm to calculate the center position of the historical objects in the current frame (see figure[1]). The current frame forms to feature maps by CNN downsampling, then use ROI Pooling[19] and fully connected layers to extract the initial appearance feature vectors of the historical objects and the newly detected objects, input them into the adaptive graph neural network, the historical objects and the newly detected objects are treated as bipartite graphs (see figure[2]), and the initial appearance feature distances and IOU are regarded as edges’ weights, which is also a part of the input of the graph neural network, combined with global spatiotemporal position and appearance information, to update the feature vectors. In the graph neural network, each dimension of the aggregated features is multiplied by an adaptive weight, which is used to learn which part of the object is more identifiable to solve the occlusion problem and the re-identification problem.

The final output of the graph neural network is a similarity matrix. In the training phase, we propose the Balanced MSE Loss. Specifically, it is to multiply the coefficients that are inversely related to the sample number before the continuous positive, continuous negative, new, and disappearing objects’ loss functions, respectively. It’s beneficial for solving the problem of unbalanced sample number distribution (see figure[5]), so that the adaptive graph neural network can learn the feature vectors we want.

Specifically, the contributions of this article are:

1. Feature alignment: We predict the motion of historical objects, and extract their features on the current frame, so that the feature of the historical object can be aligned with the feature of the object in the current frame, that is, the feature of the same object is as similar as possible.

2. AGNN: We use adaptive graph neural network to integrate global temporal and spatial appearance position information which can update features and re-identify occluded objects.

3. BMSE: We propose Balanced MSE Loss to balance the distribution of various samples.

4. Best: Both the Public and Private datasets on the MOT Challenge are at the level of state-of-the-art.

The following contents in this article are organized follows: Section 2 introduces the related work of tracking, Section 3 introduces our FGAGT specific methods, Section 4 is some ablation experiments, and Section 5 is the summary.

2. Related Work

Most people’s work is tracking based on detection. This article divides these work into two categories according to whether feature matching is used:

**Featureless matching:** sort[21] first uses Faster-Rcnn[19] to detect the objects’ location and categories, and then uses the Kalman filter[40] to predict the historical object’s position on the current frame, next, calculate the IOU and use the Hungarian algorithm to match the historical trajectories with the newly detected objects. CenterTrack[11] inherits the network structure of CenterNet[38]. CenterNet is an anchor-free model, input the original image of 3 channels, downsample by 4 times (higher resolution), and use key point estimation to find The center points of the bounding boxes (the first branch of the output, each channel represents a category), and get the bbox size (the second branch of the output) at the same time, and the third output branch is the offset of the center point. CenterTrack has four more channels on the input: the 3-channel RGB image of the previous frame and the heatmap after the maximum pooling on
Figure 1. Benefits of using PyraLK. All three pictures contain the same object in the current frame \( I_t \). The bbox in (a) is from the previous frame \( I_{t-1} \). (b) uses the pyramid LK algorithm to predict the motion of the object, so the object’s center position in the current frame can be found (bbox’s width and height remain unchanged), and (c) is the ground truth bbox. We can see that (b) and (c) are almost aligned.

Figure 2. Explanation of our bipartite graph model. Each node presents an object or its feature vector. The left part is to aggregate the feature vector of each newly detected object with the feature vectors of all historical objects to complete the feature update, and the right is the opposite of the left.

Figure 3. Explanation of unbalanced sample number distribution. Each row presents a detected object in the current frame, each column presents a historical object in the previous frame. The elements in the red rectangle column are all zero, which means the historical object does not match any newly detected object, so it is a disappeared object. Elements in the green rectangle column are all zero, which means the newly detected object does not match any historical object, so it is a newly appeared object which will be added a new id. Apart from the above samples, the element which equals 1 means they are the same object, so they are continuous positive samples. An element that equals 0 means they are different objects, so they are continuous negative samples.

Feature matching: featureless matching actually looks at the positional relationship between the historical objects and the newly detected objects. In addition to the important factor of position, the appearance of the target is also a very important factor, such as black clothes, white clothes, fat, thin, front, back, and so on are all important features that distinguish different objects. RetinaTrack\cite{36} corrects the shortcoming that the structure of RetinaNet\cite{39} is not suitable for capturing the features of each instance. The more specific instance features corresponding to the anchor shape are captured before the classification and border regression, and then add a feature embedding branch, which is used for calculating feature distance then track. FairMOT\cite{2} proposed the reasons why the features captured by the anchor-based model are not good: (1) the anchor is not aligned with the object, resulting in capturing bad features; (2) the feature is downsampled too many times, unable to extract features that are beneficial to track; (3) High-dimensional features have the risk of overfitting. Therefore, FairMOT proposes an anchor-free model, which outputs the low-dimensional Re-id feature vectors of the objects while outputting the detection results, which solves the problem of feature misalignment, improves the speed, and also helps reduce the risk of overfitting. However, the Re-id output of FairMOT is a one-hot vector. When the actual number of objects is much more than the output dimension, it will not be able to track. For the first time, Wang\cite{37} et al. proposed a framework for joint detection and tracking using GNN. The newly detected objects and historical trajectories are divided into bipartite graphs. First, the one-stage method is used to extract appearance features and output detection results. The trajectory uses LSTM\cite{49} to predict the position in the current frame. Then concat the position and appearance feature vectors, and then input them into the graph neural network to update the feature vectors, and finally use the fully connected layers output the similarity matrix for matching. There is a problem here, in the graph neural network, the weights of the edges between the newly detected objects and the historical objects are the same. This may result in the failure to learn good
features or the final feature vectors are averaged, that is, all
the feature vectors of the objects are almost the same. In
addition to tracking based on detection, there is also track-
ing to facilitate detection. Zhang[35] et al. proposed the
Tracklet-Conditioned formula. Based on the detection re-
sults of Faster-Rcnn and the known historical trajectories,
the bayesian formula is fully utilized to update the probabil-
ity of the detection category. One more feature embedding
branch is added to Faster-Rcnn architecture to calculate the
cosine similarity between objects’ feature vectors, and then
associating. But this method also has drawbacks, because the
Tracklet-Conditioned formula may only improve the ac-
curacy of the classification, but not the accuracy of the
bounding box coordinates.

We believe that to match the objects, the location infor-
mation and appearance characteristics must be fully used in
order to have a good tracking result. Although some of the
above work predicts the position of the historical objects in
the current frame, it does not use the predicted position to
update the historical objects’ feature vectors. And when an
object is occluded, a small part of the occluded target will
obviously not be occluded, we need give that small part a
large weight to get a more refined Re-id feature vector. Be-
sides, an object reappears after being completely occluded,
it is necessary to aggregate the global spatiotemporal infor-
mation to update the Re-id feature vector. In addition, most
samples do not match each other. New and disappearing
objects account for a small number. Therefore, we need to
give different weights to these loss function, so that the
neural network will not learn the incorrect distribution due
to too many negative samples.

3. The FGAGT Architecture

The model framework is shown in Figure 4. The cur-
current frame \(I_t\) is transformed into feature maps by backone,
and then the detected \(M\) bboxes are used to extract region
features, and then transformed into feature vectors through
ROI Pooling and fully connected layers. At the same time,
the bbox of the historical objects of the \(I_{t-1}\) frame is pre-
spected to be in the current frame by the pyramid-LK algo-
algorithm, plus the bbox of the previous \(T\) frame (except for the
\(I_{t-1}\) frame), a total of \(N\) historical bboxes are also trans-
formed into feature vectors through ROI Pooling and fully
connected layers. Therefore, the input of the graph neural
network has two parts: the appearance feature vectors and
bbox of the \(M\) objects in the current frame, and the appear-
ance feature vectors and bbox of the historical objects in
the previous \(T\) frame. It should be noted that the appear-
ance feature vectors are extracted from the feature maps of
the current frame, not that the historical objects is extracted
from the feature maps of the historical frames.

Finally output the similarity matrix \(S \in \mathbb{R}^{M \times N}\)

3.1. Adapted Graph Neural Networks’ Module

In 2009, Dr. Franco defined the theoretical basis of
graph neural network in his paper [15]. The graph neu-
ral network used in this article is based on this paper.
The earliest GNN mainly solved graph theory problems
in a strict sense such as molecular structure classification.
But in fact, in European spaces (such as images) or se-
quences (such as text), many common scenes can also be
converted into graphs, and then graph neural network tech-
ology can be used to model them. After 2009, there
have been some related studies on the graph neural net-
work, but there is not much disturbance. Until 2013,
on the basis of Graph Signal Processing, Bruna first pro-
gosed the graph convolution neural networks in the liter-
ature [16] based on the Spectral-domain and the spatial
domain. Since then, Graph Convolutional Networks and
its variants [41,42,43,44,45,46,47,48], Graph Recur-
rent Networks[50,51,52,53,54,55,56], Graph Attention
Networks[56,55,57], Graph Residual Networks[59,60,61]
62,63,64 have all been proposed and demonstrated strong
abilities.

We treat newly detected objects and historical objects as
a bipartite graph(see figure 2), i.e., each newly detected ob-
ject has a connection with all historical objects but there
is no connection between any two newly detected object
or two historical objects. The learning goal of GNN is to
obtain the perception of the graph of each node’s (percep-
tion is relative to compression, using existing information
features to reconstruct more complete information features)
hiding State \(h_{c}(\text{state embedding})\). For each node, its hid-
den state contains information from the neighboring node.
We use the update formula:

\[
\begin{align*}
\begin{aligned}
&h_{d,c}^{t+1} = f\left(h_{d,c}^{t}, \{h_{d,c}^{t+1}_j e_{d,c}^{i,j} \}_{j=1}^{N}\right), \; i = 1,2,\cdots, M \\
&h_{t,c}^{t+1} = f\left(h_{t,c}^{t}, \{h_{d,c}^{t+1}_j e_{t,c}^{i,j} \}_{j=1}^{N}\right), \; j = 1,2,\cdots, N.
\end{aligned}
\end{align*}
\]  

(1)

Here \(f\) is the updating function of hidden state, we use neu-
nral network instead, \(h_{d,c}^{t+1}\) is the hidden feature vector of the
i-th newly detected object in the c-th layer, and \(h_{t,c}^{t+1}\) is the
hidden feature vector of the j-th historical object in the c-th
layer. When \(c = 0\), \(h_{d,0} = f_{d}, h_{t,0} = f_{t}\), \(e_{d,c}^{i,j}\) represents
the weight of the edge between the i-th newly detected ob-
ject and the j-th historical object in the c-layer. In this article
we use an one-layer GNN, and adds an adaptive part.

When using AGNN(Adaptive Graph Neural Network) to
update the feature vectors of all objects, instead of simply
setting some unknown parameters to update the features,
but using the existing position and feature prior information
as the weight of the edge \(E = \{e_{ij}^{c}\}\) to aggregate object fea-
ture vectors then updating them. The specific aggregation
steps are as follows:
1. Calculate the initial feature similarity matrix

\[ s_{i,j} = \frac{1}{\left\| f_d^i - f_t^j \right\|^2 + 1 \times 10^{-16}} \]

\[ s_{i,j} = \frac{s_{i,j}}{\sqrt{s_{1,1}^2 + s_{1,2}^2 + \cdots + s_{i,j}^2 + \cdots + s_{i,N}^2}} \]

\[ S_h = [s_{i,j}]_{M \times N}, \quad i = 1, 2, \cdots M, \quad j = 1, 2, \cdots N \]  

(2)

2. Calculate the bbox IOU(Intersection over Union ) and form a prior similarity matrix with the result of step 1

\[ E = w \times IOU + (1 - w) \times S_h \]  

(3)

The parameter \( w \) measures the relative importance between location information and appearance feature information, and the initial value is set to 0.5.

3. Aggregation features:

\[ F_{t}^{agg} = EF_t = E \begin{bmatrix} f_t^1 \\ f_t^2 \\ \vdots \\ f_t^N \end{bmatrix} \]  

(4)

Then update the features:

\[ H_d = \sigma (F_dW_1 + \text{Sigmoid}(F_dW_a) \odot F_t^{agg}W_2) \]  

(5)

Similarly,

\[ H_t = \sigma (F_tW_1 + \text{Sigmoid}(F_tW_a) \odot F_d^{agg}W_2) \]  

(6)

Where \( \odot \) is the dot product. The values in different dimensions of the object feature vector represent the characteristics of different parts of this object being captured, and different parts may be the key to distinguishing the objects. Therefore, when aggregating features, we need to weight the values in different dimensions, how to weight needs to be determined according to the input feature vector, so we call \( W_a \) as the adaptive parameter, and \( \text{Sigmoid}(F_dW_a) \) as the adaptive weight.

The existing graph network tracking algorithm needs additional fully connected layers to reduce the dimensionality after the graph network updates the features and then calculate the Euclidean distance to measure the similarity between the features. We only need a simple one-layer graph network to unite the output feature vectors, and only need simple matrix multiplication to get the similarity matrix:

\[ h_d^i = \frac{h_d^i}{\|h_d^i\|_2} \]

\[ h_t^i = \frac{h_t^i}{\|h_t^i\|_2} \]

\[ S_{out} = H_dH_t^T \]  

(7)
The output of the same target is 1, and the output of different targets is 0. The purpose is to make the feature vectors of the same target close to coincide, and the feature vectors of different targets close to vertical. When the feature vectors are not normalized and elements are all greater than zero, this is equivalent to the closer the Euclidean distance of features between the same target and the longer the Euclidean distance between different targets, the better.

### 3.2. Blanced MSE Loss

After outputting the similarity matrix, we expect the table corresponding to the same target to be 1, and different targets to be 0. When calculating loss, we use MSE Loss. However, most targets appear continuously, with only a small number of new and disappearing targets. In addition, continuous targets have at most one positive example in the previous frame (label=1), the others are all negative examples (label=0), so the number of samples is extremely unbalanced. Therefore, we multiply a coefficient before the loss function corresponding to the new and disappearing targets and the targets of continuous positive and negative samples to balance the number of continuous positive and negative targets, new and disappearing targets, and record them as **Blanced MSE Loss**.

\[
\mathcal{L} = \alpha E_{c0} + \beta E_{c1} + \gamma E_{new} + \delta E_{dis} + \varepsilon E_w = \sum_{i=1}^{M} \sum_{j=1}^{N} \left[ \alpha \left( \hat{S}_{i,j} - S_{i,j} \right)^2 \cdot \mathbb{I}_{\text{continue}} : \mathbb{I}_{S_{i,j}}=0 + \beta \left( \hat{S}_{i,j} - S_{i,j} \right)^2 \cdot \mathbb{I}_{\text{continue}} : \mathbb{I}_{S_{i,j}}=1 \right]^2
\]

Where \( \mathbb{I}_c(S_{i,j}) = \begin{cases} 1, & \text{if } S_{i,j} \text{ is the c target} \\ 0, & \text{if } S_{i,j} \text{ isn’t the c target} \end{cases} \), \( \alpha, \beta, \gamma, \delta, \varepsilon \) are the hyperparameters.

### 3.3. Inference

In the test stage, after the similarity matrix \( S_{\text{out}} \) is obtained through the adaptive graph neural network, we then add a matrix of \( M \times M \), all elements are equal to \( \text{margin} = \pi \), added to the right side of \( S_{\text{out}} \) to form an augmented matrix, i.e., \( S_{\text{out}} = [S_{\text{out}}, \pi \times \mathbb{1}_{M \times M}] \), and then use the Hungarian algorithm to get the best match.

**Association match and new object appearing.** Assuming that the Hungarian Algorithm output is: \((i, j)\), \( i \) is the number of rows, and \( j \) is the number of columns. If \( j < N \), then \( i \) and trajectory \( j \) will be matched, and the id of \( j \) is assigned to the object \( i \). Otherwise, \( i \) is a new target, then the id of the object \( i \) is equal to \( \max \{id\} + 1 \). As shown in figure 5, take \( M = 8, N = 10 \), and take margin \( \pi = 0.2 \). According to the Hungarian algorithm, we can get that the third and eighth objects in the solution are new objects, other objects are matched, the 3rd and 8th objects add 1 and 2 to the current id number, respectively.

**Object disappearing.** When we are testing, we set \( k = 10 \), which means that the information of the object in the previous 10 frames is retained and matched with the object in the current frame. If an object in one frame does not match all objects in the next ten frames, the object is considered to disappear.

Figure 5. An example of our method. After Hungarian algorithm, we can get the 3rd and 8th objects are new objects, so their id are \( \max \{id\} + 1 \) and \( \max \{id\} + 2 \), respectively.
camera, even the appearance of the target is very similar, the target is too small, deformed, very dense, etc., which almost include real-life All scenes in. The training and test set of MOT16 Public officially provides the detection results of the DPM detector, and the training set also provides ground truth. The Public list requires us to track with the official detection results, and submit the results of the test set to evaluate the tracking method. The private list requires us to detect by ourselves and then track, and the final evaluation is also the quality of the tracking method. MOT17 Public officially provides the detection results of three detectors, namely DPM, Faster-Rcnn, and SDP. We need to track these three detection results separately and combine these three results to evaluate the quality of the tracker. So we have evaluated our method on Public and Private of MOT17 and MOT16, and both have reached the most advanced level. Among them, on the Private dataset, we use the detection results of Fair for tracking. In the ablation experiment, because it takes 3 days to submit to the MOT official website to be evaluated once, and an account can only be evaluated 4 times, we divided the MOT17 training set into two, half of which was used for training, and the other half as the validation set, using the DevKit provided by the official website to evaluate.

**Metrics:** We use the official evaluation indexes, the main index is multi-object tracking accuracy [65]:

$$MOTA = 1 - \frac{\sum t (FP_t + FN_t + IDSW_t)}{\sum t GT_t}$$

Among them, $GT_t$, $FP_t$, $FN_t$, $IDSW_t$ are the number of ground truth bbox, false positives, false negatives, and identity switches in frame $t$ respectively. In addition, the evaluation indexes include IDF1 Score [65]: The ratio of correctly identified detections over the average number of ground-truth and computed detections. Mostly tracked targets(MT): The ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span. Mostly lost targets(ML): The ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life span.

### 4.2. Implementation Details

We use VGG16, ResNet34, ResNet50, ResNet101 and whether these networks are added with FPN networks as backbone for ablation experiments. These networks are pre-trained on the coco data set, and after 30 epochs of fine-tune on the MOT data set. These parameters will not participate in the update afterwards. Of the 7 training videos, 6 of the videos’ pixel size is $1920 \times 1080$, we will resize each frame to $1333 \times 750$, and there is another video whose pixel size is $640 \times 480$, we will resize each frame to $800 \times 450$, so is the test set. In the pyramid LK algorithm, the neighborhood window we choose is $120 \times 120$, the number of iterations is 10, and the convergence error is 0.01. Hyperparameters are set to $\alpha = 25$, $\beta = 1$, $\gamma = 50$, $\delta = 50$, $\epsilon = 0.01$ according to the number of different samples in the MOT data set, the initial learning rate is $lr = 0.05$, and use cosine annealing to update the learning rate, the learning rate is updated every 30 epochs, and the final learning rate is $lr = 2.5 \times 10^{-7}$.

### 4.3. Ablation Study

We first performed an ablation experiment on the selection of backbone without using any trick, that is, associating match using similarity matrix composed of initial features and IOU of front and back frame bboxes. The results are shown in table [1] without any trick. Different network structures have a certain impact on the accuracy of tracking. Simple network structures such as VGG16 and ResNet34 can have a very fast tracking speed, but the effect is not as good as deep networks such as ResNet101-FPN. Because ResNet101-FPN is not used for detection, it simply extracts features, so it can achieve real-time tracking. Therefore, the backbone chosen for the Public and Private data sets in this article is Resnet101-fpn.

The backbone used in table 2 is Resnet101-fpn. We do ablation experiments on whether to use feature alignment, AGNN, and Balanced MSE LOSS. It can be seen from table 2 that only using feature alignment, the MOTA index is 4.2 points higher than not using any trick, and 1.3 points higher than using AGNN only, and 2.4 points higher than using BMSE only. On the basis of using feature alignment, coupled with the AGNN updating feature, it is 2.7 points higher. Finally, with the addition of BMSE, the MOTA index is 0.8 points higher. Therefore, it can be seen from table 2 that feature alignment is a key point that makes the tracking performance the most improved. AGNN also has a very significant effect on improving tracking performance, and the effect of BMSE is also good.

**Public Result:** In our method, the ResNet101-FPN method achieves the best results, so the backbone in our model uses the ResNet101-FPN structure.

It can be seen from the table that whether it is the Public data set of MOT16 or MOT17, our MOTA, MT, ML, FN indexes are the best among all the results, and the IDF1 index is also almost optimal.

**Private Result:** Similarly, our model backbone uses the structure of ResNet101-FPN, and our detector uses the de-

| Backbone       | MOTA↑ | IDF1↑ | FP↓ | FN↓ | IDSw↓ |
|----------------|-------|-------|-----|-----|-------|
| VGG16          | 56.6  | 55.1  | 4458| 67446| 1202  |
| ResNet34       | 57.2  | 55.6  | 4371| 66479| 1245  |
| ResNet50       | 58.8  | 57.1  | 4266| 63966| 1168  |
| ResNet101      | 59.7  | 58.5  | 4038| 62672| 1174  |
| ResNet101-FPN  | 61.2  | 59.1  | 3314| 61057| 986   |

Table 1. Backbone experiment on the MOT17 Public train dataset, half of the data is used as the training set, and the other half is used as the validation set.
Table 2. Ablation experiment on the MOT17 Public data set, half of the data is used as the training set, and the other half is used as the validation set. And the backbone is ResNet101-FPN.

| Method   | MOT16 | MOT17 |
|----------|-------|-------|
|          |     IDF1 |    MT |   ML |  FP |  FN | IDSw |
| RAR16     | 45.9  | 13.2  | 34.7 | 2675 | 5196 |       |
| AMIR      | 47.2  | 47.2  | 37.8 | 2675 | 5196 |       |
| MOT1D     | 47.6  | 50.9  | 37.3 | 2675 | 5196 |       |
| STRN      | 48.5  | 53.7  | 37.3 | 2675 | 5196 |       |
| KCF       | 48.8  | 43.7  | 37.3 | 2675 | 5196 |       |
| TrackerV1 | 54.5  | 52.5  | 37.3 | 2675 | 5196 |       |
| DeepMOT   | 54.8  | 53.4  | 37.3 | 2675 | 5196 |       |
| MPNTrack  | 58.6  | 52.5  | 37.3 | 2675 | 5196 |       |
| LiTs       | 61.3  | 64.7  | 37.3 | 2675 | 5196 |       |
| Unsup     | 62.4  | 58.5  | 37.3 | 2675 | 5196 |       |
| ours      | 66.5  | 62.3  | 29.5  | 26.9 | 4877 | 56600 |

Table 3. Result on MOT16 and MOT17 Public dataset.

| Method   | MOT16 | MOT17 |
|----------|-------|-------|
|          |     IDF1 |    MT |   ML |  FP |  FN | IDSw |
| EAMTT     | 52.5  | 33.2  | 34.9 | 4407 | 81223 | 910 |
| IOL       | 57.1  | 46.9  | 33.2 | 5702 | 70278 | 2167 |
| SORTwHPD16 | 59.8  | 53.8  | 25.4  | 22.7 | 8698 | 36245 |
| DeepSORT  | 61.4  | 62.2  | 32.8  | 18.2 | 12852 | 56668 |
| RAR16wVGG | 63.0  | 63.8  | 39.9  | 22.1 | 13663 | 53248 |
| KCF       | 66.1  | 65.1  | 34.0  | 20.8 | 5061 | 59514 |
| Tube_TK_Poi | 66.9  | 62.2  | 39.0  | 16.1 | 11544 | 47502 |
| CTrackerV1| 67.6  | 57.2  | 32.3  | 23.1 | 8934 | 48305 |
| ours      | 74.9  | 72.8  | 44.7  | 15.9 | 10163 | 34484 |
| Tube_TK   | 76.2  | 68.6  | 51.1  | 13.6 | 32796 | 98475 |

Table 4. Result on MOT16 and MOT17 Private dataset.

Table 5. Ablation experiment on the MOT17 Public data set, half of the data is used as the training set, and the other half is used as the validation set. Feature alignment method used by default.

| Backbone | AGNN | BMSE | MOT16 | MOT17 |
|----------|------|------|-------|-------|
|          |     IDF1 |    MT |   ML |  FP |  FN | IDSw |
| VGG16    | 62.8  | 61.7  | 39.4  | 2675 | 5196 |       |
| ResNet34 | 63.0  | 62.0  | 3804  | 5196 | 57610 | 1105 |
| ResNet50 | 63.4  | 62.5  | 3845  | 5196 | 57610 | 1027 |
| ResNet101| 64.0  | 62.8  | 3761  | 5196 | 57610 | 1069 |
| VGG16    | 67.7  | 65.3  | 3384  | 5196 | 57610 | 1013 |
| ResNet34 | 67.8  | 65.3  | 3199  | 5196 | 57610 | 1067 |
| ResNet50 | 67.8  | 65.3  | 3199  | 5196 | 57610 | 1067 |
| ResNet101| 68.0  | 65.4  | 3147  | 5196 | 57610 | 1079 |
| VGG16    | 68.5  | 66.1  | 3112  | 5196 | 57610 | 1043 |
| ResNet34 | 68.6  | 66.1  | 3066  | 5196 | 57610 | 1043 |
| ResNet50 | 68.6  | 66.1  | 3066  | 5196 | 57610 | 1043 |
| ResNet101| 68.7  | 66.3  | 2996  | 5196 | 57610 | 1043 |
| AGNN     | 65.5  | 63.8  | 2897  | 5196 | 57610 | 1043 |

5. Conclusion

In this work, we further studied how to extract better features of the objects to obtain better data association. We first predict the position of the historical objects in the current frame and extract the features, so that the feature can be aligned. Then input them into adaptive Adaptive Graph Neural Network together with the detected features, update the features that aggregate the spatiotemporal global position and appearance information, and propose the Balanced MSE Loss during training, so that the Neural Network can learn better data distribution. The No.1 result on the MOT Challenge reflects the tremendous superiority of our method.

Acknowledgement

This work was supported by Major Scientific Research Project of Zhejiang Lab (No. 2019DB0ZX01).
References

[1] Zhou, Xingyi and Koltun, Vladlen and Krähenbühl, Philipp.: Tracking Objects as Points. In: ECCV(2020).

[2] Zhang, Yifu and Wang, Chunyu and Wang, Xing-gang and Zeng, Wenjun and Liu, Wenyu.: A Simple Baseline for Multi-Object Tracking. In: arXiv preprint arXiv:2004.01888(2020).

[3] Bouguet, Jean-Yves Pyramidal.: Implementation of the Lucas Kanade Feature Tracker Description of the algorithm. In: Intel Corporation Microprocessor Research Labs (2000).

[4] Milan, A., Leal-Taixé, L., Reid, I., Roth, S. & Schindler, K.: MOT16: A Benchmark for Multi-Object Tracking. In: arXiv preprint arXiv:1603.00831(2016).

[5] Baker, S., Scharstein, D., Lewis, J.P. et al.: A Database and Evaluation Methodology for Optical Flow. In: Int J Comput Vis 92, 1-31(2011). https://doi.org/10.1007/s11263-010-0390-2

[6] Jianbo Shi, & Tomasi.: Good features to track. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR(1994). doi:10.1109/cvpr.1994.323794

[7] Jean-yves Bouguet.: Pyramidal implementation of the Lucas Kanade feature tracker. In: Intel Corporation, Microprocessor Research Labs(2000).

[8] G. Farneb?ck.: Two-Frame Motion Estimation Based on Polynomial Expansion.In: Lecture Notes in Computer Science, pp. 363-370(2003).

[9] Tao, M., Bai, J., Kohli, P., Paris, S.: SimpleFlow: A Non-iterative, Sublinear Optical Flow Algorithm. In: Computer Graphics Forum, 31(2pt1), 345–353. (2012). doi:10.1111/j.1467-8659.2012.03013.x

[10] Berthold K.P. Horn, Brian G. Schunck.: Determining optical flow. In: Artificial Intelligence, Volume 17, Issues 1-3, Pages 185-203(1981), ISSN 0004-3702, https://doi.org/10.1016/0004-3702(81)90024-2

[11] Bruce D. Lucas and Takeo Kanade.: An iterative image registration technique with an application to stereo vision. In: In Proceedings of the 7th international joint conference on Artificial intelligence - Volume 2 (IJCAI’81). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 674-679(1981).

[12] Farneback, G. (n.d.).: Very high accuracy velocity estimation using orientation tensors, parametric motion, and simultaneous segmentation of the motion field. In: Proceedings Eighth IEEE International Conference on Computer Vision. ICCV(2001). doi:10.1109/iccv.2001.937514

[13] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, P. H?usser, C. Haz?rba?, V. Golkov, P. Smagt, D. Cremers, Thomas Brox.: FlowNet: Learning Optical Flow with Convolutional Networks. In: IEEE International Conference on Computer Vision (ICCV), 2015.

[14] Ilg Eddy, Mayer Nikolaus, Saikia Tonmoy, Keuper Margret, Dosovitskiy Alexey, Brox, Thomas.: FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In: Conference: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1647-1655(2017).

[15] Scarselli F , Gori M , Tsoi A C , et al. The Graph Neural Network Model[J]. IEEE Transactions on Neural Networks, 2009, 20(1):61.

[16] Bruna J , Zaremba W , Szlam A , et al. Spectral Networks and Locally Connected Networks on Graphs[J]. Computer ence, 2013.

[17] Wenhan Luo and Junliang Xing and Anton Milan and Xiaqin Zhang and Wei Liu and Xiaowei Zhao and Tae-Kyun Kim.: Multiple Object Tracking: A Literature Review. In: arXiv, cs.CV(2014).

[18] Keni B , Rainer S . Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics[J]. Eurasp Journal on Image & Video Processing, 2008, 2008(1):246309.

[19] Ren S , He K , Girshick R , et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2015, 39(6).

[20] H. W. Kuhn. The hungarian method for the assignment problem. Naval research logistics quarterly, 1955.

[21] A. Bewley, Z. Ge, L. Ott, F. Ramos and B. Upcroft, "Simple online and realtime tracking," 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, 2016, pp. 3464-3468, doi: 10.1109/ICIP.2016.7533003.

[22] N. Wojke, A. Bewley and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, 2017, pp. 3645-3649, doi: 10.1109/ICIP.2017.8296962.
[23] K. Liu, Y. Shen and L. Chen, "Simple online and real-time tracking with spherical panoramic camera," 2018 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, 2018, pp. 1-6, doi: 10.1109/ICCE.2018.8326132.

[24] Fu H., Wu L., Jian M., Yang Y., Wang X. (2019) MF-SORT: Simple Online and Realtime Tracking with Motion Features. In: Zhao Y., Barnes N., Chen B., Westermann R., Kong X., Lin C. (eds) Image and Graphics. ICIG 2019. Lecture Notes in Computer Science, vol 11901. Springer, Cham. https://doi.org/10.1007/978-3-030-34120-6_13

[25] Sergey Menshov, Yan Wang, Andrey Zhdanov, Eugene Varlamov, Dmitry Zhdanov, "Simple online and realtime tracking people with new “soft-iot” metric," Proc. SPIE 11342, AOPC 2019: AI in Optics and Photonics, 113420M (18 December 2019); https://doi.org/10.1117/12.2547922

[26] X. Hou, Y. Wang and L. Chau, “Vehicle Tracking Using Deep SORT with Low Confidence Track Filtering,” 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Taipei, Taiwan, 2019, pp. 1-6, doi: 10.1109/AVSS.2019.8909903.

[27] Maher, A., Taha, H. Zhang, B. Realtime multi-aircraft tracking in aerial scene with deep orientation network. J Real-Time Image Proc 15, 495–507 (2018). https://doi.org/10.1007/s11554-018-0780-1

[28] S. Caelles Prat, “Video Object Segmentation by Tracking Structured Key Points and Contours,” Projecte Final de Master Oficial, UPC, Escola Tècnica Superior d’Enginyeria de Telecomunicació de Barcelona, Departament de Teoria del Senyal i Comunicacions, 2016.

[29] Yi Z., Tao X U , Dong X U , et al. Coordinating Multiple Cameras to Assist Tracking Moving Objects Based on Network Topological Structure[J]. Geomatics Information ence of Wuhan University, 2017, 42(8):1117-1122.

[30] Malinovskiy, Yegor Wu, Yao-Jan Wang, Y.. (2009). Video-Based Vehicle Detection and Tracking Using Spatiotemporal Maps. Transportation Research Record. 2121. 81-89. 10.3141/2121-09.

[31] Mittal, A. , "M2Tracker: A Multi-View Approach to Segmenting and Tracking People in a Cluttered Scene." Proc.of European Conf.on Computer Vision (2002).

[32] Segen J . A camera-based system for tracking people in real time. In: International Conference on Pattern Recognition. IEEE, 1996.

[33] Khan S M , Shah M . Tracking Multiple Occluding People by Localizing on Multiple Scene Planes[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2009, 31(3):505-519.

[34] Cohen, Isaac and Ayache, Nicholas and Sulger, Patrick.: Tracking points on deformable objects using curvature information. In: ECCV(1992), 458–466.

[35] Zheng Zhang and Dazhi Cheng and Xizhou Zhu and Stephen Lin and Jifeng Dai.: Integrated Object Detection and Tracking with Tracklet-Conditioned Detection. In: arXiv:1811.11167 [cs.CV](2018).

[36] Zhichao Lu and Vivek Rathod and Ronny Votel and Jonathan Huang.: RetinaTrack: Online Single Stage Joint Detection and Tracking. In: arXiv:2003.13870 [cs.CV](2020).

[37] Yongxin Wang and Xinshuo Weng and Kris Kitani.: Joint Detection and Multi-Object Tracking with Graph Neural Networks. In: arXiv:2006.13164 [cs.CV](2020).

[38] Xingyi Zhou and Dequan Wang and Philipp Kr?henb¨uhl.: Objects as Points. In: arXiv:1904.07850 [cs.CV](2019).

[39] Lin T Y , Goyal P , Girshick R , et al. Focal Loss for Dense Object Detection[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2017, PP(99):2999-3007.

[40] Welch, G., Bishop, G., et al.: An introduction to the kalman filter (1995)

[41] T. N. Kipf and M. Welling. 2017. Semi-supervised classification with graph convolutional networks. In Proc. of ICLR

[42] K. Hammond, P. Vanderheyst, and R. Gribonval. 2011. Wavelets on graphs via spectral graph theory. Applied and Computational Harmonic Analysis, 30(2):129–150. DOI: 10.1016/j.acha.2010.04.005

[43] M. Defferrard, X. Bresson, and P. Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In Proc. of NIPS, pages 3844–3852.

[44] R. Li, S. Wang, F. Zhu, and J. Huang. 2018b. Adaptive graph convolutional neural networks. In Proc. of AAAI.
[45] D. K. Duvenaud, D. Maclaurin, J. Aguilera-Parraiguire, R. Gomez-Bombarelli, T. D. Hirzel, A. Aspuru-Guzik, and R. P. Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In Proc. of NIPS, pages 2224–2232.

[46] M. Niepert, M. Ahmed, and K. Kutkov. 2016. Learning convolutional neural networks for graphs. In Proc. of ICML, pages 2014–2023.

[47] J. Atwood and D. Towsley. 2016. Diffusion-convolutional neural networks. In Proc. of NIPS, pages 1993–2001.

[48] Zhuang and Q. Ma. 2018. Dual graph convolutional networks for graph-based semi-supervised classification. In Proc. of WWW. DOI: 10.1145/3178876.3186116

[49] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. Neural Computation, 9(8):1735–1780. DOI: 10.1162/neco.1997.9.8.1735.

[50] Y. Li, D. Tarlow, M. Brockschmidt, and R. S. Zemel. 2016. Gated graph sequence neural networks. In Proc. of ICLR.

[51] K. S. Tai, R. Socher, and C. D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In Proc. of IJCNLP, pages 1556–1566. DOI: 10.3115/v1/p15-1150.

[52] V. Zayats and M. Ostendorf. 2018. Conversation modeling on reddit using a graph-structured LSTM. TACL, 6:121–132. DOI: 10.1162/tacl_a_00009.

[53] N. Peng, H. Poon, C. Quirk, K. Toutanova, and W.-t. Yih. 2017. Cross-sentence N-ary relation extraction with graph LSTMs. TACL, 5:101–115. DOI: 10.1162/tacl_a_00049.

[54] X. Liang, X. Shen, J. Feng, L. Lin, and S. Yan. 2016. Semantic object parsing with graph LSTM. In Proc. of ECCV, pages 125–143. DOI: 10.1007/978-3-319-46448-0_8.

[55] Y. Zhang, Q. Liu, and L. Song. 2018c. Sentence-state LSTM for text representation. In Proc. of ACL, 1:317–327. DOI: 10.18653/v1/p18-1030.

[56] A. Vaswani, N. Shazeer, N. Parmar, L. Jones, J. Uszkoreit, A. N. Gomez, and L. Kaiser. 2017. Attention is all you need. In Proc. of NIPS, pages 5998–6008.

[57] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018. Graph attention networks. In Proc. of ICLR.

[58] A. Rahimi, T. Cohn, and T. Baldwin. 2018. Semi-supervised user geolocation via graph convolutional networks. In Proc. of ACL, 1:2009–2019. DOI: 10.18653/v1/p18-1187.

[59] J. G. Zilly, R. K. Srivastava, J. Koutnik, and J. Schmidhuber. 2016. Recurrent highway networks. In Proc. of ICML, pages 4189–4198.

[60] T. Pham, T. Tran, D. Phung, and S. Venkatesh. 2017. Column networks for collective classification. In Proc. of AAAI.

[61] K. Xu, C. Li, Y. Tian, T. Sonobe, K. Kawarabayashi, and S. Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In Proc. of ICML, pages 5449–5458.

[62] G. Li, M. Muller, A. Thabet, and B. Ghanem. 2019. DeepGCNs: Can GCNs go as deep as CNNs? In Proc. of ICCV.

[63] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. 2017. Densely connected convolutional networks. In Proc. of CVPR, pages 4700–4708. DOI: 10.1109/cvpr.2017.243.

[64] F. Yu and V. Koltun. 2015. Multi-scale context aggregation by dilated convolutions. ArXiv Preprint ArXiv:1511.07122.

[65] Bernardin, K. & Stiefelhagen, R. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. Image and Video Processing, 2008(1):1-10, 2008.

[66] Ristani, E., Solera, F., Zou, R., Cucchiara, R. & Tomasi, C. Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. In ECCV workshop on Benchmarking Multi-Target Tracking, 2016.

[67] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. Tracking without bells and whistles. arXiv preprint arXiv:1903.05625, 2019.

[68] Fang, K., Xiang, Y., Li, X., Savarese, S.: Recurrent autoregressive networks for online multi-object tracking. In: WACV. pp. 466–475 (2018).

[69] Sadeghian, A., Alahi, A., Savarese, S.: Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In: ICCV. pp. 300–311 (2017).

[70] Chu, P., Fan, H., Tan, C.C., Ling, H.: Online multi-object tracking with instance-aware tracker and dynamic model refreshment. In: WACV. pp. 161–170 (2019).
[71] Chen, L., Ai, H., Zhuang, Z., Shang, C.: Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In: ICME. pp. 1–6 (2018).

[72] Xu, J., Cao, Y., Zhang, Z., Hu, H.: Spatial-temporal relation networks for multi-object tracking. In: ICCV. pp. 3988–3998 (2019).

[73] Xu, Y., Osep, A., Ban, Y., Horaud, R., Leal-Taixé, L., Alameda-Pineda, X.: How to train your deep multi-object tracker. In: CVPR (2020).

[74] G. Brasó and L. Leal-Taixé, "Learning a Neural Solver for Multiple Object Tracking," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 6246-6256, doi: 10.1109/CVPR42600.2020.00628.

[75] Hornakova, A., Henschel, R., Rosenhahn, B., and Swoboda, P.: 2020, arXiv e-prints, arXiv:2006.14550.

[76] Karthik S, Prabhu A, Gandhi V. Simple Unsupervised Multi-Object Tracking[J]. 2020.

[77] Bergmann, P., Meinhardt, T., Leal-Taixé, L.: Tracking without bells and whistles. In: ICCV(2019).

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

[79] Zhu, J., Yang, H., Liu, N., Kim, M., Zhang, W., Yang, M.H.: Online multi-object tracking with dual matching attention networks. In: ECCV. pp. 366–382 (2018).

[80] Chen, L., Ai, H., Zhuang, Z., Shang, C.: Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In: ICME. pp. 1–6(2018).

[81] Chu, P., Ling, H.: Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In: ICCV. pp. 6172–6181 (2019).

[82] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Lyu, W. Ke, Z. Xiong. Long-term Tracking with Deep Tracklet Association. In IEEE Transactions on Image Processing, 2020.

[83] Sanchez-Matilla, R., Poiesi, F., Cavallaro, A.: Online multi-target tracking with strong and weak detections. In: European Conference on Computer Vision. pp. 84–99. Springer (2016).

[84] E. Bochinski, V. Eiselein, T. Sikora. High-Speed Tracking-by-Detection Without Using Image Information. In International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE AVSS 2017, 2017.

[85] Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP). pp. 3464–3468. IEEE (2016).

[86] Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP). pp. 3645–3649. IEEE (2017).

[87] Fang, K., Xiang, Y., Li, X., Savarese, S.: Recurrent autoregressive networks for online multi-object tracking. In: 2018 IEEE Winter Conference on Applications of Computer Vision (W ACV). pp. 466–475. IEEE (2018).

[88] Yu, F., Li, W., Li, Q., Liu, Y., Shi, X., Yan, J.: Poi: Multiple object tracking with high performance detection and appearance feature. In: European Conference on Computer Vision. pp. 36–42. Springer (2016).

[89] B. Pang, Y. Li, Y. Zhang, M. Li, C. Lu. TubeTK: Adopting Tubes to Track Multi-Object in a One-Step Training Model. In CVPR, 2020.

[90] J. Peng, C. Wang, et.al. Chained-Tracker: Chaining Paired Attentive Regression Results for End-to-End Joint Multiple-Object Detection and Tracking. In ECCV Spotlight, 2020.