Multi-object Tracking in Remote Sensing Video Based on Motion and Multi-scale Local Cost Volume

Xuqian Zhu and Bin Zhang(✉)

Xi'an Jiaotong University, Xi'an 710054, China
bzhang82@xjtu.edu.cn

Abstract. Multi-object tracking (MOT) in remote sensing videos is significant in many application scenarios. That task in ordinary scenarios has been widely used in vehicle tracking and monitoring. However, due to the peculiarities of remote sensing video, many new challenges will be brought to the task. So many frameworks for ordinary scenarios are inappropriate. In this paper, we propose a novel network called a multi-scale local cost volume network (MLCVNet) that can extract multi-scale features and inter-frame motion information. We use a multi-scale local cost volume module to obtain the object's displacement information between current and historical frames, and the historic features will be mapped into the current features to obtain enhanced features through which objects can be detected and tracked. Some experiments have been conducted on remote sensing videos, which are collected from the Jilin-1 satellite, and the results have demonstrated the effectiveness and robustness of the proposed method. Experimental results show that our method achieves state-of-art performance.

Keywords: multi-object tracking · remote sensing · motion information · local cost volume

1 Instruction

MOT in ordinary videos has received widespread attention and research, and many excellent results have been applied to various scenarios. MOT in remote sensing videos is even more critical in some applications. Remote sensing satellites can easily observe large areas of target regions, track vehicle flow, and support smart city transportation. MOT in remote sensing videos has become an important research topic in remote sensing image processing. However, compared to ordinary videos, remote sensing videos face many new challenges: 1) Low discrimination between objects and the background. 2) High background noise. 3) Small object areas. 4) Lack of detailed features. 5) Cloud cover.

These challenges pose great difficulties to object tracking in remote sensing videos, so we require effective methods to overcome them. In this article, motivated by the TraDeS [1] for MOT in ordinary video, a new MOT framework is proposed, which can be applied to remote sensing videos for tracking tiny objects.

The main contributions of this paper are as follows:

1) We propose a multi-scale local cost volume mechanism that could accurately represent the motion offset of small objects than the baseline.
2) A head representing the direction of object motion is added to the network output head. The detection branch and tracking branch of output heads are connected to the not enhanced current feature and final enhanced feature, respectively.

Our proposed MOT method based on MLCVNet is designed to address some challenges in remote sensing videos. Compared to [1] in remote sensing videos, our MLCV model can more accurately match the same object between previous and current frames while effectively reducing computation. Without introducing extra network branches with excessive computational requirements as in DSFNet [2], the proposed method can meet the real-time tracking needs of multi-object in remote sensing videos while performing well. Detailed network architecture will discuss in Sect. 3.

2 Related Works

The concept of object tracking was proposed by Wax N in the 1960s [3] and was applied to pedestrian tracking. Since then, the field of object tracking has received much attention from researchers, with new theories and research results constantly emerging and being innovated. This article is about multi-object tracking, and the current research status will be briefly described below.

2.1 MOT in Ordinary Video

In the traditional object tracking framework, detection is done by establishing an appearance model to identify the object's identity, including unique features that distinguish different objects and are used for subsequent association tracking. Many of the MOT frameworks that have emerged in recent years are based on deep learning. They are roughly divided into two types of tracking frameworks: tracking-by-detection (TBD) and joint detection and tracking (JDT) [4].

The appearance model of an object can be represented by different object attributes, including color, texture, gradient, motion, and optical flow, to identify the object uniquely. By extracting a class of features or joint features of the object, it is possible to distinguish the object from the background and thus distinguish different objects. Many traditional multi-object tracking frameworks fall into this category [5–7].

Compared to traditional methods, deep learning does not require manual feature extraction and can obtain richer feature representations, often achieving better results. DeepSort [8] improves the Sort [9] method that utilizes deep learning. In the object detection phase, a detection network is used to detect the object. Then the detected object is passed to a re-identification (ReID) appearance feature extraction network for feature extraction, followed by the tracking process. The MOTDT [10] framework fully utilizes the advantages of deep neural networks to address prominent issues in TBD, such as unreliable detection and intra-class occlusion. The detection part in D&T [11] is based on the R-FCN fully convolutional network, and the tracking part incorporates the tracking ideas based on correlation and regression from single-object tracking methods into the front-end detection framework, implementing this multi-object tracking method.

In recent years, more and more research has leaned towards one-stage methods, which only require one network to accomplish object detection and appearance feature extraction simultaneously. JDE [12] proposes a network model that can integrate object detection and ReID tasks into one by incorporating the appearance ReID model into the one-shot detector. FairMOT [13] points out multiple imbalances in general anchor-based methods and proposes an improvement. FairMOT is a tracking method based on the anchor-free feature extraction network DLA [14], which adds a ReID branch on top of the detection task. CenterTrack [15] is an improvement on the CenterNet [16], adding a branch to the detection output branch to reflect the position movement vector of the object between two frames, thus implementing multi-object tracking in one network. TraDeS [1] proposes a new online joint detection and tracking model with tracking features to assist with end-to-end detection tasks. It infers the object tracking offset based on cost volume and then uses it to propagate the object features from a previous moment to improve the current frame's object detection and segmentation tasks.

2.2 MOT in Remote Sensing Video

Currently, some multi-object tracking frameworks based on remote sensing videos have been proposed. Du et al. [17] proposed a specific strategy for constructing a more robust tracker using a kernel correlation filtering (KCF) tracker and a three-frame differencing algorithm. Guo et al. [18] proposed a correlation filter Kalman filter (CFKF) tracker, which is a tracking algorithm based on a fast correlation filter (CF) for satellite video object tracking. Shao et al. [19] proposed a velocity correlation filter (VCF) algorithm to overcome the problem of insufficient brightness and color features of remote sensing video objects. Xuan et al. [20] proposed a new motion estimation (ME) algorithm based on the kernel correlation filtering (KCF) algorithm, which combines Kalman filtering and motion smoothing trajectory to reduce the boundary effects of the kernel correlation filtering algorithm.

He et al. [21] proposed a graph-based multi-task reasoning tracking framework, which models multi-object tracking as a graph feature information fusion process based on message inference. Xiao et al. [2] proposed a two-stream network that integrates object motion information and object appearance information, which the authors refer to as dynamic information and static information, respectively. It was originally used for object detection tasks in remote sensing videos, but its network can also be used for multi-object tracking.

3 Network Architecture

The overall architecture of MLCVNet consists of four main parts, as shown in Fig. 1. During the training process, there are three inputs, the current frame $\mathbf{I}^t$ at time t, the historical frame $\mathbf{I}^{t-\tau}$, and the heatmap $\mathbf{P}^{t-\tau}$ of the historical frame.

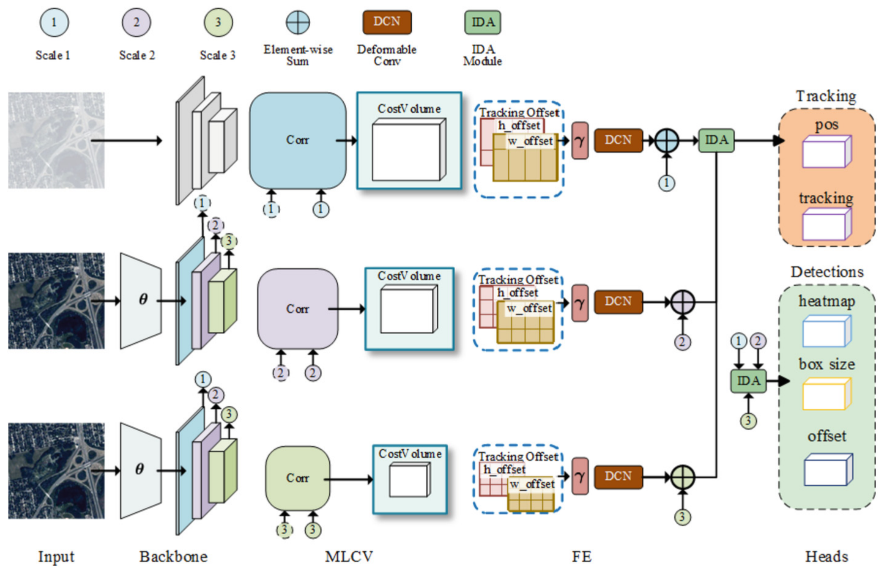

Fig. 1. The detailed network architecture includes a DLA-34 backbone, which extracts three scales of feature maps $\mathbf{f}_s^f$ and $\mathbf{f}_s^{t-\tau}$ from the input frames $\mathbf{I}^t$ and $\mathbf{I}^{t-\tau}$, respectively. These feature maps are then used in a correlation operation to produce local cost volume $\mathbf{C}_s$, which is further processed using a template operation to obtain the offset matrix $\mathbf{O}_s$. The FE module extracts motion transformation features at three scales. The resulting fusion feature is combined with the current feature to obtain an enhanced feature map connected to the output branches to produce the final outputs.

3.1 Multi-scale Local Cost Volume

Firstly, a DLA-34 network was used to extract multi-scale features from the image. The input image size of the backbone is $3 \times H_i \times W_i$. After passing it, $\mathbf{I}^t$ and $\mathbf{I}^{t-\tau}$ get three scales of feature maps, which are $\mathbf{f}_s^t \in \mathbb{R}^{C_s^f \times H_s^f \times W_s^f}$ and $\mathbf{f}_s^{t-\tau}$, respectively. The down-sampling ratios of the feature maps are 2, 4, and 8, denoted as s, where $H_s^f = \frac{H_i}{s}$, $W_s^f = \frac{W_i}{s}$.

The second part is MLCV module. The feature maps $\mathbf{f}_s^t$ and $\mathbf{f}_s^{t-\tau}$ at corresponding scales were first handled by a correlation operation to obtain a local cost volume $\mathbf{C}_s \in \mathbb{R}^{H_s^d \times W_s^d \times H_s^f \times W_s^f}$ between the current frame and the previous frame. Specifically, a correlation operation is performed at each pixel position in the feature map $\mathbf{f}_s^t$, using a correlation kernel $\mathbf{K}_{x,y}^t$ of size $k \times k$ centered at the position $\left(x_f, y_f\right)$. Then a search window of size $H_s^d \times W_s^d$ centered at the corresponding position in feature map $\mathbf{f}_s^{t-\tau}$ is slid and correlated, resulting in a vector $\mathbf{C}_{x_f, y_f} \in \mathbb{R}^{H_d \times W_d}$ of length $H_s^d \times W_s^d$, which stores the correlation values of the kernel of the feature $\mathbf{f}_s^t$ and all kernels in the window of the feature $\mathbf{f}_s^{t-\tau}$. This vector reflects the matching degree between the object in the current frame and the possible positions of the object in the previous frame. The process is shown in Fig. 2.

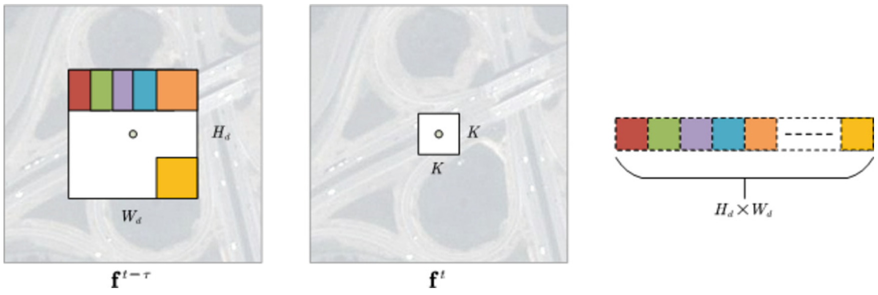

Fig. 2. During the correlation operation, a kernel centered at position (x, y) in feature map $\mathbf{f}^t$ is slid in a search window at the corresponding position in feature map $\mathbf{f}^{t-\tau}$. The kernel $\mathbf{K}^t_{x,y}$ correlates with every kernel $\mathbf{K}^{t-\tau}_{x',y'}$ in the search window, and each pair of kernels produces a value $\mathbf{C}_{x,y,x',y'}$.

If we ignore the difference in the scales s, the process of correlation operation is the same. The value $\mathbf{C}_{x,y,x',y'}$ in the vector is handled by doing an inner product of the two kernel vectors, which is obtained by the following equation:

$$\mathbf{C}_{x,y,x',y'} = \mathbf{K}^t_{x,y}\mathbf{K}^{t-\tau\top}_{x',y'} \tag{1}$$

A maximum value in the vector indicates the highest matching degree between the two kernels because they represent objects' partial features at the corresponding positions in the two frames. So, the highest matching value means these two objects are most likely to match. The complete correlation operation is performed for all positions in frame $\mathbf{f}^t_s$, resulting in a local cost volume $\mathbf{C}$. The operation is expressed as:

$$\mathbf{C} = \mathrm{Corr}\big(\mathbf{f}^t, \mathbf{f}^{t-\tau}, d', k\big) \tag{2}$$

where d' represents the displacement of the search window, $d' = \left\lfloor \frac{d}{2} \right\rfloor$, and $d = d' \times 2 + 1$. k is the size of the correlation kernel, and the default value is set to 3.

The local cost volume $\mathbf{C}$ obtained by the correlation operation is a four-dimensional matrix with dimensions $\left[1, H_d \times W_d, H_f, W_f\right]$. It needs to be reshaped into dimensions $\left[1, H_d, W_d, H_f \times W_f\right]$, and then the maximum value is taken in the second and third dimensions to obtain the maximum cost volume values $\mathbf{C}_H \in \mathbb{R}^{H_f \times W_f \times H_d}$ and $\mathbf{C}_W \in \mathbb{R}^{H_f \times W_f \times W_d}$ of each pixel between feature map $\mathbf{f}^t$ ant the corresponding search window of $\mathbf{f}^{t-\tau}$ in the height and width directions, respectively. Then, using the preset vertical and horizontal offset templates $\mathbf{V} \in \mathbb{R}^{H_f \times W_f \times H_d}$ and $\mathbf{H} \in \mathbb{R}^{H_f \times W_f \times W_d}$, the vectors of the two templates at position (i, j) are denoted as $\mathbf{V}_{i,j} \in \mathbb{R}^{H_d}$ and $\mathbf{H}_{i,j} \in \mathbb{R}^{W_d}$, respectively. By multiplying the values from the softmaxed cost volume in the two directions, the position offset vector $\mathbf{O}_{i,j} = \left[\mathbf{C}^H_{i,j}\mathbf{V}_{i,j}, \mathbf{C}^W_{i,j}\mathbf{H}_{i,j}\right]^\top$ with the maximum matching value from time t to $t - \tau$ at position (i, j) can be obtained, and the tracking offset matrix $\mathbf{O} \in \mathbb{R}^{H_f \times W_f \times 2}$ of all pixels in the feature map can be obtained.

The third part is the feature enhancement module, which is simplification of the MFW module [1], through which the enhanced feature $\widetilde{\mathbf{f}}^t_q$ will be obtained. But our FE model works on multi-scale, and multi-scale features will be fused by the IDA model [14]. After that we get $\widetilde{\mathbf{f}}_s^t$ at three scales, these features will be fused by IDA model at last. The final enhanced feature $\widetilde{\mathbf{f}}^t$ is obtained in Eq. 3.

$$\widetilde{\mathbf{f}}^t = IDA\left(\mathbf{f}_s^t\right), \quad s = 2, 4, 8 \tag{3}$$

3.2 Motion Direction Head

The output heads, which includes the detection and tracking branches. The detection branch is similar with [15], while the tracking branch contains a tracking offset head and a pos head. Each head of the detection branch is followed by a convolutional operation after the current time feature $\mathbf{f}^t$ to predict the corresponding information. The tracking offset head and pos head output by directly connecting a convolution to the final enhanced feature $\widetilde{\mathbf{f}}^t$, which are used to predict the position offset of the object, and the motion direction of the object, respectively.

For the tracking branch, in order to learn the position offset of the object more accurately between t and $t - \tau$, some improvements have been made. In the heads of [1], the enhanced feature $\widetilde{\mathbf{f}}^t$ is connected to both the detection branch and the tracking branch, which is detrimental to the learning of the tracking offset head and pos head in the tracking branch. However, in our framework, the enhanced feature is only connected to the tracking branch to ensure that the MLCV and FE modules can better learn the enhanced features for object position offset.

For the pos head, which is used to predict the direction of object motion. Because for remote sensing, the video is captured from a top-down perspective, so the motion of the object in the video is equivalent to moving on a plane. The enhanced feature hides motion information of object within it. Therefore, it is considered to output the direction of object motion in the multi-frames as a header so that the network can learn the object's motion information accurately. Specifically, the ground is roughly divided into eight directions: up, down, left, right, upper right, lower right, lower left, and upper left. That is, each pixel is represented by a vector of length 8. If an object exists in that pixel position, the direction index corresponding to the object's motion direction in that vector is set to 1, and others are set to 0. As shown in Fig. 3, the object relative to the origin of the coordinate system in the right-side figure moves in the upper right direction compared to its position in the left figure.

The size of the pos feature map is $8 \times H_o \times W_o$, where $H_o = H_i/2, W_o = W_i/2$. If a real object box $\mathbf{b}^i = \left(x_1^i, y_1^i, x_2^i, y_2^i\right)$ is in the image, with center at the position $\left(c_x^i, c_y^i\right)$. The vector at the position $\left(\tilde{c}_x^i, \tilde{c}_y^i\right) = \left(\left\lfloor \frac{c_x^i}{2} \right\rfloor, \left\lfloor \frac{c_y^i}{2} \right\rfloor\right)$ on the feature represents the motion direction of an object i. For example, in Fig. 4, the motion direction of the object is upper right, and the vector at the corresponding position in the true label of pos is $\mathbf{p}^i = [0, 1, 0, 0, 0, 0, 0, 0]$. The loss function of the motion direction header uses the Mean Squared Error loss, and the loss function is shown in Eq. 4.

$$L_{pos} = \frac{1}{N} \sum_{i=1}^{N} \left(\widehat{\mathbf{p}}^i - \mathbf{p}^i\right)^2 \tag{4}$$

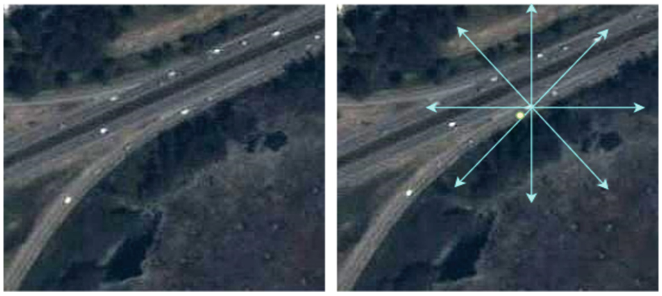

Fig. 3. An example to illustrate the ground truth of pos head, the object relative to the origin of the coordinate system in the right figure moves in the upper right direction compared to its position in the left figure.

where N is the number of objects in the video frames.

The overall loss of network consists of two head branches together, and the total loss function can be obtained by summing up the different branches by assigning certain weights to them.

$$L_{total} = w_1 L_{heat} + w_2 L_{box} + w_3 L_{tr} + w_4 L_{pos} \tag{5}$$

where $w_1, w_2, w_3,$ and w_4 represent weight values for the four losses. L_{heat} and L_{box} are the detection losses as in [15]. L_{tr} is the tracking offset loss as in [1].

4 Experiments

4.1 Datasets and Implementation Details

To validate the performance of the proposed multi-object tracking framework, it is tested on a remote sensing video dataset. The dataset used in this paper is provided by the DSFNet, which proposed a method for object detection in remote sensing images, but the dataset format also supports MOTChallenge multi-object tracking. The videos in the dataset were captured by the Jilin-1 video satellite, and the training set contains 72 videos, while the test set contains 7 videos. During the training process, some image data augmentation techniques are applied, including flipping and color space transformation. The experiment mainly focuses on vehicle-like objects in the video.

Table 1. Detail experimental environments.

Environment	Version
Operate system	Ubuntu18.04
CPU	12-core Intel(R) Xeon(R) Platinum 8255C
GPU	RTX3090 24G
CUDA	10.3
Python	3.8
Pytorch	1.7

The specific experimental environments are shown in Table 1.

To train MLCVNet, we utilized the Adam optimizer with a batch size of 8 and an initial learning rate of 1.25×10^{-4}. The entire network was trained for 30 epochs before termination. In the MLCV module, we used a kernel size of 3×3 when down-sampling by a factor of 2 and a search window size of 7×7. For down-sampling by a factor of 4, the kernel size was set to 3×3, and the search window size was set to 5×5. When down-sampling by a factor of 8, we used a kernel size of 1×1 and a search window size of 3×3.

In order to verify the effectiveness of the method proposed in this thesis, some multi-object tracking frameworks with outstanding performance are selected for comparison, including CenterTrack [15], FairMOT [13], DSFNet [2], and TraDeS [1]. The experimental results are shown in Table 2.

Table 2. Experimental results of each tracking framework on the remote sensing video test set

Method	IDs↓	MT↑	ML↓	FP↓	FN↓	IDF1↑	MOTA↑	FPS↑
CenterTrack	2618	4.4%	47.3%	25.6%	59.6%	24.9%	11.9%	**26**
FairMOT	840	14.8%	25.9%	37.9%	48.4%	42.4%	12.8%	22
TraDeS	1869	21.1%	45.1%	24.4%	53.4%	36.8%	20.2%	16
DSFNet	**740**	**51.9%**	17.6%	**23.3%**	25.4%	70.0%	50.5%	2
MLCVNet	750	50.7%	**13.8%**	24.2%	**24.0%**	**70.7%**	**51.0%**	14

Some frameworks that perform well in regular videos, such as CenterTrack, FairMOT, and TraDeS, rely mainly on object appearance features for learning. This leads to poor results in remote sensing videos because of some prominent characteristics of remote sensing videos. The DSFNet framework integrates object motion information and object appearance information, which the authors refer to as dynamic information and static information, respectively. It was originally used for object detection tasks in remote sensing videos, but its network can also be used for multi-object tracking. Therefore, it performs well in the test set, but due to the addition of a motion information branch and the output size of its network being consistent with the original image size, the computational cost of the entire network is very high, resulting in an FPS of only 2 during testing.

Several indicators of MLCVNet have reached the best level, with the MOTA indicator reaching 51.0%, consistent with the [2]. Because the output feature map of the MLCVNet network is down-sampled by 2 times, and the motion branch is not introduced, the computational cost of the overall network is much smaller than that of [2], and its inference speed reaches 14 frames per second. The results are shown in Fig. 4, with one complete result image selected for each framework, and the FPS indicator of each framework is marked in the upper-left corner.

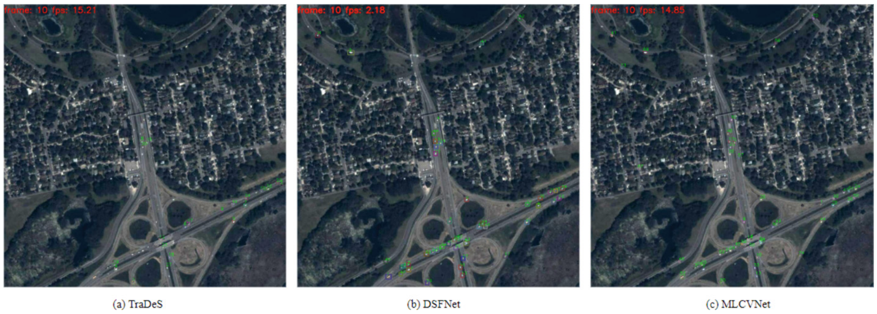

Fig. 4. Illustration of the FPS results for each framework run on the test set. (b) FPS indicator for DSFNet is only 2. (c) MLCVNet runs at 14 frames per second.

4.2 Ablation Studies

In order to prove the effectiveness of the three improvement schemes proposed in this paper, the ablation experiments are now conducted for these different schemes of the original baseline TraDeS, TraDeS with improved head connection (TraDeS*), improved TraDeS with added MLCV module (TraDeS*+MLCV), improved TraDeS with added MLCV module and pos head (MLCVNet), respectively, and the results of the experiments are shown in Table 3.

Table 3. Experimental results of ablation studies

Method	IDs	MT	ML	FP	FN	IDF1	MOTA
TraDeS	1869	21.1%	45.1%	24.4%	53.4%	36.8%	20.2%
TraDeS*	935	24.3%	35.4%	20.7%	47.7%	40.3%	30.6%
TraDeS*+MLCV	841	55.4%	10.6%	33.4%	23.3%	60.5%	42.5%
MLCVNet	750	50.7%	13.8%	24.2%	24.0%	70.7%	51.0%

From Table 3, it can be seen that both adding the MLCV module, adding the MLCV module and pos head to the baseline have improved accuracy, which fully proves the effectiveness of the MLCV module and pos head.

When adding the pos head to MLCVNet, it is to learn the tracking information of the object more accurately, but it can also serve as an auxiliary branch to learn detection information, which can effectively help learn the heatmap head information and improve the confidence of the learned objects' information. To verify this idea, the confidence information of all objects was extracted from the test set videos before and after adding the pos head to the framework. The confidence values were then divided into intervals of 0.1 within the range of [0, 1). The number of detected objects within each interval was counted and the results were plotted, as shown in Fig. 5.

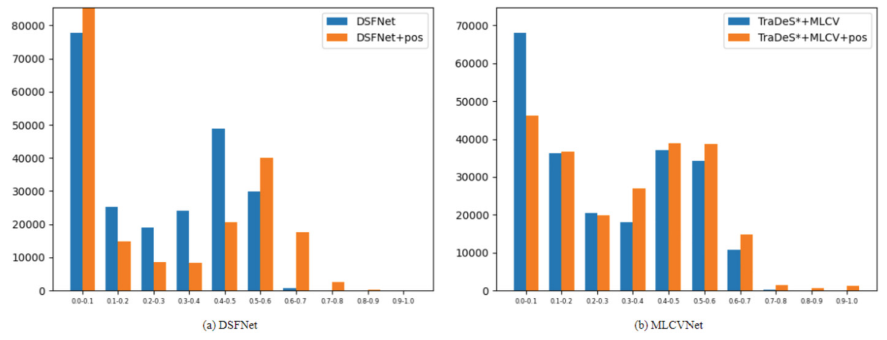

Fig. 5. The figure shows the change in the distribution of confidence scores of all detection results obtained when the two tracking frameworks run on the test set before and after adding the pos head.

The above figure shows that (a) is the result of testing on DSFNet. After adding the pos branch, the number of low confidence objects in DSFNet has decreased significantly, and more detection objects with confidence scores above 0.5 are output by the framework. Figure 5 (b) is the result of testing on the framework proposed in this paper. By comparing the MLCVNet network with and without the pos branch, it can be seen that the number of low confidence objects has slightly decreased and the number of objects with scores above 0.3 has increased. Although the overall effect is not as obvious as in (a), the overall distribution of confidence scores is also moving towards the high score interval. From the analysis of the results in Fig. 5 (a) and (b), it can be concluded that the pos branch proposed in this paper is effective in helping the network learn detection information of the objects.

5 Conclusion

In this paper, we propose a novel multi-object tracking framework for remote sensing videos based on the TraDeS framework. We make three improvements to address the prominent issues in remote sensing videos and achieve good performance in tracking tiny objects in terms of accuracy and real-time processing. Firstly, we improve the head connection of the framework, enabling the network to learn better tracking and detection information separately. Secondly, the MLCV module utilizes the kernel and local search window mechanism to extract the motion information of small objects in remote sensing videos more accurately. Lastly, we add a pos head to the tracking branch of the network's output head to represent the direction of object motion, which helps the network to learn more accurate object detection information. The results of comparative and ablation experiments show that the proposed method is effective and achieves excellent performance in multi-object tracking for remote sensing videos.

References

1. Wu, J., Cao, J., Song, L., Wang, Y., Yang, M., Yuan, J.: Track to detect and segment: an online multi-object tracker. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 12352–12361 (2021)
2. Xiao, C., et al.: DSFNet: dynamic and static fusion network for moving object detection in satellite videos. IEEE Geosci. Remote Sens. Lett. **19**, 1–5 (2021)
3. Wax, N.: Signal-to-noise improvement and the statistics of track populations. J. Appl. Phys. **26**, 586–595 (1955)
4. Wang, Q., Zheng, Y., Pan, P., Xu, Y.: Multiple object tracking with correlation learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3876–3886 (2021)
5. Ullah, M., Cheikh, F.A., Imran, A.S.: HoG based real-time multi-target tracking in bayesian framework. In: 2016 13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 416–422. IEEE (2016)
6. Andriyenko, A., Schindler, K.: Multi-target tracking by continuous energy minimization. In: CVPR 2011, pp. 1265–1272. IEEE (2011)
7. Yin, J., Wang, W., Meng, Q., Yang, R., Shen, J.: A unified object motion and affinity model for online multi-object tracking. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6768–6777 (2020)
8. 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)
9. 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)
10. Chen, L., Ai, H., Zhuang, Z., Shang, C.: Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In: 2018 IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6. IEEE (2018)
11. Feichtenhofer, C., Pinz, A., Zisserman, A.: Detect to track and track to detect. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3038–3046 (2017)
12. Wang, Z., Zheng, L., Liu, Y., Li, Y., Wang, S.: Towards real-time multi-object tracking. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12356, pp. 107–122. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58621-8_7
13. Zhang, Y., Wang, C., Wang, X., Zeng, W., Liu, W.: FairMOT: on the fairness of detection and re-identification in multiple object tracking. Int. J. Comput. Vision **129**, 3069–3087 (2021)
14. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2018)
15. Zhou, X., Koltun, V., Krähenbühl, P.: Tracking objects as points. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12349, pp. 474–490. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58548-8_28
16. Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., Tian, Q.: CenterNet: keypoint triplets for object detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6569–6578 (2019)
17. Du, B., Sun, Y., Cai, S., Wu, C., Du, Q.: Object tracking in satellite videos by fusing the kernel correlation filter and the three-frame-difference algorithm. IEEE Geosci. Remote Sens. Lett. **15**, 168–172 (2017)
18. Guo, Y., Yang, D., Chen, Z.: Object tracking on satellite videos: a correlation filter-based tracking method with trajectory correction by Kalman filter. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. **12**, 3538–3551 (2019)

19. Shao, J., Du, B., Wu, C., Zhang, L.: Tracking objects from satellite videos: a velocity feature based correlation filter. IEEE Trans. Geosci. Remote Sens. **57**, 7860–7871 (2019)
20. Xuan, S., Li, S., Han, M., Wan, X., Xia, G.-S.: Object tracking in satellite videos by improved correlation filters with motion estimations. IEEE Trans. Geosci. Remote Sens. **58**, 1074–1086 (2019)
21. He, Q., Sun, X., Yan, Z., Li, B., Fu, K.: Multi-object tracking in satellite videos with graph-based multitask modeling. IEEE Trans. Geosci. Remote Sens. **60**, 1–13 (2022)