Proposals from binary tree and spatio-temporal tunnel for temporal segmentation of rough videos*

ZHANG Yunzuo and GUO Kaina**

School of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang 050043, China

(Received 15 June 2022; Revised 22 August 2022)

©Tianjin University of Technology 2022

Existing temporal segmentation methods suffer from the problems of high computational complexity and complicated steps. To address this issue, we present a method that combines the binary tree and spatio-temporal tunnel (STT) for temporal segmentation of rough videos. First, we compute initial cumulative spatio-temporal flow to determine flow overflow of sub-video which is divided from a rough video. Second, the decision tree is generated by combining binary tree and balance factor to dynamically adjust the sampling line of the STT. Finally, pixels on the sampling line are extracted to generate an adaptive STT for temporal proposals. Experimental results show that the computational complexity of the proposed method is significantly better than that of the comparison methods while ensuring accuracy.

Document code: A **Article ID:** 1673-1905(2022)12-0763-6

DOI https://doi.org/10.1007/s11801-022-2103-9

With the development of intelligent media, video systems have been widely used in daily life and security protection. At the same time, a large amount of video data has been generated, which brings difficulties for people to quickly browse and retrieve[1,2]. Temporal segmentation can extract motion segments from videos with many still segments, as shown in Fig.1. It is an effective means to quickly obtain information from lengthy videos[3,4].

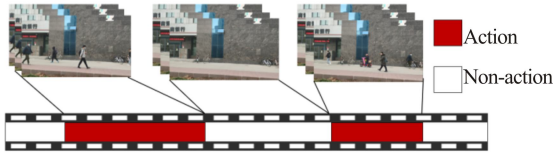


Fig.1 Our method retrieves the locations of the active regions in an unsupervised way

Most existing temporal action proposal methods detect motion objects based on superficial visual information such as the appearance and motion characteristics. These methods extract frames with motion objects to form action proposals. MURTAZA et al[5] realized temporal action proposals by calculating the energy of motion history images to reflect the spatio-temporal information of motion objects in videos. QU et al[6] and NAWAZ et al[7] perform a difference operation on the background image and the continuous inter-frame difference image after filtering. These methods also work well for slow-moving objects. GUO et al[8] improved the traditional optical flow methods, and improved the accuracy of action proposals by additionally calculating the average motion vector of the four boundaries. Recent methods produce either proposals using sparse reconstruction[9], sparse dictionaries[10], and spatio-temporal paths[11] based methods. However, these methods require processing the entire amount of video spatial data and detecting moving objects frame by frame for temporal action segmentation, which is computationally expensive and time-consuming for large-scale videos.

With the rise of artificial intelligence, deep learning methods have been widely used in the fields of object detection and action proposals[12-15]. YU et al[16] proposed an object-guided external storage network with storage efficiency handled by object-guided hard attention to selectively store valuable features. CHEN et al[17] proposed a unified framework for time propagation and cross-scale refinement to seek strategies that balance performance and cost. SHEN et al[18] proposed a set of learning-from-scratch object detector design principles, the key of which is deep supervision. QU et al[19] proposed a pyramid attention module to obtain key object information, and dedicated a dilated convolutional block to provide semantic information and geometric details for motion foregrounds. The methods based on deep learning have been proven to be effective, but there are

* This work has been supported by the National Natural Science Foundation of China (Nos.61702347 and 62027801), the Natural Science Foundation of Hebei Province (Nos.F2022210007 and F2017210161), the Science and Technology Project of Hebei Education Department (Nos.ZD2022100 and QN2017132), and the Central Guidance on Local Science and Technology Development Fund (No.226Z0501G).

** E-mail: zhangyunzuo888@sina.com

still many defects. On the one hand, these methods require a large amount of data to pre-train the model, leading to they cannot achieve an unbiased estimation of data laws in application scenarios with limited data. On the other hand, models based on deep learning have high computational complexity and are difficult to apply on scenes with limited computing power.

Aiming at the problems of large amounts of calculation and high computational complexity of existing methods, ZHANG et al[20] proposed a progressive spatio-temporal tunnel (STT) to generate spatio-temporal action proposals. This method only processes pixels on the sampling line rather than processing the full amount of video spatial data to reduce calculation. However, in the real video scene, the movement of objects is uncertain and random. Since the progressive STT has to manually determine the number and position of the sampling lines, it cannot consider the computing speed and accuracy for video with changing motion conditions. To address this issue, we demonstrate the proposals from binary tree and STT for temporal segmentation of rough videos, which dynamically adjusts the sampling lines by generating a binary decision tree. The adaptive sampling of videos in different motion modes forms an adaptive STT, which improves the accuracy while reducing the computational complexity. The proposed method only processes pixels on the sampling line instead of all pixels, which greatly reduces the amount of computation. Simultaneously, the proposed method does not require a large amount of data to pre-train the model, which further reduces the computational complexity.

Here, the temporal segmentation of rough videos is produced by generating adaptive STT. Fig.2 shows the overview of the proposed method. First, we divide the input video into sub-videos of equal length and calculate the cumulative spatio-temporal flow to generate the flow overflow of sub-videos. Second, we calculate the balance factor according to flow overflow, and combine the binary tree to produce the decision tree. Finally, the sampling line is adjusted dynamically with a decision tree to generate an adaptive spatio-temporal tube for action proposals.

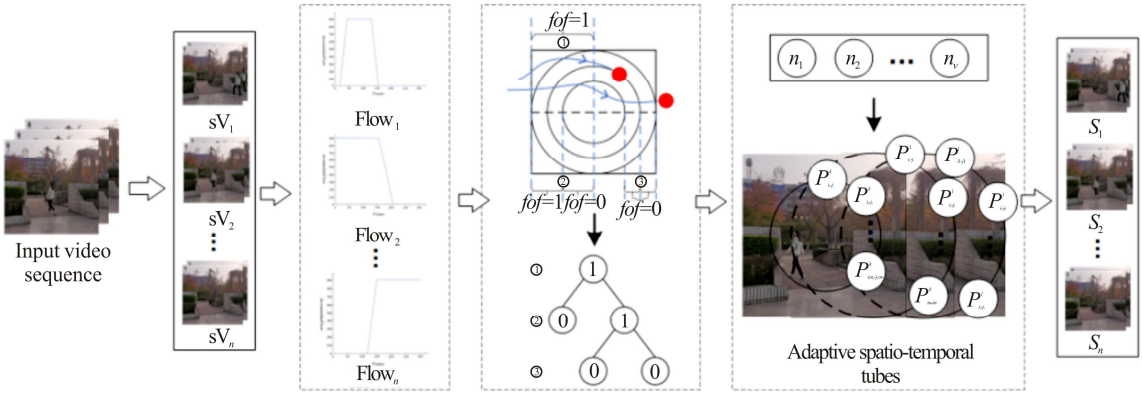


Fig.2 Overview of the proposed method

STT is formed by capturing motion objects with circular sampling lines in video frames according to ratios of video sequences. We use STT because it only needs to process the pixels on sampling lines of video to construct the STT flow model[20]. The STT method can greatly reduce data processing.

The input video sequence is divided into sub-videos of equal length to adjust dynamically sample lines. For a long rough video with N frames, there are total $n=\lceil N/V \rceil$ sub-videos, where V is the length of each sub-video. For motion analysis, the model extracts pixels on each circular sampling line to form STT. Suppose the perimeter of a sample line is L, the coordinates of pixel (x, y) on the STT are given as follows

$$\begin{cases} x = centerX - R \times \sin(\dfrac{2\pi S}{L}) \\ y = centerY + R \times \cos(\dfrac{2\pi S}{L}) \end{cases}, \tag{1}$$

where $(centerX, centerY)$ represents the center of circular sampling line and the range of S is S=1, 2,..., L.

STT introduces sub-sampling lines to analyze the motion direction of objects in the video. The pixel value of the object entering the sampling area is assigned as $I(x, y)$=1, and the pixel value of the object exiting the sampling area is assigned as $I(x, y)$=−1 to calculate the cumulative spatio-temporal flow CF as

$$CF(f_n) = \sum_{i=1}^{n} F(f_i), \tag{2}$$

$$F(f_i) = \mathrm{sum}(I(x, y)), I \in \Omega, \tag{3}$$

where Ω is the area with motion object.

As shown in Fig.3, the object enters the sampling area from behind the door and moves along the trajectory marked by the dotted line to the right boundary. The circular sampling line of progressive STT[20] only captures the objects exiting the sampling area. The spatio-temporal flow calculated by progressive STT is always less than 0. A similar situation includes that the object enters from the sampling area boundary and moves to the door for exiting the sampling area.

Fig.3 A video with spatio-temporal flow less than 0

In the real world, people browse videos for people, cars, animals and other objects with a certain volume when looking for motion objects in videos. These kinds of objects occupy multiple pixels in video rather than one[21]. Therefore, for a video sequence with the size of $W\times H\times T$, we define the tolerance for motion segmentation as

$$\delta=\frac{3c}{p}, \tag{4}$$

where p is the frame rate of the video sequence, W and H are the width and height of the video frame respectively, T is the length of the video sequence, and c=min(W, H).

We suppose that the length of the sub-video is N, and the width and height of the video frame are W and H, respectively. First, the circular sample line inscribed in the sub-video frame is used as the initial sample line. The initial sampling parameter of the initial sampling line is r_1=min{W/2, H/2}, l=1, where r is the radius of the circular sampling line, and l is the number of circular sampling lines. We calculate the initial cumulative spatio-temporal flow of the sub-video according to Eq.(2), and calculate the flow overflow *fof* as

$$fof=\begin{cases}1 & CF(f_{\text{end}})>\delta\\ 0 & \left|CF(f_{\text{end}})\right|<\delta\\ -1 & CF(f_{\text{end}})<-\delta\end{cases}, \tag{5}$$

where f_{end} is the end frame of the current sub-video, CF is the initial cumulative spatio-temporal flow, and δ is the tolerance.

In this letter, flow overflow with l=1 is regarded as the root node to generate a decision tree. Simultaneously, the parameter of sub-video i=1 is set. For the sub-video with *fof*=1, it means that the number of motion objects that are captured by the sampling line entering the sampling area is more than exiting. For the sub-video with *fof*=0, it means that the sampling line can integrally capture motion objects. For the sub-video with *fof*=−1, it means that the number of motion objects that are captured by the sampling line entering the sampling area is less than exiting. We set the balance factor *BF* for each node based on the above analysis as

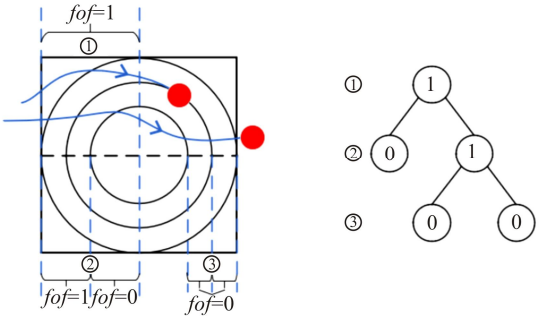
$$BF=\begin{cases}T & fof=0\\ F & \left|fof\right|=1\end{cases}. \tag{6}$$

A binary tree is made of nodes, where each node contains a left pointer, a right pointer, and a data element. The root pointer points to the top node in the binary tree. The left and right pointers recursively point to subtrees on either side, and the left and right subtrees are also binary trees. In this letter, the rough video is divided into small sub-videos. The decision tree is generated to adaptively sample, and dynamically find the optimal sampling position of every sub-video.

The decision tree is generated by combining STT and the binary tree. The Ω_{l-1} and Ω_l are two sampling areas formed by sampling with sampling parameter q_l. We analyze the *BF* of the two sampling areas. There is no need to adjust the sampling line with *BF*=*T* since the sampling strategy is optimal. We generate the leaf node of the decision tree with *BF*=*F* according to the value of flow overflow. The calculation of the newly generated leaf node is shown as

$$i_l=\begin{cases}2i_{l-1} & \boldsymbol{C}(\Omega_{l-1})=F\\ 2i_{l-1}+1 & \boldsymbol{C}(\Omega_l)=F\end{cases}, \tag{7}$$

where i_l=2i_{l-1} indicates that the sampling line is added on the inside of sampling area in the process of adjusting the sampling line. Simultaneously, a left subtree is added to the leaf node of decision tree. As shown in Fig.4, the red points represent motion objects, the blue lines represent motion trajectory, and the blue arrows represent the motion direction of objects. The figure briefly shows the process of dynamically adjusting sampling line and generating decision tree with *fof*=1. Similarly, i_l=2i_{l-1}+1 indicates sampling line is added on the outside of sampling area. A right subtree is added to the leaf node of decision tree. Fig.5 briefly shows the process of dynamically adjusting sampling lines and generating the decision tree with *fof*=−1.



(a) (b)

Fig.4 The process of generating a decision tree and dynamically adjusting sampling lines with flow overflow of 1: (a) Sampling lines; (b) Decision tree

The sampling parameters of the (l+1)th sampling line are calculated as follows

$$d=\lfloor\log_2 i\rfloor+1, \tag{8}$$

$$j=i-(2^{d-1}-1), \tag{9}$$

$$q_l = \frac{2j-1}{2^{d-1}} \times q_{l-1}, \tag{10}$$

where d is the depth of the decision tree, and j is the serial number of the node at the dth layer.

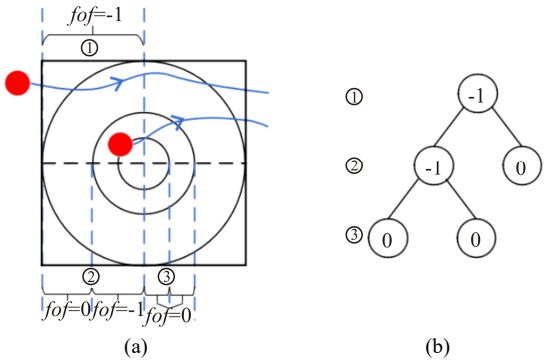


Fig.5 The process of generating a decision tree and dynamically adjusting sampling lines with flow overflow of −1: (a) Sampling lines; (b) Decision tree

We calculate the cumulative spatio-temporal flow of the newly generated sampling area. Meanwhile, the decision tree is generated until each leaf node of the decision tree of the sub-video meets $BF=T$.

The frame number of the sub-video $V = \{x_k\}_{k=1}^{T/n}$ that has completed adaptive sampling is [1, T/n], where $x_k \in R^{w \times h \times 3}$, and w, h, and 3 are the sizes of width, height, and channel number for each frame, respectively. The pixels on the sampling lines are extracted to construct an adaptive STT. The cumulative spatio-temporal flow is calculated and recorded according to Eq.(2). $Y = \{y_k\}_{k=1}^{T/n}$, $y_t \in \{0,1\}$ is the kth binary label which represents whether the kth frame is selected as a temporal action proposal. y_k is calculated as

$$y_k = \begin{cases} 1 & CF(f_k) > 0 \\ 0 & \text{else} \end{cases}. \tag{11}$$

The temporal action proposals S_v segmented from the sub-video segment is selected as follows

$$S_v = \{x_k \mid k \in E\}, \tag{12}$$

where $E = \{k \in [1, T/n] \mid y_k = 1\}$. The temporal action proposals of each calculated sub-video are spliced and output to generate the action proposals of the rough video.

We choose VISOR[22] and self-collected dataset for experiments. VISOR is a dataset of long red-green-blue (RGB) video recordings of people doing prescribed actions. It consists of rough videos covering a wide range of topics, including scenarios of streets, gates and crossroads. This dataset provides a ground truth consisting of temporal markers and action labels. The self-collected dataset is captured in the real world, including multiple real scenarios and kinds of motion states. It has a large variation in object size and intensity contrast. The method is implemented on an Intel Core i5-8265U CPU and 16.00 GB memory, with the software of Matlab R2018b.

To analyze the effectiveness of the proposed method, the calculating time (Tc), the precision rate (P), the recall rate (R) and F_1-*score* (F_1) are adopted to measure the performance of the method. All frames identified as action proposals are counted. Frames containing motion objects are defined as true positives (N_{TP}), and frames without motion objects are defined as false positives (N_{FP}). Frames that contain motion objects but are not identified as action proposals are defined as false negatives (N_{FN}). The P, R and F_1 are calculated as

$$P = \frac{N_{TP}}{N_{TP} + N_{FP}}, \tag{13}$$

$$R = \frac{N_{TP}}{N_{TP} + N_{FN}}, \tag{14}$$

$$F_1 = \frac{2 \times P \times R}{P + R}. \tag{15}$$

The performance of the proposed method is measured using Tc, P, R and F_1 as evaluation indicators for 10 videos. The proposed method is compared with the methods in Refs.[5], [7], [8] and [20]. Tab.1 shows the average calculating time (aTc), average precision rate (aP), average recall rate (aR) and average F_1-*score* (aF_1) of the comparison methods and the proposed method on 10 videos.

Tab.1 Comparison of experimental results

Method	aTc (s)	aP (%)	aR (%)	aF_1 (%)
Ref.[5]	931.07	83.68	80.46	82.04
Ref.[7]	478.91	90.13	80.79	85.20
Ref.[8]	683.24	88.07	87.19	87.63
Ref.[20] (1C)	34.16	64.93	60.32	62.54
Ref.[20] (2C)	76.30	63.67	67.14	65.36
Ref.[20] (3C)	137.87	60.12	74.93	66.71
Proposed method	**58.31**	**95.55**	82.64	**88.63**

In Tab.1, it can be seen that the average calculating speed of the proposed method is much higher than that of the methods in Refs.[5], [7], [8] and [20] (2C, 3C). Although the average calculating speed of the proposed method is slightly lower than that of the method in Ref.[20] (1C), the aP, aR and aF_1 of the proposed method have obvious advantages. We suppose the size of input video is $W \times H \times T$, where $W \times H$ is the size of video frame, and T is the length of the video sequence. The computational complexity of the proposed method is $O(T/n \times 2\pi \times (nR + \sum_{i=1}^{n} R_i))$, the computational complexity of the method in Ref.[20] is $O(T \times 2\pi \times (R + R/2 + R/3 + ...))$, and the computational complexity of the methods in Refs.[5], [7] and [8] is $O(W \times H \times T)$. The computational complexity of the proposed method is significantly superior to the comparison methods, which greatly reduces the calculating time.

As we can see from Tab.1, the results of the action proposals generated by the proposed method and the method in Ref.[20] are similar to videos with $BF=T$. Nevertheless, the method in Ref.[20] cannot completely

capture the motion trajectory for the video with *BF*=*F*. The proposed method constructs adaptive STT by dynamically adjusting the sampling line, which is suitable for the scene with variable motion. The *aR* of the proposed method is slightly lower than that of the method in Ref.[7], because the proposed method only calculates the pixels on the sampling line rather than full spatial data. Considering aF_1 and *aTc* comprehensively, the performance of the proposed method is better than that of the comparison method, and the calculating speed is greatly improved.

A great temporal action proposals method should achieve high *P* and *R* by finding motion frames as many as possible in a video[10]. In Tab.2, we summarize the comparison results of our method with Ref.[20] (2C) that have the closest calculation amount for both the VISOR and the self-collected datasets. Our method achieves a good *aR* of 82.83% for VISOR as compared to the method in Ref.[20] (2C). Similarly, Tab.2 also shows the method in Ref.[20] (2C) produces lower *P* since the method in Ref.[20] (2C) produces more false positive proposals. For self-collected, the proposed method achieves the *aR* of 82.44%, while the method in Ref.[20] (2C) produces the *aR* of 56.74%. We observe a better performance of our method against the Ref.[20] (2C) for different videos.

Tab.2 Temporal action segmentation results for VISOR and self-collected datasets

Video name	*P* (%)	*R* (%)	*P* (%)	*R* (%)
	Proposed method		Ref.[20] (2C)	
	VISOR[22]			
Video 1	96.28	89.07	94.91	85.92
Video 2	88.21	82.42	98.53	80.71
Video 3	100	74.83	100	79.44
Video 4	100	86.37	32.09	54.73
Video 5	93.52	81.47	27.47	39.01
Average	95.60	82.83	70.6	67.85
	Self-collected dataset			
Video 6	100	91.22	100	90.61
Video 7	91.85	80.29	94.52	89.37
Video 8	86.48	86.40	21.03	63.97
Video 9	99.18	75.81	40.07	33.62
Video 10	100	78.47	22.37	54.58
Average	95.50	82.44	56.74	66.43

To verify the correctness of the proposed method, two representative videos Video 4 and Video 8 are selected for analysis. Fig.6 shows the results of action proposals using the proposed method on two videos respectively. And we compare action proposals with the ground truth of videos. The horizontal direction in the figure represents the direction of the time axis, the black blocks represent the ground truth in the experimental videos, and the red blocks represent the action proposed by the proposed method.

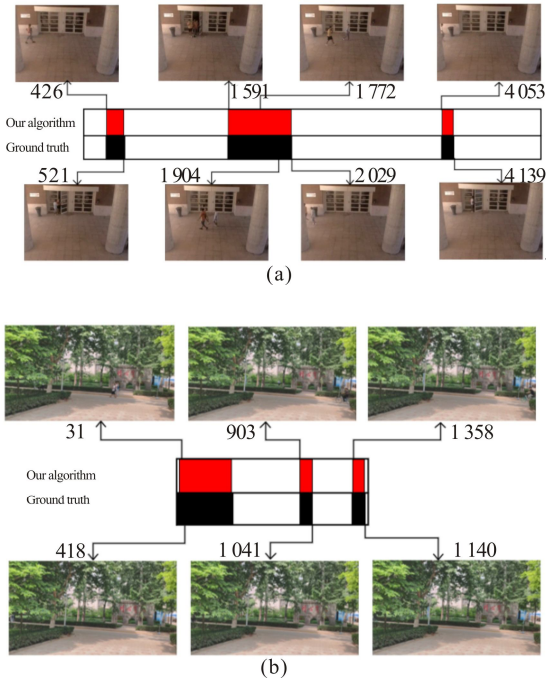


Fig.6 Final segmentation results where the ground truth is shown in black color, and segmentation results of our method are shown in red color for (a) VISOR and (b) self-collected datasets

As can be seen from Fig.6(a), the 426th, 1 591st and 4 053rd frames are the start frames of the action proposals, and there are motion objects in the corresponding video frames that enter the sampling area. The 521st, 2 029th and 4 139th frames are the end frames of the action proposals, and there are motion objects in the corresponding video frames that exit the sampling area. We randomly extract the frames of the action proposals using our method. There are motion objects in the 1 772nd and 1 904th frames. In Fig.6(b), there are motion objects entering in the 31st, 903rd and 1 358th frames, which are the start frames of the action proposals. In the 418th, 1 041st and 1 140th frames, there are motion objects that exit the sampling area. These frames are the end frames of the action proposals. The above analysis confirms the correctness of the proposed method.

In this letter, we have proposed a combining binary tree and STT for temporal segmentation of rough videos method by constructing decision tree. The method produces action proposals, which can directly segment the uncut videos with action segments. We calculate the spatio-temporal flow to generate flow overflow. Next, we create balance factor for every sub-video to construct decision tree. Finally, the adaptive STT is produced by adjusting dynamically sampling line to segment action proposals. From the experimental comparison and analysis, the proposed method solves the problem of high computational complexity of the existing methods. Our method achieves high robustness and is suitable for scenes with variable motion.

Statements and Declarations

The authors declare that there are no conflicts of interest related to this article.

References

[1] PENG J L, ZHAO Y L, WANG L M. Research on video abnormal behavior detection based on deep learning[J]. Laser & optoelectronics progress, 2021, 58(06)：51-61.

[2] ZHANG Z, NIE Y, SUN H, et al. Multi-view video synopsis via simultaneous object-shifting and view-switching optimization[J]. IEEE transactions on image processing, 2020, 29：971-985.

[3] LI T Y, BING B, WU X X. Boundary discrimination and proposal evaluation for temporal action proposal generation[J]. Multimedia tools and applications, 2021, 80(02)：2123-2139.

[4] AN P, LIANG J X, MA J. LiDAR-camera-system-based 3D object detection with proposal selection and grid attention pooling[J]. Applied optics, 2022, 61(11)：2998-3007.

[5] MURTAZA F, YOUSAF M H, VELASTIN S A. PMHI：proposals from motion history images for temporal segmentation of long uncut videos[J]. IEEE signal processing letters, 2018, 25(02)：179-183.

[6] QU J J, XIN Y H. Combined continuous frame difference with background difference method for moving object detection[J]. Acta photonica sinica, 2014, 43(07)：219-226.

[7] NAWAZ M, YAN H. Saliency detection using deep features and affinity-based robust background subtraction[J]. IEEE transactions on multimedia, 2021, 23(01)：2902-2916.

[8] GUO F, WANG W G, SHEN Z Y, et al. Motion-aware rapid video saliency detection[J]. IEEE transactions on circuits and systems for video technology, 2020, 30(12)：4887-4898.

[9] CONG R, LEI J, FU H, et al. Video saliency detection via sparsity-based reconstruction and propagation[J]. IEEE transactions on image processing, 2019, 28(10)：4819-4831.

[10] HEILBRON F C, NIEBLES J C, GHANEM B. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos[C]//Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 27-30, 2016, Las Vegas, NV, USA. New York：IEEE, 2016：1914-1923.

[11] WU Q, QUO H, WU X, et al. Fast action localization based on spatio-temporal path search[C]//Proceeding of 2017 IEEE International Conference on Image Processing (ICIP), September 18-20, 2017, Beijing, China. New York：IEEE, 2017：3350-3354.

[12] QIU J, WANG L, WANG Y, et al. Efficient proposals：scale estimation for object proposals in pedestrian detection tasks[J]. IEEE signal processing letters, 2020, 27(01)：855-859.

[13] PENG W, SHI J, ZHAO G. Spatial temporal graph deconvolutional network for skeleton-based human sction recognition[J]. IEEE signal processing letters, 2021, 28(01)：244-248.

[14] KUEHNE H, RICHARD A, GALL J. A hybrid RNN-HMM approach for weakly supervised temporal action segmentation[J]. IEEE transactions on pattern analysis and machine intelligence, 2020, 42(04)：765-779.

[15] LIU Z, WAQAS M, YANG J, et al. A multi-task CNN for maritime target detection[J]. IEEE signal processing letters, 2021, 28(01)：434-438.

[16] YU G, YUAN J. Fast action proposals for human action detection and search[C]//Proceeding of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 8-10, 2015, Boston, USA. New York：IEEE, 2015：1302-1311.

[17] CHEN K, WANG J, YANG S, et al. Optimizing video object detection via a scale-time lattice[C]//Proceeding of 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 18-21, 2018, Salt Lake City, UT, USA. New York：IEEE, 2018：7814-7823.

[18] SHEN Z, LIU Z, LI J, et al. Object detection from scratch with deep supervision[J]. IEEE transactions on pattern analysis and machine intelligence, 2020, 42(02)：398-412.

[19] QU S, ZHANG H, WU W, et al. Symmetric pyramid attention convolutional neural network for moving object detection[J]. Signal, image and video processing, 2021, 15(08)：1747-1755.

[20] ZHANG Y Z, LI W X, YANG P L. Surveillance video motion segmentation based on the progressive spatio-temporal tunnel flow model[J]. Electronics letters, 2021, 57(13)：505-507.

[21] ZHUANG X T. Research on deep learning networks for small object detection based on multi-level feature fusion[D]. Nanjing：Nanjing University of Posts and Telecommunications, 2021.

[22] VEZZANI R, CUCCHIARA R. Video surveillance online repository (VISOR)：an integrated framework[J]. Multimedia tools and applications, 2010, 50(01)：359-380.