Global Two-Stream Network for Temporal Action Proposal Generation

Zhou Lei 1,a, Junzhou Wang 1,* , shenbo Chen 1,b, Dawei Niu2,c

1School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China
2zhejiang Uniview Technologies co., Led

*aemail: leiz@shu.edu.cn, bemail: schen@shu.edu.cn, cemail: niudawei@uniview.com

*Corresponding author’s e-mail: whosjackie@shu.edu.cn

Abstract. Temporal action detection is a practical but challenging task. The current temporal action detection task has major shortcomings in the accuracy of proposal generation. The extraction of long-range temporal information features and the fusion of two-stream features for the task of temporal action proposal generation are still a challenge that needs to be improved. In this paper, we propose the Global Two-Stream Network, which innovatively introduces the Non-Local operation to extract the global background information from the features of the generated candidate proposals. And the backbone network of two-streams is used for better utilization of two-stream features to generate temporal action proposal segments with precise bounds and high confidence.

1. Introduction

Computer vision is a very important application area of artificial intelligence that has not been out of the limelight. Its essence is the use of computers to analyze, process, and understand the content of an image or video. Computers are now more sophisticated in some aspects of image processing than in video processing [1,2], for example: image classification, object detection, etc. However, the information that images can provide is static and the amount of information is less compared to the dynamic temporal information provided by video. Nowadays with the wide availability of various video devices such as cameras, the amount of video data has now increased significantly. Therefore people are now also starting to focus on the application of computers in video processing, and quality algorithms for processing video content have become a hot research topic. There are objects in the video, as well as various behaviors and activities, so the key to understanding the video content is how to identify the objects or behaviors in the video. One of the fundamental tasks in the field of video processing is video action recognition. This task is to extract and classify features from human segmented and cropped video clips by machine learning and other methods, and finally output the category corresponding to the clip. Most of today's deep learning driven action recognition algorithms exist as video classification tasks. Attempts have been made to crop the raw natural videos at manual level so that each video in the dataset contains only specific actions without information such as context. However, a large number of natural videos in daily life are often uncropped and contain a lot of background information, so it is infeasible and labor-intensive to use manual cropping for each recognition. Thus, the temporal action recognition algorithm is born.
The difference between the tasks of temporal action detection and action recognition is similar to the relationship between image classification and object detection in image processing. The methods are also broadly classified into single-stage methods [3] and two-stage methods [4,5,8]. The two-stage approach is to first generate temporal action proposals and then classify the temporal action proposals. Despite the high accuracy of current classification tasks, there is still a very large performance shortcoming in terms of accuracy of detection [4,5]. Therefore much attention has been paid to the proposal generation [4,5,8,11]. In order to generate high quality proposals, many proposal generation methods in recent years use sliding windows with different temporal duration for candidate proposal generation, and then use classifiers to classify these candidate proposals [6]. Another is to evaluate them at the top of the frame level, and finally combine them into candidate proposals, and then perform candidate proposal evaluation [5]. Although both approaches work well, they both have the disadvantage of having fewer global features and not fully utilizing RGB features and optical flow features. In the video action detection task, the two stream model structure proves to be effective [18,19,20] because the two-stream model extracts not only the original RGB graphic information but also incorporates a range of optical flow information that can represent motion information. Motion information is very important in video, and temporal action proposal generation is also a class of video tasks, so this paper is inspired by the study of K. Simonyan et al [19]. to use the two-stream structure for the task of temporal action proposal generation. As for the extraction of global information, we introduce the study of Wang et al. [13]. The extracted global information is integrated into the features. Therefore, in this paper, we propose Global Two-Stream Network, which will use non-local operations [13] and a designed two-stream backbone network to solve this problem, together with a bottom-up initial proposal generation method and a proposal localization method to design the overall framework.

2. Related Work

2.1. Temporal Action Proposal Generation

Temporal action proposal generation is a relatively new video understanding task that localizes actions from natural video without manual cropping, i.e., marking the start and end times of an action segment to obtain candidate proposals. It is similar to the proposal generation phase of object detection in image understanding [5]. And the general methods of temporal action proposal generation is mainly divided into traditional machine learning based algorithms [9,11] and deep learning based algorithms [4,5,6,10].

The current mainstream temporal action proposal generation methods are deep learning based methods, which are generally divided into bottom-up methods [5,8,10] and top-down methods [4,6,7]. Top-down approaches are based on manually defined or time-distributed anchor frames or sliding window methods to generate initial proposals, and then use binary classifiers and regression to evaluate confidence and adjust the bounds on these segments. However, using sliding windows fixes the size of the generated proposals at the beginning, while real scenario video segments have flexible boundaries and potentially longer duration, and these methods cannot be adjusted at a finer granularity when performing binary classification or boundary regression [4]. The bottom-up method was developed to solve this problem. Among them, a paper [10] proposed Temporal Actionness Grouping (TAG), a method for finer-grained evaluation at the frame level. Which uses a trained binary classifier to discriminate each video snippet. These small snippets are then aggregated by setting thresholds to gradually form candidate proposals. Later on, a variant of this method emerged [5,8] where a confidence evaluation is performed for each temporal position to get a confidence score of whether each temporal position (exactly to the frame) contains an action, and then some matching rules are set to perform a temporal position stitching on these positions to get the initial proposal. It can be seen that by forming candidate proposals from fine-grained aggregation or splicing, it is possible to form candidate proposal results with more flexible boundaries. Although this method does improve the accuracy of the proposal boundaries, the confidence is computed at local positions and there is a problem that only local information is extracted without obtaining a global information due to the local nature of the computation of the convolutional layer. Temporal proposal generation can reduce the labor
consumption of cropping video. It can be used in real-life scenarios such as video hotspot extraction and intelligent surveillance, and can also be applied to the temporal proposal generation part of two-stage temporal behavior detection.

2.2. Non-local Operation
For image content understanding and other aspects, convolutional neural networks have been important feature extraction tools, but the processing of convolutional networks determines that they can only extract features from the local part of the object at a time, which also causes the problem of difficulty in obtaining a global information. Non-local network\(^{[13]}\) was inspired from the Non-local means approach \(^{[12]}\). A more general method applied to computer vision is obtained. Regardless of the distance, the Non-local operation compensates to some extent for the local nature of the convolution operation by computing the relationship between two locations, thus obtaining a long range of dependencies and thus capturing the features of the background information. The current method of temporal action proposal generation is still insufficient for the extraction of global information, so the Non-local operation is introduced inspired by the Non-local network architecture, and the reason is to obtain a higher quality proposal fragment by refining the background information of the features. It is known that the Non-local module has been used for video classification, object detection and instance segmentation experiments before this, but it has not been applied to this aspect of temporal action proposal generation research.

3. Methods and Materials
This section will explain the model structure of the paper, including feature extraction network, backbone network, proposal generation module, confidence evaluation module, and finally post-processing module. The overall framework of the network is shown in Figure 1. The structure of the non-local module in the global two-stream module is shown in Figure 2.

![Figure 1](image)

Figure 1 The model architecture of this paper, including video representation network, global two-stream module, confidence evaluation module, proposal generation module and post-processing module.

3.1. Video Representation
According to the initial feature processing method proposed by the temporal action recognition in recent years \(^{[5,8]}\), for an untrimmed video containing \(l_o\) frames, denote it as \(F = \{f_i\}_{i=1}^{l_o}\), \(l_o\) is the total number...
of RGB frames in the video, \( f_i \) represents the RGB image of the \( i \)-th frame in the video sequence. Each untrimmed video also has a label, which is expressed as \( \Psi_g = \{q_{g,j} = (t_{s,j}, t_{e,j})\}_{j=1}^{N_g} \). Where \( N_g \) represents the number of actual temporal action segment annotations contained in the video, and \( t_{s,j}, t_{e,j} \) represents the start and end time of the marked temporal action segment in the video.

### 3.2. Feature Encoding

The untrimmed long video \( F = \{f_i\}_{i=1}^{l_v} \) is sampled at a fixed interval \( \alpha \) into video segments \( \{s_t\}_{t=1}^{l_s} \). Where \( f_i \) is the image frame with the temporal position \( i \), \( l_v \) is the total number of frames in the video, \( l_s = \frac{l_v}{\alpha} \), where \( s_t \) is the divided video segment, a segment contains an RGB frame and a stack of optical flow frames obtained with the RGB frame as the center, and \( l_x \) is the total number of divided video clips. These fragments were extracted through the spatial network and optical flow network\(^{[18]}\), and the RGB feature \( R_t \) and the optical flow feature \( G_t \) were obtained respectively. Then the feature of each segment can be expressed as a two-stream feature sequence \( \{R_t, G_t\}_{t=1}^{l_s} \) and \( l_f \) is set to a fixed length \( L \), which is convenient for processing and reduce computational consumption. The obtained feature sequence \( \{R_t, G_t\}_{t=1}^{l_s} \) is used in the processing of the following modules.

### 3.3. Global Two-Stream Module

In this section, global two-stream module is introduced to extract the global information features. The purpose is to extract features for the following two modules. In order to better process the two-stream information, this paper does not directly send it to the base module after processing through the two-stream network like the BSN method\(^{[5]}\), but divides the obtained two-stream features into RGB features and optical flow features separately, then process them parallelly, at last performs feature fusion. Research shows that the effect of such a processing method\(^{[7]}\) is better than that of fusion processing in advance. The two stream structures of the global two-stream module are similar, both of which are one-dimensional convolution module and non-local operation structure. They are expressed as

\[
S = \text{Conv1d}_{12} \left( \text{Non1d}_1 \left( \text{Conv1d}_{11}(R) \right) \right)
\]

\[
T = \text{Conv1d}_{22} \left( \text{Non1d}_2 \left( \text{Conv1d}_{21}(G) \right) \right)
\]

respectively, where \( \text{Non1d}_x \) represents the non-local layer of path \( X \). After processing the features of the two stream respectively, the features of the two stream are added at each position to get the final feature.

A non-local layer added to the network provides information about the association between each location. Especially for data such as video, it is particularly important to obtain large range of information rather than local information. Its formula is as follows:

\[
y_i = \frac{1}{C(x)} \sum_{y_j} f(x_i, x_j) g(x_j)
\]

In the above formula, \( x \) is the input, \( y \) is the output, and \( i, j \) represent the spatial position in the image. \( f \) is a function to calculate the similarity relationship between any two positions, and \( g \) is a mapping function:

\[
g(x_i) = W_g x_j
\]

Where, \( W \) is the parameter to be learned. This operation is to obtain a weight matrix by calculating the similarity of any two points in the space position, obtain a global information, and finally assign global information to each position in the space, which can solve the problem of feature extraction without global information mentioned above. It should be noted that in non-local operation, \( f \) exists in various forms, such as Gaussian, Embedded Gaussian, etc. In this paper, the form of Embedded Gaussian is selected:

\[
f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)}
\]
Figure 2 Model structure of Non-local module: The module used in this paper is a one-dimensional Non-local module, so the rectangular square in the figure represents a one-dimensional convolutional layer, and 1×100 represents a convolutional layer with a channel number of 100 and a convolutional kernel size of 1.

In the framework of this paper, a convolutional layer is used to perform the specific operation. As shown in Figure 2, the mappings $\theta$, $\phi$ and $g$ in the above equation correspond to three one-dimensional convolution operations, and the channels are half of the number of channels of the input features. Where the $\otimes$, $\oplus$ represents matrix multiplication and element-level addition, respectively. The final one-dimensional convolution restores the channels of the output features to the number of channels of the original input features. Since the Non-local module of this paper is chosen in the form of Embedded Gaussian, the Softmax operation is used here. It can be seen that after mapping through the two convolutional layers, the features are matrix multiplied, and then the relationship between the output temporal position and each temporal position of the feature is calculated, and thus the global position information is extracted, which to a certain extent compensates for the drawback caused by the local nature of the convolutional operation.

3.4. Proposal Generation Module
The task of the proposal generation module is to evaluate the confidence score at the beginning and end of each temporal position. Since untrimmed videos may contain action durations with different temporal lengths and span large distances, a fixed-size sliding window approach may lead to inflexible boundaries for proposals that are not flexible enough to accommodate real situations with large temporal differences, and approaches to build pyramids by multi-scale resolution features [22] or to build proposals at the frame level [5,8] have emerged as a solution. This paper is inspired by the approach of constructing proposals through frame level. The global two-stream information features extracted through base module are put into the proposal generation module, and the output is two score sequences of 1×T shape. Where $T$ is the temporal length of the video feature sequence. The module consists of two parts, each of which has the same structure and is

$$p = \text{Sigmoid} \left( \text{Conv1d}_{s2} \left( \text{Conv1d}_{s1}(f) \right) \right)$$

The end of the module outputs the confidence of the beginning and end of each temporal location respectively. In the inference stage, after obtaining the confidence of the beginning and end of each temporal location, we need to generate candidate proposals based on these scores. When selecting a starting position, for example, for each temporal position $t_n$, score $p_{str}^n$, one of two conditions is met: 1) $p_{str}^n > 0.5 \times \max(p)$ 2) $p_{str}^n = \max(p)$, put it into the candidate sequences $G_s$, for the end of the sequence we created in accordance with the law. Finally, the temporal location in the two sequences that
meet the requirements are joined together. It should be noted that the beginning position should be less than the end position, and then we can get the candidate proposals \( \sigma = \{ s_t, t_e, p_{s_t}, p_{s_e}, p_{c_t}, p_{c_e} \} \), where \( p_{p_e}, p_{p_r} \) are the confidence scores of the corresponding proposal categories and regression generated in the next section.

3.5. Confidence Evaluation Module

The purpose of this module is to evaluate the confidence for each qualifying temporal location. In order to generate the confidence map, we were inspired by the BM layer generation mechanism in the BMN model \([8]\). Its purpose is to transform the two-dimensional \( C \times T \) global two-stream feature into a feature map of \( C \times S \times M \times T \) size that can be input into the proposed evaluation module. Where \( S \) is the number of samples, and \( M \) is the maximum duration length proposed in the video. So how to implement such a process? In order to achieve parallel computation and reduce the overall training time of the model, the weighted map is obtained in advance, and then the final required feature map is obtained by dot product the weighted map with the global two-stream feature. The weight map samples evenly on the temporal according to each predefined proposal. If the temporal location is not divisible, for example, the temporal location is \( t_n \) and the position weight of \( \text{floor}(t_n) \) is reset to \( 1 - \text{decimal}(t_n) \), the position weight of \( \text{floor}(t_n) + 1 \) is reset to \( \text{decimal}(t_n) \), and the rest of the positions are set to 0. Such a weight matrix is generated for each proposal in this video, and finally the \( S \times T \times M \times T \) sampling mask is obtained. The feature map is obtained after dot product with the two-dimensional global feature.

It is then fed into a series of convolutional layers for processing. Firstly, a one-dimensional convolution layer is sent to extract features at the temporal sequence position. Then, a three-dimensional convolution layer is used to conduct convolution operation in the dimension of \( S \) and reduce its dimension to 1. Then, four two-dimensional convolution layers are followed to reduce dimensions at first and then extract and aggregate the feature information of adjacent proposals. Finally, a sigmoid layer is attached to obtain two confidence maps with the size of \( M \times T \). Each position on the map is a proposed binary classification score \( p_{p_e} \) and regression confidence score \( p_{p_r} \).

3.6. The Training of The Network

This section talks about label generation and the choice of loss function. For the proposal generation module, we need to generate label for each start and end temporal location. In order to facilitate calculation, we extend the start and end positions of the label value by a certain percentage, for example: \( b_{g_{x}} = \{ t_{s} - g/12, t_{s} + g/12 \} \), where \( g = t_{s} - t_{e} \). Then correspondingly for each temporal position \( t \), we also extend it to \( b_{p} = \{ t_{i} - d/3, t_{i} + d/3 \}, d = (t_{i+1} - t_{i}) \). Then we calculate IoR of each \( b_{p} \) with \( b_{g_{x}}, b_{g_{e}} \). IoR is defined as the ratio of the coverage of temporal position and true value to the duration of this region. We use the cross entropy loss function:

\[
L_{PGM} = L_{c_{11}} + L_{c_{12}}
\]  

For the confidence evaluation module, a label with the same size as the confidence map needs to be generated. For each position on the map, we calculate the intersection ratio with all the labels \( \delta_{g} = \{ t_{s}, t_{e} \} \), and find the largest IoU as the label of that position. In this module we use two loss functions, cross entropy loss function and L2 loss, and integrated into a multi-objective loss function:

\[
L_{CEM} = L_{c} + \omega L_{R}
\]

Finally, add the loss functions of each module and add the L2 regularization term to get our training target.

3.7. Post-processing

In the evaluation process, after the operation of the proposal generation module and the confidence evaluation module, we get the candidate proposal \( \sigma = \{ t_{s}, t_{e}, p_{p_{p_{e}}}, p_{p_{r}}, p_{c_{e}}, p_{c_{r}} \} \). So many scores are difficult to unify the standard, so we follow the method of BSN \([5]\) to carry out a fusion of scores:

\[
p_{\text{final}} = p_{p_{p_{e}}} \cdot p_{p_{r}} \cdot \sqrt{p_{c_{e}} \cdot p_{c_{r}}}
\]
In the post-processing step, since there are many overlapping and redundant proposals in the process of generating proposals, here we only need to cover the maximum proposals with ground truth, so we need to suppress some of the proposals to ensure that the retrieved proposals are the optimal ones as much as possible. So that the best recall is achieved with as few offers as possible at a high tIOU. This algorithm was previously often used in the task of object detection [1]. Here Soft-NMS is used [14], compared to the normal NMS, this improved method does not directly discard the proposals with high overlap with the highest scoring proposal, but sets a weight, the more overlap with the highest scoring proposal the more it suppresses the score of that box, thus compensating for the effect problem caused by the missed and false detection of NMS due to poor threshold setting. Through this method, we get the final proposal result \( \tau_i = \{ t_a, t_e, p_{final} \}_{i=1}^N \), where \( N \) represents the number of final proposals.

4. Experiments

The experimental environment used in this paper is NVIDIA RTX3090, Intel Core i9-10900K CPU, 32GB DDR4. Operating system: 64bit Ubuntu 20.04.1. Software environment: Python3.8 and PyTorch 1.7.1. The parameter of the experiment is batch size=16, epoch number=11, learning rate=0.001.

4.1. Dataset

ActivityNet-1.3 dataset is used in this paper, which is the largest dataset in the task of temporal action recognition and temporal action proposal generation. The dataset contains more than 700 hours of video covering more than 200 categories of action, accounting for about 50% of the training set, and about 25% of the validation and testing sets.

4.2. Evaluation Metrics

For the temporal action proposal generation method, there is no need for classification. The mainstream evaluation method, Average Recall vs. Average Number of Proposals per Video curves (AR-AN), is used to evaluate the performance of the model. In this case, we refer to proposals that are greater than or equal to a certain threshold value, which refers to the tIOU for calculating each proposal and the true value, namely the temporal IOU. Average Recall represents the sum of proposed recalls of all videos divided by the total number of videos. The Average Number represents the Average Number of proposals found per video, divided by the total Number of proposals submitted by the total Number of videos in the dataset. Take AN as the horizontal axis, start from 0, step size is 1, the maximum is 100; Take AR as the vertical axis, draw the curve, and find the area under the curve.

4.3. Comparisons with State-of-the-Art

We compared the model performance in this paper with Lin et al. [15], CTAP [17], MGG [16] and BMN [8], and the comparison results are shown in Table 1

| Method          | T.Lin et al. | CTAP | BSN  | MGG  | BMN  | Ours  |
|-----------------|--------------|------|------|------|------|-------|
| AR@100          | 73.01        | 73.17| 74.16| 74.54| 75.01| 75.75 |
| AUC             | 64.40        | 65.72| 66.17| 66.43| 67.10| 67.71 |

It can be seen from the data in Table 1 that the performance of the model proposed in this paper is significantly improved compared with all the other models compared. Because the non-local module in this paper makes up for the convolutional layer’s limitation of only extracting local features, it enhances the model’s extraction of video features with a long temporal duration, and overcomes the previous model’s insufficient extraction of background global information to a certain extend. In addition, the
enhanced ability of two-stream network to extract two-stream information makes the performance of the model in this paper significantly improved.

And we evaluated the curves of AR-AN under different tIOU thresholds. This is shown in Figure 3.

![Figure 3 Plot of AR-AN at each threshold for the model in this paper.](image)

5. Conclusion

This paper describes a new method of generating temporal action proposal. The method adopts the non-local module structure, which is optimized and enhanced in extracting the global information. In addition, a network designed specifically for extracting the two-stream structure is used to extract the information of the two-stream feature. The resulting proposal from this model has a) flexible boundary; 2) precise boundary; 3) reliable confidence. In fact, I think more research needs to be done on the extraction of global context information in the direction of temporal action proposal generation.

Acknowledgments

The authors would like to thank the Heigh Performance Computing Center of Shanghai University for providing the computing resources and technical support.

References

[1] Ren, S., He, K., Girshick, R.B., & Sun, J. (2015) Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39: 1137-1149.

[2] Girshick, R. (2015) Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV). Santiago. pp. 1440-1448.

[3] Yeung, S., Russakovsky, O., Mori, G., Li, F. F. (2016) End-to-End Learning of Action Detection from Frame Glimpses in Videos. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA. pp. 2678-2687.

[4] Gao, J., Yang, Z., Sun, C., Chen, K., and Nevatia, R. (2017) Turn tap: Temporal unit regression network for temporal action proposals. In: 2017 IEEE International Conference on Computer Vision (ICCV). Venice, Italy. pp. 3648–3656.

[5] Lin, T., Zhao, X., Haisheng, S. (2018) Bsn: Boundary sensitive network for temporal action proposal generation. In: European Conference on Computer Vision, 2018. Munich, Germany. pp. 3-21

[6] Shou, Z., Wang, D., , Chang, S.-F. (2016) Temporal action localization in untrimmed videos via multi-stage cnns. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA. pp. 1049–1058.
[7] Chao, Y., Vijayanarasimhan, S., Sudheendra, S., Seybold, B., Bryan, R., Ross, D. A., Deng, J., Sukthankar, R. (2018) Rethinking the faster r-cnn architecture for temporal action localization. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Salt Lake City, UT, USA. pp. 1130-1139.

[8] Lin, T., Liu, X., Li, X., Ding, E., Wen, S. (2019) Bmn: Boundary-matching network for temporal action proposal generation. In: The IEEE International Conference on Computer Vision (ICCV), 2019. Seoul, Korea (South). pp. 3888-3897.

[9] Wang, H., Kläser, A., Schmid, C., and Liu, C.-L. (2011). Action recognition by dense trajectories. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Providence, RI. pp. 3169-3176.

[10] Xiong, Y., Zhao, Y., Wang, L., Lin, D., Tang, X. (2017) A pursuit of temporal accuracy in general activity detection. http://arxiv.org/abs/1703.02716.

[11] Heilbron, F. C., Niebles, J. C., Ghanem, B. (2016) Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2016). Las Vegas, NV, USA. pp. 1914-1923.

[12] Buades, A., Coll, B., Morel, J.-M. (2005) A non-local algorithm for image denoising. In: Computer Vision and Pattern Recognition (CVPR). San Diego, CA, USA. pp. 60-65.

[13] Wang, X., Girshick, R. B., Gupta, A., & He, K. (2018) Non-local Neural Networks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA. pp. 7794-7803.

[14] Bodla, N., Singh, B., Chellappa, R., Davis, L. S. (2017) Soft-nms improving object detection with one line of code. In: 2017 IEEE International Conference on Computer Vision (ICCV). Venice, Italy. Pp. 5562–5570.

[15] Lin, T., Zhao, X., and Shou, Z. (2017) Temporal convolution based action proposal: Submission to activitynet 2017. https://arxiv.org/abs/1707.06750

[16] Liu, Y.; Ma, L.; Zhang, Y.; Liu, W.; and Chang, S.-F. (2019) Multi-granularity generator for temporal action proposal. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA. pp. 3599-3608.

[17] Gao, J., Chen, K., Nevatia, R. (2018) Ctap: Complementary temporal action proposal generation. In: European Conference on Computer Vision (ECCV), Munich, Germany. pp. 68–83.

[18] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L. V. (2016) Temporal segment networks: Towards good practices for deep action recognition. In: European Conference on Computer Vision (ECCV), 2016. Amsterdam, The Netherlands. pp. 20-36.

[19] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., & Gool, L. J. (2019) Temporal Segment Networks for Action Recognition in Videos. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, 41: 2740-2755.

[20] Feichtenhofer, Christoph, A. Pinz and Andrew Zisserman. (2016) Convolutional Two-Stream Network Fusion for Video Action Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA. pp. 1933-1941.

[21] Carreira, J., & Zisserman, A. (2017) Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Hawaii. pp. 4724-4733.

[22] J. Yuan, B. Ni, X. Yang, and A. A. Kassim. (2016) Temporal action localization with pyramid of score distribution features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA. pp 3093–3102.