Augmented Transformer with Adaptive Graph for Temporal Action Proposal Generation

Shuning Chang[1,2]*, Pichao Wang[2]†, Fan Wang[2], Hao Li[2], Jiashi Feng[1],

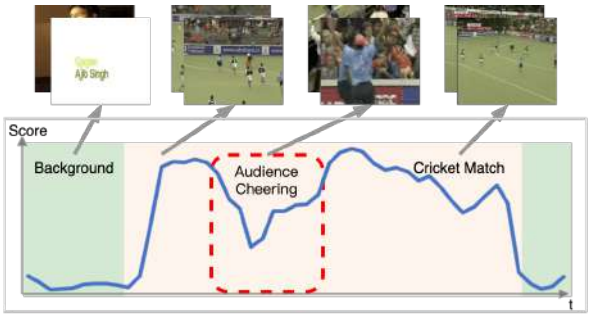
[1]National University of Singapore [2] Alibaba Group
changshuning@u.nus.edu, {pichao.wang,fan.w,lihao.lh}@alibaba-inc.com, elefjia@nus.edu

Abstract

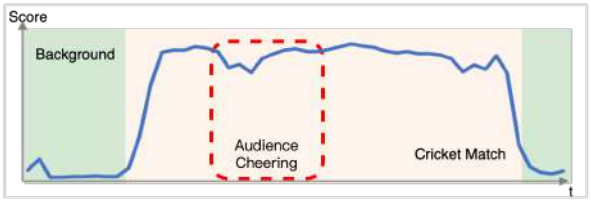
Temporal action proposal generation (TAPG) is a fundamental and challenging task in video understanding, especially in temporal action detection. Most previous works focus on capturing the local temporal context and can well locate simple action instances with clean frames and clear boundaries. However, they generally fail in complicated scenarios where interested actions involve irrelevant frames and background clutters, and the local temporal context becomes less effective. To deal with these problems, we present an augmented transformer with adaptive graph network (ATAG) to exploit both long-range and local temporal contexts for TAPG. Specifically, we enhance the vanilla transformer by equipping a snippet actionness loss and a front block, dubbed augmented transformer, and it improves the abilities of capturing long-range dependencies and learning robust feature for noisy action instances. Moreover, an adaptive graph convolutional network (GCN) is proposed to build local temporal context by mining the position information and difference between adjacent features. The features from the two modules carry rich semantic information of the video, and are fused for effective sequential proposal generation. Extensive experiments are conducted on two challenging datasets, THUMOS14 and ActivityNet1.3, and the results demonstrate that our method outperforms state-of-the-art TAPG methods. Our code will be released soon.

1. Introduction

Temporal action detection, which aims at locating specified actions from untrimmed videos in the temporal dimension, has been widely applied to various tasks such as video understanding [24, 29], surveillance [23], *etc*. Similar to object detection, the pipeline of temporal action detection is usually divided into two stages: temporal action proposal generation (TAPG) and action classification. The performance of such two-stage temporal action detectors is largely determined by the proposal quality from TAPG.



(a) BMN+Actionness predictor.



(b) Global context capture.

Figure 1: Benefit of using long-range dependencies. We apply BMN [16] and our method to detect the specified action *cricket match* in a video segment, which also includes an audience cheering segment highlighted in red dotted boxes. We plot the curves of snippet-wise actionness scores for predictions by BMN without using global context in (a) and our method using global context in (b). It can be seen the one with global context shows higher confidence scores for the frames of audience cheering, demonstrating the benefits brought by global context for temporal action proposal generation.

*Work done during an internship at Alibaba Group.
†Corresponding author.

Prior works focus on exploiting the local temporal context information [5, 16, 5, 33] for generating high quality proposals. For example, [5, 16] apply a pre-defined ratio to extend the temporal action boundaries, and use temporal convolutional layers or dilated convolutional layers to increase the receptive field; [33] leverages a GCN model to aggregate local temporal and semantic context. Using local context information can effectively build relations among neighbors of current snippets, which works well for locating simple action instances with clean frames and clear boundaries, but is sensitive to action/background representation. However, the real-world cases are often complicated, with many irrelevant frames and background clutters over the interested actions. Only exploiting the local temporal context lacks the semantic understanding for the whole video, often leading to failures of locating precise boundaries. As shown in Figure 1a, an actionness predictor is added into a recent method BMN [16] to show its performance on a complicated video. We can see that, if we only leverage local temporal context for detecting the specified action (i.e. *cricket match*), the long audience cheering segment embedded in frames of the specified action obviously gives low action confidence scores. But if a longer range is observed (in Figure 1b), the model is able to understand that the audience cheering is also part of the match, and thus treating this segment and the actions before and after as a whole can avoid wrong boundary prediction. From this example, we claim that long-range temporal dependencies should also be exploited for more comprehensive video understanding in complicated situations. In this work, we propose an augmented transformer with adaptive graph network (ATAG) to take advantage of the two types of contexts for effective temporal proposal generation. It includes an augmented transformer which mines long-range temporal context for noisy action instance localization, and an adaptive GCN which captures local temporal context.

Specifically, to locate noisy action instance, one way is to adequately understand the holistic video and regard the noisy frames as a part of action according to the semantics. The transformer is adopted to achieve this purpose, which has shown superior capability of capturing long-term dependencies [10, 11]. However, the vanilla transformer has two issues if directly applied in our task. First, the loss functions in the traditional TAPG methods only supervise proposal-level signal and cannot directly guide the transformer to aggregate long-range features to snippet-level features. Therefore, we apply a snippet actionness loss to binarily classify each snippet into action or background according to the snippet-level features output by the transformer. The snippets, especially for those noisy snippets, need to extract helpful information from other snippets to be correctly classified, thus the snippet actionness loss explicitly forces the transformer to selectively learn long-range dependencies. On the other hand, in the vanilla transformer, when generating the queries and keys, the receptive field is constraint to the snippet itself and it is not a robust feature learning style for these noisy snippets. To deal with this problem, we introduce a front block on top of vanilla transformer. The front block is a convolution-based lightweight network, which expands the temporal receptive field and filters out noisy frame features.

For the local temporal context capture, we focus on exploiting the position and gradient/difference information between adjacent features. We propose an adaptive GCN to build the local temporal context, where two adjacency matrices are carefully designed. One matrix has all the elements generated during training. It is similar to conventional convolutional layers but different positions correspond to different kernels, which represents the common pattern for all the training data and position information. It can adaptively determine whether the farther snippets should have smaller weights than the central one, or whether the snippets near the edge of the video should be assigned with smaller weights. The other is a content-based adjacency matrix, which captures the difference between node pairs and represents the unique pattern for each data. By combining the above two adjacency matrices, our data-driven graph increases the flexibility for graph construction and brings more generality to build local context relationships in various samples.

We demonstrate the effectiveness of ATAG on two challenging datasets, THUMOS14 and ActivityNet13. Our model outperforms well established state-of-the-art methods significantly.

2. Related work

Attention mechanism for long-range dependencies. Long-range attention mechanism is to compute the response at a position in a sequence by accessing all positions and taking their weighted average in the embedding space. It is broadly leveraged in natural language processing (NLP) and computer vision. Vaswani et al. [25] introduce a self-attention mechanism, called transformer, capturing long-range dependencies among words in one sentence to address the machine translation task. Following [25], many transformer-based models are proposed and show great potential to tackle various tasks. Devlin et al. [7] apply the bidirectional training of Transformer and successfully deal with a broad set of NLP tasks. Girdhar et al. [10] utilize transformer to aggregate features from the spatiotemporal context for recognizing and localizing human actions. Carion et al. [4] employ transformers to build an end-to-end object detection model. Besides transformers, there are some other manners to utilize the attention mechanism to capture long-range dependencies. For example, Wang et al. [30] embed non-local structure into the action recognition net-

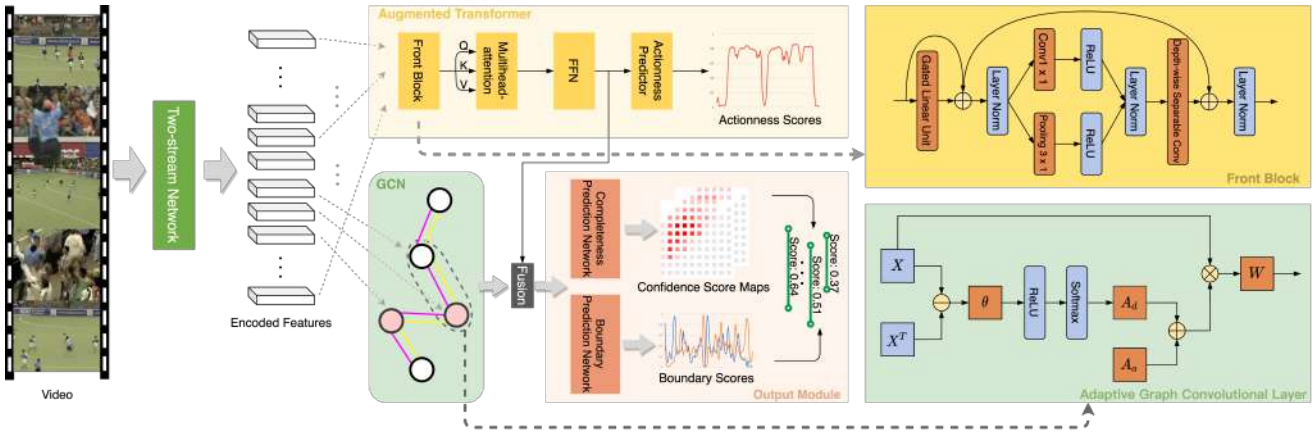


Figure 2: Illustration on the architecture of our method. It first employs a two-stream network to extract snippet-level encoded features. Then, it deploys an augmented transformer and an adaptive graph to capture global and local context, respectively. After fusing global and local features, an output module generates the desired predictions.

work to capture long-range dependencies. Wu et al. [31] introduce long-term feature banks to analyze videos. How to effectively exploit temporal context is a pivotal problem in video analysis and long-range dependencies has been applied widely [10, 30, 31, 14]. However, the long-range dependencies for TAPG has not been well explored, especially in the complicated noisy scenarios.

Temporal action proposal generation (TAPG). Early TAPG methods use temporal sliding window [19, 26] and snippet-wise probability [22, 35] to generate candidate proposals. The former fails to handle ground truth action instances with various durations, while the latter is sensitive to noise and can hardly handle actions with long duration. Recently, most methods adopt multi-scale anchors to generate proposals [5, 17, 18, 16, 33, 34, 1], similar with the idea in anchor-based object detection [20]. Chao et al. [5] adopt a multi-tower architecture to align the receptive field and extend proposal boundaries by a pre-defined ratio which is widely applied by later methods. In BSN [17], Lin et al. directly predict the probability of boundaries and cover flexible temporal duration of action instances. Based on BSN, Lin et al. [16] further propose a boundary-matching mechanism to evaluate the completeness of densely distributed proposals. The ideas of GCN have been adopted in some work, Xu et al. [33] extract temporal context and semantic context by GCN, but applies conventional temporal convolutional layers for actual implementation in temporal context extraction. Bai et al. [1] perform a GCN to build the relationship between the boundaries and the content. Therefore, they do not apply GCN to capture local temporal context.

Above methods have made significant progress in this field. However, they have limited capability of building local dependencies from the local context, leading to degraded performance for complicated cases. Our proposed ATAG has the ability to build both long-range and local temporal dependencies to achieve better video understanding.

3. Method

The overall architecture of the proposed augmented transformer with adaptive graph network (ATAG) is shown in Fig. 2. In this section, we first describe the problem formulation and backbone feature extraction in Sec. 3.1. The core augmented transformer with adaptive graph architecture is then introduced in Sec. 3.2, including augmented transformer and adaptive GCN as well, followed by the output module (Sec. 3.3) used to generate final prediction, and the training (Sec. 3.4) and inference (Sec. 3.5) stage.

3.1. Problem formulation

We denote an untrimmed video $\mathcal{X}$ as a frame sequence $\mathcal{X} = \{x_n\}_{n=1}^{l}$ with l frames, where x_n is its n-th RGB frame; and the temporal annotations for the N_g action instances included in $\mathcal{X}$ are denoted as $\psi_g = \{\varphi_n\}_{n=1}^{N_g}$ with $\varphi_n = (t_{s,n}, t_{e,n})$. Here $t_{s,n}$ and $t_{e,n}$ denote the start and end time index of the action instance φ_n respectively. The target of TAPG is to generate temporal proposals each covering a ground truth action instance as accurate as possible. Unlike the temporal action detection task, here the action instance categories need not be considered.

Following previous TAPG methods [16, 33, 1], we adopt a pre-trained encoding model to extract features of the input video $\mathcal{X}$, which serve as inputs to our developed TAPG model. In particular, for any given video $\mathcal{X}$, we split it into T continuous and non-overlapped snippets. We then adopt a two-stream network [28] for extracting T snippet features of C-dimension, which are then concatenated to form the

video features $F \in \mathbb{R}^{T\times C}$ and fed into the our model for proposal generation.

3.2. Augmented transformer with adaptive graph architecture.

ATAG introduces a dual-path module, consisting of an augmented transformer and adaptive GCN for global and local temporal context capture, respectively. The input feature F is split into two parts along the channel dimension, denoted as F^g and F^l, which are then fed into the two modules.

Augmented transformer. We apply transformers to improve the semantic representation of snippet-level features by capturing long-range dependencies. A vanilla transformer block is composed of a self-attention layer and position-wise feed-forward layers, performing multi-head attention to compute a weighted sum of input snippet features $F^g \in \mathbb{R}^{T\times C}$, resulting an augmented feature $\tilde{F}^g$ with global context. In particular, for each attention head, the attention map is computed by matching the transformed input features (a.k.a. the queries) $Q = f(F^g)$ to another transformation of the input features (a.k.a. the keys) $K = g(F^g)$, with f and g being learnable linear transformation. Namely,

$$A = Attention(F) = \mathrm{softmax}(\frac{QK^\top}{\sqrt{d}}), \qquad (1)$$

where $A \in \mathbb{R}^{T\times T}$ is the generated attention map, and d is the dimension for Q and K. The above attention map computation can find the relationship between features Q and K of global snippets and then aggregate their information into another linear transformation of input features $V = h(F^g)$ (a.k.a. the values). The multi-head attention is followed by an FFN layer, which contains two linear layers with ReLU activation and a residual connection after each layer. We also include layer normalization and dropout to facilitate training.

Directly employing the vanilla transformer presents two issues here. First, the main task of TAPG is the boundary regression, and the corresponding loss functions supervise proposal-level signal and cannot guide the transformer to effectively learn long-term semantic relations for each snippet. Second, according to Eq. 1, the computation of any element A_{mn} in the attention map matrix A depends on only the features of snippet m and n, namely, F^g_m and F^g_n, which indicates that the attention map generation does not consider any temporal context, especially in limit transformer layers. To solve the above issues, we add a snippet actionness loss and a front block to the vanilla transformer and it is dubbed augmented transformer.

The snippet actionness loss explicitly guides the transformer to learn effective long-range dependencies at snippet level. An actionness predictor is equipped upon the FFN, and it is used to predict the probability of an action instance existing in the input snippet, by minimizing the following binary classification loss L_a over action/background categories:

$$L_a = \sum_{i=1}^{T} \alpha^+ \cdot g_i^a \cdot \log p_i^a + \alpha^- \cdot (1-g_i^a) \cdot \log(1-p_i^a), \quad (2)$$

where $\alpha^+ = T/\sum(g_i^a)$, $\alpha^- = T/\sum(1-g_i^a)$, and p_i^a and g_i^a are the actionness predicted probability and action/background label of snippet i.

This loss function supervises the network to obtain snippet-wise classification only depending on each snippet feature from the output of transformer, which is critical to achieve noisy action instances localization. For those noisy snippets that have action labels, the network shall build correct relations among the noisy snippets and others to make correct decisions. Thus, this loss function helps the transformer learn how to selectively capture long-range dependencies and understand the complicated noisy action instances.

The front block is a light-weight network consisting of three parts to expand the temporal receptive field. We first apply a gated linear unit [6], followed by a parallel 1×1 convolutional layer and 3×1 average pooling layer with stride 1 to expand the receptive field, and the small 3×1 average pooling layer can also smooth the snippet-level feature to filter out the tiny noisy frames. The last part is a convolutional layer with a large kernel size, for example, 7×1. To avoid over-fitting from large size kernels, we adopt the depth-wise separable convolution. We apply residual connection for each part and layer normalization behind each part. The structure of front block is depicted in the top-right corner in Fig. 2.

Adaptive graph convolutional layer. To capture the local context, we design a new graph convolution layer to construct the local branch. We build a video graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{v_i\}_{i=1}^T$ and $\mathcal{E}$ represent the node and edge sets, respectively. Each node represents a snippet and each edge shows the dependency between two snippets. For local context modeling, edges between two nodes are constructed according to their temporal distance. The edge set is defined as:

$$\mathcal{E} = \{(v_i, v_{i+k}) | i \in \{1, ..., T\}; k \in \{0, \pm 1, ..., \pm\delta\}\}, \quad (3)$$

where T is the number of snippets, and δ is defined as the maximum connection distance. Our graph convolution operation $\mathcal{F}$ is defined as:

$$\mathcal{F}(F^l) = W \cdot F^l \cdot (A_a + A_d) \odot M, \qquad (4)$$

where $W \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ includes trainable weights, M is a fixed $T \times T$ binary mask matrix to limit the connection range according to δ, and $\odot$ is the dot product. Instead of using a pre-defined adjacency matrix, the matrix in Eq. 4 is divided into two parts: A_a and A_d and treated differently.

The first part (A_a) is completely parameterized and optimized in the training phase. The elements in A_a can be arbitrary values without any constraints, which means that the edges of the graph are completely learned based on the training data and the positions of nodes in the video.

The second part (A_d) is a data-dependent graph which adaptively learns a unique graph for each video. Our GCN is used to capture local temporal context, therefore we pay more attention to the difference of features. To determine whether there is a connection between two nodes and how strong the connection is, the difference of two nodes m and n is computed as:

$$A_{d,mn} = \frac{exp(\sigma(\theta^T|f_m - f_n|))}{\sum_{i=1}^{N} exp(\sigma(\theta^T|f_m - fi|))}, \tag{5}$$

where σ is an activation function, θ^T is the trainable parameter vector to reduce the dimension of $f_m - f_n$ to 1. The detailed visual description is shown in Figure 2.

Global-local fusion. Early fusion and late fusion are widely used in two-stream feature fusion. In our method, we use the early fusion, *i.e.*, the global and local features are fused by concatenation before the output module.

3.3. Output module

After obtaining global and local features, we feed them into our output module to generate the proposal boundaries and completeness scores, which are applied for proposal generation during inference.

Temporal boundary classification. Due to our global-local combination mechanism, the two-branch features have precise and discriminative action/background representation. Here, we input them into two same convolutional networks consisting of two temporal convolutional layers and a sigmoid activation function to generate the probability of the start p^s and end p^e for each snippet, respectively.

Completeness regression. Besides the prediction of boundaries, the network is trained to predict the completeness of proposals to boost the final results. Our completeness regression network follows [16] and also introduces the Boundary-Matching mechanism to generate completeness scores for densely distributed proposals. For each proposal, we sample 32 features from the corresponding two-branch features. Sampled features are then input into the completeness prediction network to generate two completeness score maps $M^{cc} \in \mathbb{R}^{T \times D}$ for completeness classification, and $M^{cr} \in \mathbb{R}^{T \times D}$ for completeness regression, where D is the maximum proposal duration and is set depending on the dataset.

3.4. Training

We apply weighted binary logistic regression loss function L_{bl} for start and end classification losses, denoted as L_s and L_e, where L_{bl} is denoted as:

$$L_{bl} = \sum_{i=1}^{T} (\alpha^+ \cdot g_i \cdot \log p_i + \alpha^- \cdot (1 - g_i) \cdot \log(1 - p_i)), \tag{6}$$

where $\alpha^+ = T/\sum(g_i)$, $\alpha^- = T/\sum(1 - g_i)$, and p_i and g_i are the predicted probability and ground truth of i-th snippet. Following BMN [16], our completeness loss L_{com} consists of two different loss functions, binary classification loss and regression loss:

$$L_{com} = L_c(G^c, M^{cc}) + \lambda L_r(G^c, M^{cr}), \tag{7}$$

where L_c is also a binary logistic regression loss function and its formula is as same as Eq. 6 and L_r is l_2 loss, and λ is set as 10.

The model is trained in the form of a multi-task loss function, with the overall loss function defined as:

$$L = L_{com} + \lambda_1 L_a + \lambda_2 L_s + \lambda_3 L_e, \tag{8}$$

where λ_1, λ_2, and λ_3 are three scalars to balance the three terms. The method of label assignment will be described in detail in the supplementary material.

3.5. Inference

During inference, a proposal set $\psi_p = \{\phi_n = (t_s, t_e, p^s_{t_s}, p^e_{t_e}, p^{cc}_{t_s,t_e}, p^{cr}_{t_s,t_e})\}_{n=1}^{N}$ is generated, where $p^s_{t_s}$, $p^e_{t_e}$ are start and end probabilities in t_s and t_e, and $p^{cc}_{t_s,t_e}$, $p^{cr}_{t_s,t_e}$ are completeness classification score and completeness regression score of proposal from $[t_s, t_e]$. To obtain final results, it is necessary to perform steps of score fusion and redundant proposal suppression.

Score fusion. In order to make full use of various predicted scores for each proposal ϕ_n, we fuse its boundary probabilities and completeness scores by multiplication. The confidence score p^f can be defined as:

$$p^f = p^s_{t_s} \cdot p^e_{t_e} \cdot p^{cc}_{t_s,t_e} \cdot p^{cr}_{t_s,t_e}. \tag{9}$$

Hence, the proposals set can be re-written as $\psi_p = \{\phi_n = (t_s, t_e, p^f)\}_{n=1}^{N}$.

Redundant proposals suppression. Because of our dense proposal generation, it is inevitable to output numerous redundant proposals which highly overlap with each other. Thus, based on the completeness score for proposals, we apply the Soft-NMS algorithm to remove redundant proposals. Candidate proposal set ψ_p turns to be $\psi'_p = \{\phi_n = (t_s, t_e, p^{f\prime})\}_{n=1}^{N'}$, where $p^{f\prime}$ and N' is the final completeness score and the number of final proposals, respectively.

4. Experiments

4.1. Datasets and setup

Datasets. The evaluation is performed on two popular TAPG benchmark datasets, THUMOS14 [12] and ActivityNet1.3 [3]. We use the subset of THUMOS14 that provides frame-wise action annotations. The model is trained with 200 untrimmed videos from its validation set and evaluated using 213 untrimmed videos from the test set. This dataset is challenging due to the large variations of the frequency and duration of action instances across videos. ActivityNet1.3 is a large-scale dataset covering 200 complicated human activity classes with 19,994 untrimmed videos, which are divided into training, validation and test set in the ratio 2:1:1. The model is trained on the training set and evaluated with the validation set.

Implementation details. For feature encoding, following previous works [16, 1], we adopt two-stream network [28] pre-trained on training set of Activity1.3. The frame interval is set to 5 and 16 on THUMOS14 and Activity1.3, respectively. For ActivityNet1.3, we resize video feature sequences by linear interpolation to 100, *i.e.* $T = 100$, and set the maximum duration length as 100. For THUMOS14, we slide the window on video feature sequence with 50% overlap and $T = 128$ and set the maximum duration length as 64, which can cover 98% action instances.

Adam [13] optimizer is adopted and the learning rate is set to 10^{-3} and decayed by a factor of 0.1 after every 10 epochs. To preserve the positional information in the transformer, we adopt sine positional encoding and add it to queries and keys. There are 8 heads in the multi-head attention module is 8. $\delta = 2$ in local branch. In Eq. 8, the coefficients λ_1, λ_2 and λ_3 are all set to 1. Early fusion is adopted unless otherwise specified.

4.2. Temporal action proposal generation

Temporal action proposal generation aims to produce high quality proposals that have high IoU with ground truth action instances and high recall. To verify proposal quality, Average Recall (AR) under multiple IoU thresholds are calculated. For ActivityNet1.3 and THUMOS14, the IoU thresholds used are $[0.5 : 0.05 : 0.95]$ and $[0.5 : 0.05 : 1.0]$, respectively. AR under different Average Number of proposals (AN) is defined as AR@AN, and the area under the AR vs. AN curve is named AUC. We use AR@AN and AUC as our metrics to evaluate our model.

Comparison with state-of-the-art methods. Our model is compared with state-of-the-art TAPG methods on THUMOS14 and ActivityNet1.3. Table 1 shows the comparison on THUMOS14. To ensure a fair comparison, both C3D and two-stream features used by previous methods are adopted in our experiments. Following previous methods [17, 16, 15, 1], we apply NMS and soft-NMS as post-processing methods to evaluate our method, respectively. For both C3D and two-stream features, our results outperform other state-of-the-art methods by a large margin. Additionally, NMS achieves better average recall performance than soft-NMS under small proposal numbers.

The results on ActivityNet1.3 are summarized in Table 2. Our method again surpasses all of the leading methods.

These experiments demonstrate the effectiveness of our model by adequately exploiting the global and local context.

Visualization and analysis. Fig. 4 visualizes some representative results on ActivityNet1.3 and THUMOS14 featuring different challenging cases. The proposals with the highest k scores are visualized in each video, where k is the number of ground truth. In the top video, background frames and action frames have the similar scene and there are many irrelevant frames which are nearly the same as background frames in the action. However, our top-1 proposal still successfully aligns the position of ground truth action instance. Our network is trained to regard the noisy snippets in action instance as action. We concern that whether the network will overfit on the noisy frames and many background snippets are mistakenly treated as action instance. In the middle video, a cheering segment exists between the true action and background so it needs to be classified as background, which our model predicts correctly. Note that this is different from the case in Fig. 1, where the cheering segment is embedded in an action instance. The correct prediction made by our model shows that it correctly understands the semantic information and does not overfit on the noisy frames. The bottom video has multiple ground truth action instances, while our top-5 proposals perfectly cover them in an accurate way, suggesting the high quality of our generated proposals. More examples and comparison with the previous method are shown in the supplement material.

4.3. Ablation study

To verify the effectiveness of our method, we conduct the following ablation studies on the validation set of ActivityNet1.3 dataset.

Feature	Methods	AR@AN @50	@100	@200	@500	@1000
C3D	SCNN-prop [22]	17.22	26.17	37.01	51.57	58.20
	SST [2]	19.90	28.36	37.90	51.58	60.27
	TURN [9]	19.63	27.96	38.34	53.52	60.75
	BSN [17]+NMS	27.19	35.38	43.61	53.77	59.50
	BSN [17]+SNMS	29.58	37.38	45.55	54.67	59.48
	MGG [18]	29.11	36.31	44.32	54.95	60.98
	BMN [16]+NMS	29.04	37.72	46.79	56.07	60.96
	BMN [16]+SNMS	32.73	40.68	47.86	56.42	60.44
	DBG [15]+NMS	32.55	41.07	48.83	57.58	59.55
	DBG [15]+SNMS	30.55	38.82	46.56	56.42	62.17
	BC-GNN+NMS [1]	33.56	41.20	48.23	56.54	59.76
	BC-GNN+SNMS [1]	33.31	40.93	48.15	56.62	60.41
	ATAG (Ours)+NMS	**34.91**	**42.34**	48.64	57.96	62.03
	ATAG (Ours)+SNMS	34.47	41.92	**49.60**	**58.49**	**62.24**
2stream	TAG [35]	18.55	29.00	39.61	-	-
	TURN [9]	21.86	31.89	43.02	57.63	64.17
	CTAP [8]	32.49	42.61	51.97	-	-
	BSN [17]+NMS	35.41	43.55	52.23	61.35	65.10
	BSN [17]+SNMS	37.46	46.06	53.21	60.64	64.52
	BMN [16]+NMS	37.15	46.75	54.84	62.19	65.22
	BMN [16]+SNMS	39.36	47.72	54.70	62.07	65.49
	MGG [18]	39.93	47.75	54.65	61.36	64.06
	DBG [15]+NMS	40.89	49.24	55.76	61.43	61.95
	DBG [15]+SNMS	37.32	46.67	54.50	62.21	66.40
	BC-GNN+NMS [1]	41.15	50.35	56.23	61.45	66.00
	BC-GNN+SNMS [1]	40.50	49.60	56.33	62.80	66.57
	ATAG (Ours)+NMS	**43.60**	**52.21**	**59.67**	65.98	69.24
	ATAG (Ours)+SNMS	43.52	51.86	59.48	**66.04**	**70.28**

Table 1: Comparison of our model with state-of-the-art methods on THUMOS14 testing set. All the results are reported in percentage. SNMS stands for Soft-NMS.

Method	[17]	[16]	[18]	[1]	[34]	ours
AR@100(val)	74.16	75.01	74.54	76.73	75.27	**76.75**
AUC(val)	66.17	67.10	66.43	68.05	66.51	**68.50**
AUC(test)	66.26	67.19	66.47	-	-	**68.45**

Table 2: Comparison of our model with state-of-the-art methods BSN [17], BMN [16], MGG [18], BC-GCN [1], BUMR [34] on ActivityNet1.3. All the results are reported in percentage.

Component analysis. The augmented transformer and adaptive GCN are the core components in our method. We evaluate several variants of this model by ablating its augmented transformer and GCN in Table 3. *Base* represents the version without augmented transformer and adaptive GCN. We can see that any component can individually improve the performance. The existence of snippet actionness loss and front block can boost the effect of vanilla transformer, which demonstrates that they can assist the transformer to learn long-range dependencies. For local context capture, the result shows that adaptive graph is beneficial for this task and removing any adjacency matrix will degrade the performance. With the global and local fusion, the full model reaches the best result.

Figure 3: Visualization examples of generated proposals on ActivityNet1.3 and THUMOS14. The red boxes highlight some frames in the action instances.

Early fusion *vs*. late fusion. We explore other fusion strategies. Specifically, in the early fusion, besides our early fusion with concatenation, we also evaluate the global and local features are fused by summation; in the late fusion, each path is individually fed into independent prediction networks of the output module and concatenate the dual-branch features till entering classifiers to aggregates predictions of boundaries and completeness scores. We show the results in Table 3. The Early fusion with concatenation outperforms other fusion strategies.

Number of transformer layers. Stacking multi-layer transformers normally brings better performance, which is demonstrated in other tasks, for instance, NLP [25, 7] and object detection [4]. We stack different numbers of transformers and show their results in Table 4. Different from other tasks, more transformers in our model do not improve the performance. We believe the reason is that the scale of datasets in this task is too small to train a deeper transformer network. Although there are about 20k videos in ActivityNet1.3, the density of action instances is low and each video includes only 1.5 action instances on average, which is far less than the datasets in other tasks.

Our GCN *vs*. other schemes. For the local context capture, we use a GCN with a hybrid adaptive adjacency matrix. Further, we experiment with different structures to replace our GCN, including (i) a general GCN with predefined adjacency matrix, (ii) a self-attention style GCN where the adjacency matrix A is calculated by $A = softmax(X^T W_\theta^T W_\phi X)$ where X is the input feature sequence and W_θ and W_ϕ are trainable weights, and (iii) a

Variants of model	AR@100	AUC
Base	75.10	66.80
Base + VT	75.42	67.48
Base + VT + Snippet actionness loss	75.76	67.74
Base + VT + Front block	75.85	67.78
Base + Augmented transformer	76.19	68.01
Base + GCN wo/A_a	75.50	67.31
Base + GCN wo/A_d	75.58	67.54
Base + GCN	75.79	67.81
Late fusion	76.26	68.25
Early fusion - summation	75.99	67.88
Early fusion - concatenation	76.75	68.50

Table 3: Performance evaluation on different components of our model and comparison of different fusion strategies. VT is short for vanilla transformer.

No. of Transformer	AR@100	AUC
1	76.75	68.50
2	76.64	68.35
3	75.94	67.83

Table 4: Performance evaluation on different number of transformer layers.

Structure	AR@100	AUC
General GCN	75.28	67.24
Self-attention GCN	74.76	66.50
Convolutional layer	75.20	67.20
Our GCN	75.79	67.81

Table 5: Comparison of our GCN with other schemes.

convolutional layer. Table 5 compares all these variants, with our choice outperforming other two variations.

4.4. Temporal action detection with our proposals

The important usage of action proposals is for temporal action detection. Thus, evaluating the performance of proposals in the temporal action detection task is another aspect to verify the quality of proposals. We adopt the evaluation metric Mean Average Precision (mAP) from the temporal action detection task. mAP with IoU thresholds $[0.5 : 0.05 : 0.95]$ are used on ActivityNet1.3 and mAP with IoU thresholds $\{0.3, 0.4, 0.5, 0.6, 0.7\}$ are used on THUMOS14.

We adopt the two-stage "detection by classifying proposals" temporal action detection framework. On ActivityNet1.3, the proposals are ranked according to their confidence scores and top 100 proposals per video are selected. Then, for each video, its top-1 video-level classification result will be obtained from [36] and all the proposals of this video share this classification result as their action classes.

Method	0.5	0.75	0.95	Average
CDC [21]	43.83	25.88	0.21	22.77
SSN [32]	39.12	23.48	5.49	23.98
BSN [17]+[36]	46.45	29.96	8.02	30.03
BMN [16]+[36]	50.07	34.78	8.29	33.85
GTAD [33]+[36]	50.36	34.60	9.02	34.09
BC-GNN [1]+[36]	50.56	34.75	9.37	34.26
ATAG (Ours)+[36]	**50.92**	**35.35**	**9.71**	**34.68**

Table 6: Action detection results on validation set of ActivityNet1.3 dataset in terms of mAP at IoU 0.5, 0.75 and 0.95, and average mAP.

Method	0.7	0.6	0.5	0.4	0.3
TURN [9]	6.3	14.1	24.5	35.3	46.3
BSN [17]	20.0	28.4	36.9	45.0	53.5
MGG [18]	21.3	29.5	37.4	46.8	53.9
BMN [16]	20.5	29.7	38.8	47.4	56.0
GTAD [33]	23.4	30.8	40.2	47.6	54.5
BC-GNN [1]	23.1	31.2	40.4	49.1	57.1
ATAG (Ours)	**28.0**	**38.0**	**47.3**	**53.1**	**62.0**

Table 7: Comparison between our approach and other temporal action detection methods on THUMOS-14.

On THUMOS14, we adopt top-2 video-level classification results generated by UntrimmedNet [27] as labels for top 200 proposals per video.

The results on ActivityNet1.3 and THUMOS14 are shown in Table 6 and Table 7, respectively. Our method achieves new state-of-the-art results on both datasets, which validates the quality of our proposals and the effectiveness of our method. The high mAP reflects that our model can predict the good proposals with high scores, and reduce the number of false positives.

5. Conclusion

In this paper, we propose an augmented transformer with adaptive graph network (ATAG) for temporal action proposal generation, which presents an augmented transformer and a new adaptive GCN to capture long-range temporal context and local temporal context, respectively. Our augmented transformer can effectively enhance the video understanding and noisy action instance localization. For local context capture, the newly designed adaptive GCN with two types of matrices is leveraged to build local temporal relationships. Extensive experiments show that our model achieves new state-of-the-art performance in temporal action proposal generation and action detection on THUMOS14 and ActivityNet1.3 datasets.

Acknowledgement

This work was supported by Alibaba Group through Alibaba Research Intern Program.

References

[1] Yueran Bai, Yingying Wang, Yunhai Tong, Yang Yang, Qiyue Liu, and Junhui Liu. Boundary content graph neural network for temporal action proposal generation. *arXiv preprint arXiv:2008.01432*, 2020.

[2] Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *CVPR*, 2017.

[3] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.

[5] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *CVPR*, 2018.

[6] Yann N Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In *International conference on machine learning*, pages 933–941. PMLR, 2017.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] Jiyang Gao, Kan Chen, and Ram Nevatia. Ctap: Complementary temporal action proposal generation. In *ECCV*, 2018.

[9] Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *ICCV*, 2017.

[10] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *CVPR*, 2019.

[11] Liang Han, Pichao Wang, Zhaozheng Yin, Fan Wang, and Hao Li. Exploiting better feature aggregation for video object detection. In *ACM MM*, 2020.

[12] Yu Gang Jiang, Jingen Liu, A Roshan Zamir, George Toderici, Ivan Laptev, Mubarak Shah, and Rahul Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014.

[13] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[14] Alexander Kozlov, Vadim Andronov, and Yana Gritsenko. Lightweight network architecture for real-time action recognition. In *SAC*, 2020.

[15] Chuming Lin, Jian Li, Yabiao Wang, Ying Tai, Donghao Luo, Zhipeng Cui, Chengjie Wang, Jilin Li, Feiyue Huang, and Rongrong Ji. Fast learning of temporal action proposal via dense boundary generator. In *AAAI*, 2020.

[16] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *ICCV*, 2019.

[17] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*, 2018.

[18] Yuan Liu, Lin Ma, Yifeng Zhang, Wei Liu, and Shih-Fu Chang. Multi-granularity generator for temporal action proposal. In *CVPR*, 2019.

[19] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. The lear submission at thumos 2014. 2014.

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[21] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. In *CVPR*, 2017.

[22] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016.

[23] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. In *CVPR*, 2018.

[24] Chen Sun, Sanketh Shetty, Rahul Sukthankar, and Ram Nevatia. Temporal localization of fine-grained actions in videos by domain transfer from web images. In *ACM MM*, 2015.

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.

[26] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge*, 2014.

[27] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, 2017.

[28] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016.

[29] Pichao Wang, Wanqing Li, Philip Ogunbona, Jun Wan, and Sergio Escalera. Rgb-d-based human motion recognition with deep learning: A survey. *CVIU*, 171:118–139, 2018.

[30] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018.

[31] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *CVPR*, 2019.

[32] Yuanjun Xiong, Yue Zhao, Limin Wang, Dahua Lin, and Xiaoou Tang. A pursuit of temporal accuracy in general activity detection. *arXiv preprint arXiv:1703.02716*, 2017.

[33] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *CVPR*, 2020.

[34] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *ECCV*, 2020.

[35] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *ICCV*, 2017.

[36] Y Zhao, B Zhang, Z Wu, S Yang, L Zhou, S Yan, L Wang, Y Xiong, D Lin, Y Qiao, et al. Cuhk & ethz & siat submission to activitynet challenge 2017. *arXiv preprint arXiv:1710.08011*, 8, 2017.

6. Supplementary

6.1. Label assignment

The methods of label assignment in each loss functions are described in detail below.

For boundary classification loss, we need to generate ground truth of boundary label including start g^s and end g^e. Given a ground truth action instance $\phi_n = (t_s, t_e)$, we denote its start and end region as $r_s = [t_s - 1.5\Delta t, t_s + 1.5\Delta t]$ and $r_e = [t_e - 1.5\Delta t, t_e + 1.5\Delta t]$ respectively, where Δt is the temporal interval between two adjacent snippets. Then, we compute overlap ratio IoR of each snippet interval with r_s and r_e separately, where IoR is defined as the overlap ratio with ground truth proportional. If one snippet interval is overlapped with multiple actions, we take the maximum IoR. Finally, we assign positive labels to the locations with IoR > 0.5; otherwise negative labels.

For actionness classification loss, we need to generate ground truth of actionness label g^a. Similar to boundary label generation, we calculate the IoR of each snippet interval with ground truth of action instances and use the same threshold 0.5 to assign a positive or negative label.

For proposal completeness loss, we need to generate proposal completeness label map G^c. For a proposal $\phi_{i,j} = (t_j, t_j + t_i)$, we compute its Intersection-over-Union (IoU) with all the ground truth action instances, and denote the maximum IoU as $G^c[i, j]$.

6.2. Additional visualization

Figure 4 illustrates more visualization examples and comparison with classical method BMN [16]. The proposals of our method and BMN[1] with the highest k scores are visualized in each video, where k is the number of ground truth. All the examples belong to complicated scenarios where interested actions involve irrelevant frames and background clutters. Traditional BMN fails to be robust for noisy frames and outputs incomplete proposal in the first four videos. In the last two videos, although its predictions cover the whole action instances, we speculate the reason is that the network overfits the noisy frames since those proposals also contain massive background frames. By comparison, our method presents precise boundary predictions in all the cases. Those examples adequately demonstrate that our method can understand video semantics and deal with complicated and noisy action instances by modeling long-range and local temporal context, by equipping our novel augmented transformer and adaptive GCN; in contrast, previous methods, such as BMN, focusing on the single local temporal context lack this capacity.

[1]The reproduction code we use is from `https://github.com/JJBOY/BMN-Boundary-Matching-Network`, whose performance is sightly higher than paper's.

layer	kernel size	stride	group	act	output size
Actionness Predictor					
conv1d	3	1	4	ReLU	$256\times T$
conv1d	1	1	1	Sigmoid	$1\times T$
Completeness Prediction Network					
BM layer	-	-	-	-	$256\times 32\times D\times T$
conv3d	(32,1,1)	(32,0,0)	1	ReLU	$512\times 1\times D\times T$
squeeze	-	-	-	-	$512\times D\times T$
conv2d	(1,1)	(1,1)	1	ReLU	$128\times D\times T$
conv2d	(3,3)	(1,1)	1	ReLU	$128\times D\times T$
conv2d	(3,3)	(1,1)	1	ReLU	$128\times D\times T$
conv2d	(1,1)	(1,1)	1	Sigmoid	$2\times D\times T$
Boundary Prediction Network					
conv1d	3	1	4	ReLU	$256\times T$
conv1d	1	1	1	Sigmoid	$2\times T$

Table 8: The detailed architectures of some modules in our method.T is the number of snippets and D represents the maximum duratin length.

6.3. Detailed architecture

Table 8 presents the architecture of our actionness predictor in augmented transformer and output module including completeness prediction network and boundary prediction network.

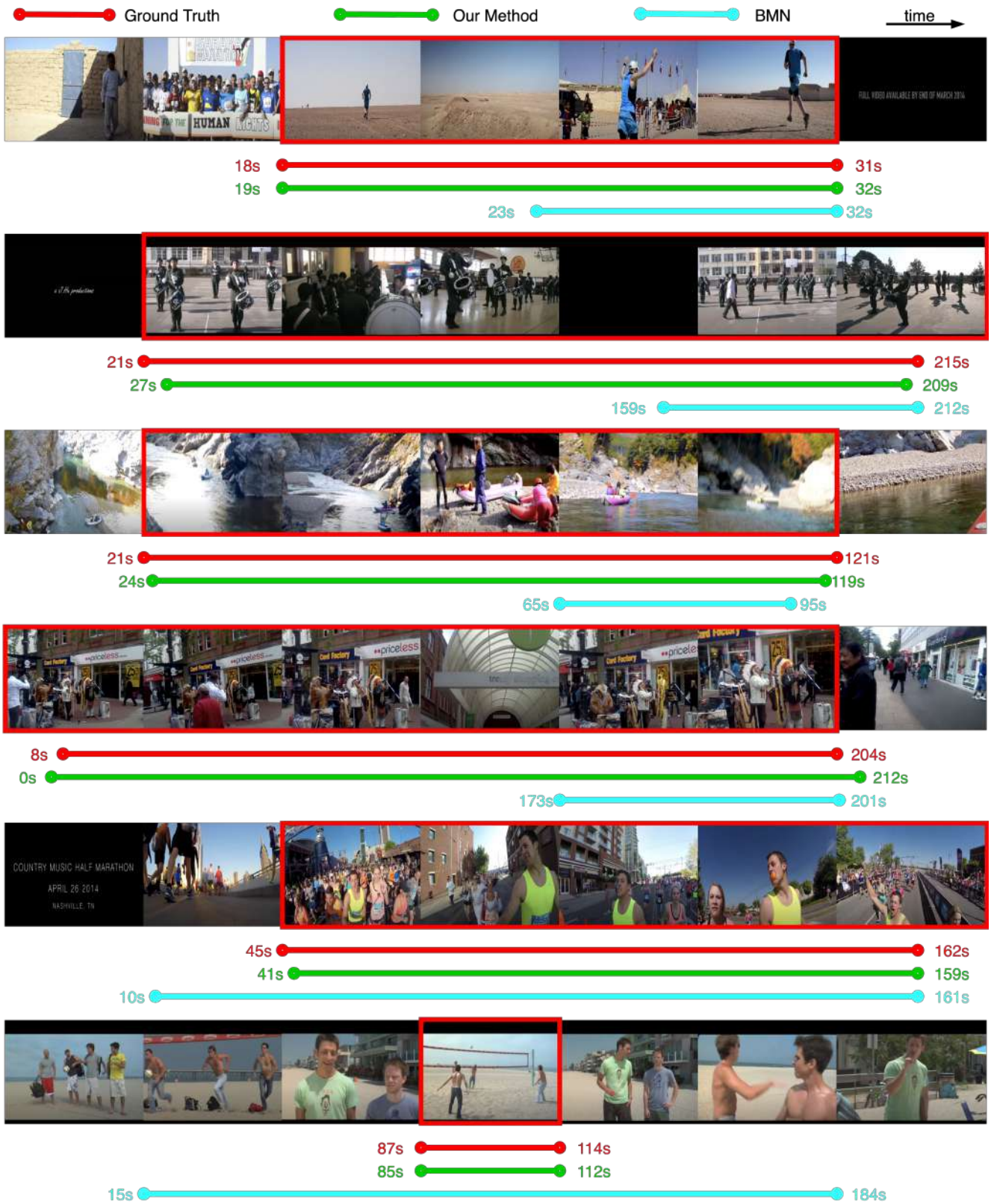


Figure 4: Visualization examples of generated proposals on ActivityNet1.3 and THUMOS14. The red boxes highlight some frames in the action instances.