Learning Salient Boundary Feature for Anchor-free Temporal Action Localization

Chuming Lin*[1], Chengming Xu*[2], Donghao Luo[1], Yabiao Wang[1], Ying Tai[1],
Chengjie Wang[1], Jilin Li[1], Feiyue Huang[1], Yanwei Fu[2]
[1]Youtu Lab, Tencent, [2]Fudan University, China
{chuminglin, michaelluo, caseywang, yingtai, jasoncjwang, jerolinli, garyhuang}@tencent.com
{cmxu18, yanweifu}@fudan.edu.cn

Abstract

Temporal action localization is an important yet challenging task in video understanding. Typically, such a task aims at inferring both the action category and localization of the start and end frame for each action instance in a long, untrimmed video. While most current models achieve good results by using pre-defined anchors and numerous actionness, such methods could be bothered with both large number of outputs and heavy tuning of locations and sizes corresponding to different anchors. Instead, anchor-free methods is lighter, getting rid of redundant hyper-parameters, but gains few attention. In this paper, we propose the first purely anchor-free temporal localization method, which is both efficient and effective. Our model includes (i) an end-to-end trainable basic predictor, (ii) a saliency-based refinement module to gather more valuable boundary features for each proposal with a novel boundary pooling, and (iii) several consistency constraints to make sure our model can find the accurate boundary given arbitrary proposals. Extensive experiments show that our method beats all anchor-based and actionness-guided methods with a remarkable margin on THUMOS14, achieving state-of-the-art results, and comparable ones on ActivityNet v1.3. Code is available at https://github.com/TencentYoutuResearch/ActionDetection-AFSD.

1. Introduction

Recently, with the progress of technology, a dramatically increasing number of videos have been stored and accessible from various daily activities. Temporal Action Localization (TAL), as a fundamental aspect of video understanding, thus plays an important role in real life, extending in several practical applications such as video analysis and summarization, human interaction, *etc*. Compared with action recognition that takes medium-range videos as input and only requires class labels as prediction, TAL is aimed at not only classifying every activity instance in each video, but also looking for the accurate temporal locations of them.

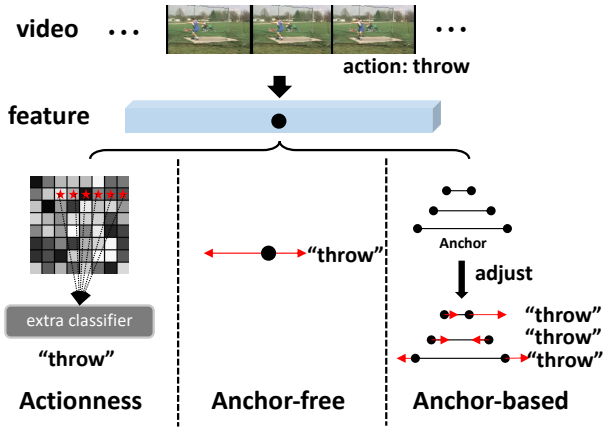

Figure 1. Compared with actionness and anchor-based methods, anchor-free method is more efficient and flexible to produce fewer proposals without any extra classifier and pre-defined anchors.

Current TAL models mainly focus on learning actionness of each frame [20, 18, 17, 26, 27, 24] or adjustment of pre-defined anchors [38, 23, 22], named actionness-guided methods and anchor-based methods, as shown in Fig. 1. In spite of reasonable good results on benchmark datasets, such methods are still limited to the following points: (1) Both methods will produce a bunch of redundant proposals. For example, given a video with T frames, we have to produce $\mathcal{O}(T^2)$ and $C \cdot T$ proposals for the "actionness-guided" BSN [20], and "anchor-based" R-C3D [38], individually. Here C is the number of pre-defined anchors. These proposals lead to prohibitive computational cost in both calcu-

* indicates equal contributions.
This work was done when Chengming Xu was an intern at Tencent Youtu Lab. Yanwei Fu is the corresponding author.

lating the training loss and post-processing for testing. (2) Actionness-guided methods can solely provide predictions of temporal boundary, while they have to rely on the extra model such as S-CNN [33] and P-GCN [41] for classification. Nevertheless, the models of two stages are isolated and thus incapable of sharing information for the end-to-end update. (3) Typically, anchor-based methods are very sensitive to some critical hyper-parameters, such as the number and size of pre-defined anchors; and it is very non-trivial to tune these hyper-parameters in the real-world applications.

Alternatively, an efficient recipe for localization is to resort to the *anchor-free* method, which does not require predefined anchors. Typically, this type of method only generates one proposal for each temporal location in the form of a pair of values representing the distance between the start and end moments to the current location, individually.

In contrast to the existed methods, anchor-free model saves huge amount of pre-defined anchors, while assembling both boundary regression and classification in one model, thus being productive. Furthermore, even though some pilot studies, *e.g.*, Yang *et. al.* [40] observed relatively weak results for anchor-free methods, the supporting evidence in object detection [42] shows that such methods with well designed network structure and training strategy should, in principle, be comparable with anchor-based ones.

To this end, in this paper we propose a novel purely anchor-free TAL framework dubbed Anchor-Free Saliency-based Detector (AFSD). Essentially, we first build a naive anchor-free predictor containing an end-to-end trainable backbone network, a feature pyramid network and a simple prediction network which outputs the action class and the temporal distance of the start and end from each location. To learn a more accurate boundary, we refer to former TAL methods [20, 18] indicating the importance of boundary or context feature. These works obtain such features mainly by merging the neighbor of start and end moments with convolutions or mean pooling. However, we claim that in fact moment-level feature is more valuable than region-level feature for distinguishing whether an action starts or ends. As shown in Fig. 2, the background regions near the start and end moments are showing other irrelevant scenes, while regions inside the action are almost the same, which cannot provide any information for judging if the action starts or ends. Such an example indicates the importance of a moment-level feature.

Therefore, we propose a novel boundary pooling which, instead of aggregating the whole region, tries to find the most salient moment-level feature for both start and end regions. We further equip the boundary pooling with a newly proposed Boundary Consistency Learning (BCL) strategy to regularize the pooling operation to provide the correct boundary features for each action. In detail, we employ a modified ground truth signal indicating start and end moments to guide the model. Then, we rearrange the video clips to help model discriminate background and action features by self-supervised contrastive learning. We conduct extensive experiments on THUMOS14 and ActivityNet1.3. On THUMOS14 our model attains 3.7% improvement on *m*AP@0.5 against the state-of-the-art methods. The results on ActivityNet1.3 are also comparable.

Figure 2. An action instance of cliff diving. Note that the start and end moments of the movement are more salient than others, which can bring us significant information to judge the boundary and completeness of the action.

In summary, our paper has the following contributions:

1. We, for the first time, propose a purely anchor-free temporal action localization model. This model enjoys not only less hyper-parameter to tune and less outputs to process, but also better performance, thus making the best of both worlds.

2. To make full use of the anchor-free framework, we discuss the impact of boundary features and propose novel boundary pooling method whose output is used along with the coarse proposals to generate fine-grained predictions. Moreover, we introduce a novel Boundary Consistency Learning strategy which can constrain the model to learn better boundary feature.

2. Related Work

Anchor-based Localization Anchor-based localization models rely on adjusting the pre-defined anchors. TURN [10] aggregates features from basic video unit for clip-level features, which are used to classify the activity and regress the temporal boundary. R-C3D [38] takes the inspiration from faster-RCNN [31], which utilizes a streamline including proposal generation, proposal-wise pooling and final prediction. GTAN [23] modifies the pooling procedure, adopting a weighted average via a learnable Gaussian kernel for each proposal. Due to the fixed pre-defined anchors, such methods are not flexible when it comes to various action classes. Different from them, our model does not require tuning extra hyper-parameters for anchors, thus more efficient.

Actionness-guided Localization Unlike anchor-based methods, actionness-guided methods mainly focus on evaluating 'actionness', which indicates the probability of a potential action, for each frame or clip in a video. The actionness is afterwards post-processed to generate action proposals. Zhao *et. al.* designed SSN [44] in which course proposals are first divided into three semantic parts, learned

respectively. Next probability of activity and completeness is predicted and used to merge different proposals. Lin *et. al.* proposed BSN [20] which learns to predict start, end and actionness of each temporal location. The proposals are generated by gathering locations with high start and end probabilities, with low confidence ones further abandoned by an evaluation module. They later improved this framework to BMN [18], which additionally generates a Boundary-Matching confidence map to help get better proposals. While no pre-defined anchors are required for actionness-guided methods, such methods are more like enumeration methods where all possible combinations of temporal locations are considered, thus totally different from anchor-free localization where boundaries are directly predicted for each time step.

Anchor-Free Object Detection Analogous to TAL, there is a surge of the usage of anchor-free methods in object detection. YOLO [30] is the most well-known anchor-free method, in which a neural network model is directly used to predict coordinates of bounding boxes from raw images. Such framework is too simple, thus suffering from poor performance. Following works mainly focus on improving performance via setting different prediction targets and using more detailed features. CornerNet [16] lets the model learn to predict top-left and bottom-right keypoints of each bounding box. FCOS [35] aims to learn the distance to boundaries of each spatial location and utilizes feature pyramid for objects with diverse scales. BorderDet [28] modifies the RoI pooling into BorderAlign to get more powerful proposal-level feature. We take inspiration from these methods to design a basic anchor-free localizer, along with making full use of the temporal insights of videos to propose novel refinement strategy and consistency learning.

Contrastive Learning There has been an increasing interest in contrastive learning used in unsupervised learning. Compared with the application of contrastive learning in image understanding [8, 13], fewer contributions of contrastive learning have been made in video understanding. Guillaume *et. al.* [25] proposed a temporal contrastive training strategy for action recognition, in which an autoregressive model is used to predict future video segment given enough information, and then compare the prediction with ground truth. Gong *et. al.* [12] adopted a contrastive scoring method to evaluate action proposals in unsupervised temporal localization in the inference phase. Compared with these works, we make further trial into leveraging contrastive learning to help train a supervised temporal localization model, which has never been studied before.

3. Method

Denote a video dataset as $\mathcal{T} = \{\mathcal{T}^{train}, \mathcal{T}^{test}\}$, each data instance $\{X, \Psi_X\}$ contains a video $X = \{x_t\}_{t=1}^{T}$ with T RGB frames or optical flows. The corresponding annotation Ψ_X can be depicted as tuples $\{(\phi_m, y_m)\}_{m=1}^{M_X}$ where M_X is the number of action instances in X, $\phi_m = (\psi_m, \xi_m)$ denotes the start time, end time and y_m indicates the action category. Our goal is to train a model to predict proposals with class scores which could have high recall and precision with the ground truth on the test set $\mathcal{T}^{test}$.

Overview We propose a purely anchor-free architecture named AFSD which is shown in Fig. 3. Concretely, given a video X, we first process the video with a backbone network and a feature pyramid network. Take RGB frames as example, for each video X, we use a Kinetics pre-trained I3D [6] model to extract a 3D feature $F \in \mathbb{R}^{T^{'} \times C^{'} \times H^{'} \times W^{'}}$, where $T^{'}, C^{'}, H^{'}, W^{'}$ denote the time step, channel, height and width individually. This feature is afterwards flattened along the last three dimensions to a 1D feature sequence. Such a sequence can contain the temporal and spatial information of whole video. We then exert a feature pyramid network including several temporal convolutions, of which the detailed architecture is shown in our supplement, to merge the spatial dimension and aggregate the temporal dimension in different levels. The pyramid features are further utilized to generate a coarse proposal sequence $\{(\hat{\psi}_i, \hat{\xi}_i, \hat{y}_i^C)\}$ with a basic anchor-free prediction module (Sec. 3.1), which includes a simple regressor and classifier. After that for each proposal, the predicted temporal regions are employed to get the salient boundary features with the boundary pooling (Sec. 3.2). The boundary features are exploited together with the feature pyramid to output a fine-grained prediction $\{(\Delta\hat{\psi}_i, \Delta\hat{\xi}_i, \hat{y}_i^R)\}$ for both temporal regression and action classification.

3.1. Basic Prediction Module

We first build a basic anchor-free prediction module to get coarse temporal boundaries. For instance, for the feature of the l-th FPN level $f^l \in \mathbb{R}^{T_l \times C}$, we first project it to features f_{loc}^l and f_{cls}^l embedded in two latent space corresponding to localization and classification by two branches with two temporal convolutions respectively. Both of these two features f_{loc}^l and f_{cls}^l are processed with one layer of temporal convolution shared among all FPN layers to get coarse start and end boundary distances $(\hat{d}_i^s, \hat{d}_i^e)$ and class score $\hat{y}_i$ for each location i. Next we can get start and end time for i-th time step in l-th level as follow:

$$\begin{aligned} \hat{\psi}_i &= i * 2^l - \hat{d}_i^s, \\ \hat{\xi}_i &= i * 2^l + \hat{d}_i^e. \end{aligned} \tag{1}$$

T_l proposals are generated for l-th FPN layer in all. Such a simple framework can already temporally detect actions in a anchor-free manner, which enjoys several merits including no requirement for pre-defined anchors and less but more accurate predictions as discussed in Sec. 1. In the following sections, we focus on designing appropriate modules and

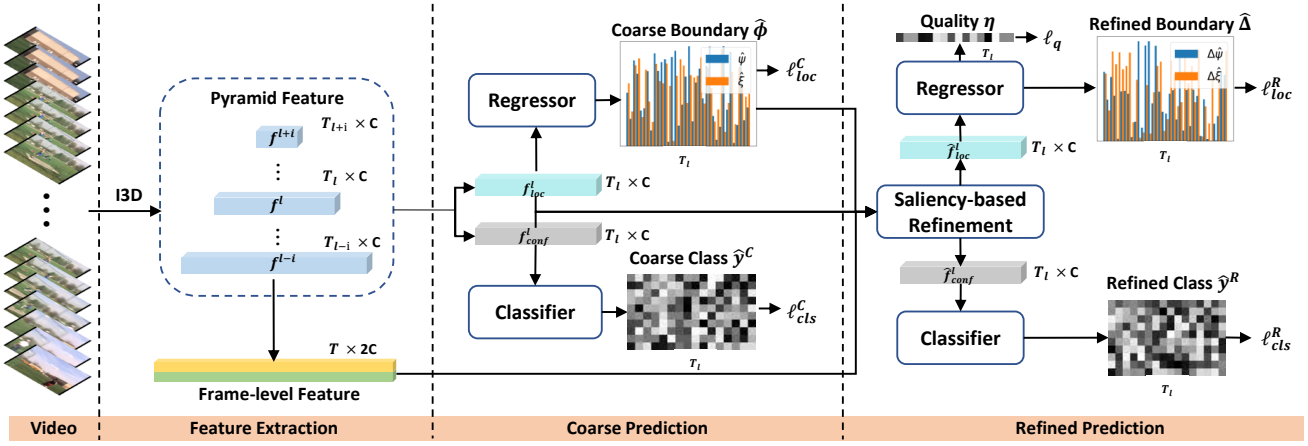

Figure 3. The overview of our approach. Given an input video X, we employ I3D model to extract feature and construct 1D temporal pyramid features. Next, each pyramid feature is utilized to generate coarse proposals via basic prediction module. Finally, our saliency-based refinement module will adjust the class score, start and end boundaries and predict the corresponding quality score for each coarse proposal. Note that our model is a fully end-to-end method and trained with I3D feature extraction network without any preprocessing.

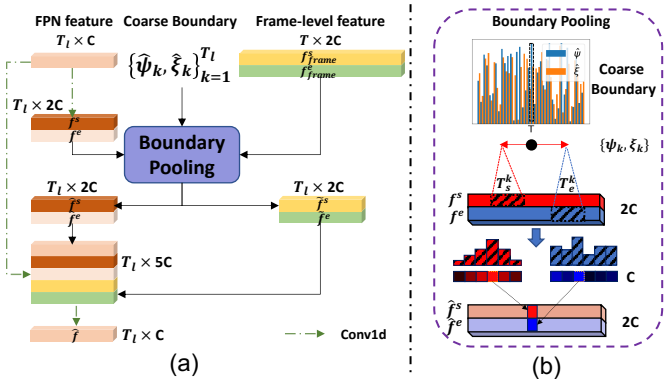

Figure 4. (a) Saliency-based Refinement Module: utilize coarse boundaries, FPN feature and frame-level feature to construct salient boundary feature. (b) Boundary Pooling: search salient moment features in the boundary regions of the input feature.

training strategy for anchor-free TAL methods with better performance.

3.2. Saliency-based Refinement Module

As mentioned in Sec. 1, several existing works have shown the importance of boundary feature in TAL, especially for predicting the temporal distance. However, since different action instances could have various lengths, it is hard to attain these boundary information for all proposals via several simple temporal convolutions because of the limited receptive field. Therefore, we propose a saliency-based refinement module which is illustrated in Fig. 4(a), where we utilize the FPN features along with the coarse proposals to help our model gather boundary features to refine the predictions. For simplicity, we omit the subscript standing for FPN layers in the following detail. Take the localization feature f_{loc} for example, first we project it into two latent spaces sensitive to start and end activities respectively via convolutional layers:

$$\begin{aligned} f^s &= \sigma(\text{GN}(\text{Conv1}(f_{loc}))) \quad \in \mathbb{R}^{T_l \times C}, \\ f^e &= \sigma(\text{GN}(\text{Conv2}(f_{loc}))) \quad \in \mathbb{R}^{T_l \times C}, \end{aligned} \tag{2}$$

where σ and GN denote ReLU and Group Normalization [37]. With the projection, the model can learn start and end sensitive signals separately, thus leaving less learning load to the FPN features for better training.

Then, given the coarse boundary results $\{(\hat{\psi}_k, \hat{\xi}_k)\}_{k=1}^{T_l}$ of the corresponding l-th FPN level, the k-th start and end regions T_s^k, T_e^k are constructed as:

$$\begin{aligned} T_s^k &= \left[\hat{\psi}_k - \frac{\hat{w}_k}{\delta_a}, \hat{\psi}_k + \frac{\hat{w}_k}{\delta_b}\right], \\ T_e^k &= \left[\hat{\xi}_k - \frac{\hat{w}_k}{\delta_b}, \hat{\xi}_k + \frac{\hat{w}_k}{\delta_a}\right], \end{aligned} \tag{3}$$

where $\hat{w}_k = \hat{\psi}_k - \hat{\xi}_k$ means the length of proposal, δ_a, δ_b are hyper-parameters controlling the proportion of selected regions outside and inside the proposal.

Next, an aggregation function $\mathcal{A}$ is applied to f^s and f^e in start and end regions respectively to collect the relevant boundary features. Despite a lot of instantiations of $\mathcal{A}$ in former works, such as mean pooling [10], Gaussian weighted average [23] and directly gathering and concatenating [17], these methods would introduce useless knowledge from frames not representing the action boundaries, thus blocking the model from precise predictions. Hence we propose a novel boundary pooling method to get the moment-level boundary feature $\hat{f}^s, \hat{f}^e \in \mathbb{R}^{T_l \times C}$ as shown in Fig. 4(b). The boundary pooling works as following:

$$\begin{aligned} \hat{f}^s(k,i) &= \max_{j \in T_s^k} f^s(j,i) \quad i = 1, \cdots, C, \\ \hat{f}^e(k,i) &= \max_{j \in T_e^k} f^e(j,i) \quad i = 1, \cdots, C. \end{aligned} \tag{4}$$

Maximization is utilized aiming to select the largest activated cell, *i.e.*, the most salient moment, for each channel along the temporal region. Note that as FPN goes deeper, the temporal dimension decreases to be too small for boundary pooling to find the appropriate boundary. Therefore, we add a shared frame-level feature f_{frame} by applying upsample and several convolutions to the bottom FPN feature, from which start and end frame-level features $\tilde{f}^s, \tilde{f}^e$ are extracted for each proposal with the same projection and pooling procedure as in Eq. 2 and Eq. 4.

After boundary pooling, the refined features are built by concatenating original features and all boundary features. A temporal convolution is applied to reduce channels:

$$\hat{f} = \text{Conv}(f||\hat{f}^s||\hat{f}^e||\tilde{f}^s||\tilde{f}^e), \tag{5}$$

where $\cdot||\cdot$ denotes channel-wise concatenation. These features are again employed as input of a simple network with temporal convolution to predict offsets of regression $(\Delta\hat{\psi}, \Delta\hat{\xi})$, which can be added to the coarse predictions to get a fine-grained ones $(\tilde{\psi}, \tilde{\xi})$, and refined class score $\hat{y}^R$.

3.3. Boundary Consistency Learning

Although the boundary pooling can extract the most salient features, it cannot guarantee that the pooled features represent the true action boundary. Such a property is pivotal since if boundary pooling focuses on a background frame, the model cannot have enough useful information, and thus being misled to wrong results. To regularize our boundary pooling, we further propose Boundary Consistency Learning (BCL), which has two components: Activation Guided Learning and Boundary Contrastive Learning.

Activation Guided Learning Specifically, we re-scale the sensitive features f^s and f^e and take channel-wise mean:

$$\tilde{g}^s = \text{mean}(\tanh(f^s)), \tilde{g}^e = \text{mean}(\tanh(f^e)). \tag{6}$$

These two features can be seen as confidence revealing the probability of occurrence of start or end moment. We can obtain the ground truth $g^s, g^e \in \mathbb{R}^T$ by following definition:

$$\begin{aligned} g^s(i) &= \mathbb{I}(i \in \mathcal{B}(\psi_m) \text{ for } \forall m \in [1, M_X]), \\ g^e(i) &= \mathbb{I}(i \in \mathcal{B}(\xi_m) \text{ for } \forall m \in [1, M_X]), \end{aligned} \tag{7}$$

where $\mathcal{B}(\cdot)$ denotes the neighbor and $\mathbb{I}(\cdot)$ is the indicator function. After that we can calculate the Cross Entropy:

$$\ell_{act} = \text{BCE}(g^s, \tilde{g}^s) + \text{BCE}(g^e, \tilde{g}^e), \tag{8}$$

where BCE denotes the binary cross entropy. With g^s and g^e as guidance, we can constrain the feature to have high activation at the occurrence and closure of each action.

Boundary Contrastive Learning Consider a video X containing an action instance A and other background. If we split the action and fill in a random part of background, we will have two incomplete action fragments A_1, A_2 and background Bg between them. Applying boundary pooling to these three regions leads to three pairs of features $(f^s_{A_1}, f^e_{A_1}), (f^s_{A_2}, f^e_{A_2}), (f^s_{Bg}, f^e_{Bg})$. In general, since A_1 and A_2 are continuous, the sensitive features $f^e_{A_1}$ and $f^s_{A_2}$ should be similar to each other and distant to f^s_{Bg} and f^e_{Bg} if we restrict boundary pooling to only make use of few frames inside the actions (*i.e.*, a large δ_b in Eq. 3). However, this property could be broken when model is learnt to take high activations on background. In such situation, $f^e_{A_1}$ would be close to f^s_{Bg} and $f^s_{A_2}$ would be close to f^e_{Bg}. Therefore, a good way to guarantee appropriate features is to apply the contrastive learning on these features, enlarging the distance between the sensitive features of action fragments and background. Formally, it can be realized by utilizing the following triplet objective function:

$$\ell_{trip} = \max(\|f^e_{A_1} - f^s_{A_2}\|^2 - \|f^e_{A_1} - f_{Bg}\|^2 + 1, 0), \tag{9}$$

where $f_{Bg} \in \left\{f^s_{Bg}, f^e_{Bg}\right\}$. In practice, we first count the minimal action length w_{min} in one video. Next if the video has an action instance whose length is larger than $2 \cdot w_{min}$ and a background clip with length w_{min}, we include this video into our contrastive learning pool and implement the above splitting procedure. In this way, both split action and background are long enough to be distinguished. In together, our Boundary Consistency Learning can be summarized into the following form:

$$\ell_{con} = \ell_{act} + \ell_{trip}. \tag{10}$$

3.4. Training and Inference

Label Assignment For coarse prediction, we assign each location i as a positive sample to ground truth j when $\psi_j \le i \le \xi_j$ is satisfied. For refined prediction, we calculate the temporal IoU (tIoU) between each coarse boundary prediction and the corresponding ground truth. A location i is regarded as positive if its tIoU score is greater than 0.5. Denote N_C, N_R as the number of positive samples for coarse and refined predictions individually.

Training Once having both the coarse and refined prediction of temporal boundary and class label, we can optimize the model with the following objective function:

$$\mathcal{L} = \ell^C_{cls} + \lambda\ell^C_{loc} + \ell^R_{cls} + \lambda\ell^R_{loc} + \gamma\ell_q, \tag{11}$$

where λ, γ are hyper-parameters, $\ell^C_{cls}, \ell^R_{cls}$ are softmax focal loss ℓ_{focal} [21] between both classification prediction $\left\{\hat{y}^C, \hat{y}^R\right\}$ and ground truth labels y:

$$\ell_{cls}(\{\hat{y}_i\}) = \frac{1}{N}\sum_i \ell_{focal}(\hat{y}_i, y_i), \tag{12}$$

where $N \in \{N_C, N_R\}$. ℓ^C_{loc} is a tIoU loss between coarse boundaries $\hat{\phi}_i = (\hat{\psi}_i, \hat{\xi}_i)$ and the corresponding ground

truth $\phi_i = (\psi_i, \xi_i)$. ℓ_{loc}^R is a L1 loss between the predicted offset $\hat{\Delta}_i = (\Delta\hat{\psi}_i, \Delta\hat{\xi}_i)$ and the corresponding offset label $\Delta_i = (\Delta\psi_i, \Delta\xi_i)$:

$$\begin{aligned}\ell_{loc}^C(\{\hat{\phi}_i\}) &= \frac{1}{N_C}\sum_i \mathbb{I}(y_i \geq 1)(1 - \frac{|\hat{\phi}_i \cap \phi_i|}{|\hat{\phi}_i \cup \phi_i|}),\\ \ell_{loc}^R(\{\hat{\Delta}_i\}) &= \frac{1}{N_R}\sum_i \mathbb{I}(y_i \geq 1)(|\hat{\Delta}_i - \Delta_i|).\end{aligned} \tag{13}$$

ℓ_q is a quality loss used to suppress the proposals with low quality. As a counterpart in the object detection, FCOS [35] proposes the centerness of each spatial location as the quality target. However, such definition of centerness for actions is vague since it is hard to decide the exact frame being a start or end signal of an action, and thus it is inappropriate to directly use centerness in TAL. To better predict quality of proposals, we utilize tIoU between boundary prediction $\tilde{\phi}$ and the location labels as the learning target of quality confidence η generated from refined feature $\tilde{f}$:

$$\ell_q(\{\eta_i\}) = \frac{1}{N_R}\sum_i \mathbb{I}(y_i \geq 1)\text{BCE}(\eta_i, \frac{|\tilde{\phi}_i \cap \phi_i|}{|\tilde{\phi}_i \cup \phi_i|}). \tag{14}$$

For each batch in training, we first optimize the model with $\mathcal{L}$. Then we seek the data available for BCL in Sec. 3.3 and train the model with ℓ_{con} in Eq. 10.

Inference For the i-th temporal location in l-th FPN layer, the final predictions are formalized through all outputs from our model, including coarse predictions $\hat{\psi}_{l,i}, \hat{\xi}_{l,i}, \hat{y}_{l,i}^C$ and refined ones $\Delta\hat{\psi}_{l,i}, \Delta\hat{\xi}_{l,i}, \hat{y}_{l,i}^R, \eta_{l,i}$, in the following form:

$$\begin{aligned}\hat{w}_{l,i} &= \hat{\xi}_{l,i} - \hat{\psi}_{l,i},\\ \tilde{\psi}_{l,i} &= \hat{\psi}_{l,i} + \frac{1}{2}\hat{w}_{l,i}\Delta\hat{\psi}_{l,i},\\ \tilde{\xi}_{l,i} &= \hat{\xi}_{l,i} + \frac{1}{2}\hat{w}_{l,i}\Delta\hat{\xi}_{l,i},\\ \hat{y}_{l,i} &= \frac{1}{2}(\hat{y}_{l,i}^C + \hat{y}_{l,i}^R)\eta_{l,i}.\end{aligned} \tag{15}$$

We then assemble all predictions and process them with Soft-NMS [3] to suppress redundant proposals.

4. Experiments

4.1. Datasets and Settings

Datasets To validate the efficacy of our model, we conduct extensive experiments on commonly-used benchmark THUMOS14 [14] and ActivityNet1.3 [5]. *THUMOS14* is composed of 200 validation videos and 212 testing videos from 20 categories labeled for temporal localization. *ActivityNet1.3* has 19,994 videos with 200 action classes. We follow the former setting [20] to split this dataset into training, validation and testing subset by 2:1:1.

Implementation Details On THUMOS14, we sample both RGB and optical flow frames at 10 frames per second (fps) and split video into clips. The length of each clip T is set as 256 frames. Adjacent clips have a temporal overlap of m frames and m is set to 30 in training and 128 in testing. On ActivityNet1.3, we sample frames using different fps and ensure the number of video frames is 768 for each video. Thus, each video has only one clip with 768 frames. On both datasets, the frame spatial size is set to 96×96. Random crop, horizontal flipping are used as data augmentation in training. To extract features of clips, we finetune a I3D [6] model pre-trained on Kinetics.

Our model is trained for 16 epoches using Adam [15] with learning rate of 10^{-5}, weight decay of 10^{-3}. Batch size is set to 1. We set δ_a to 4 and δ_b to 100 for ℓ_{con} and δ_b to 10 for other loss terms. The weight of loss λ is set to 10 on THUMOS14 and 1 on ActivityNet1.3 and γ is set to 1 empirically. In the testing phase, the results of RGB and optical flow frames are averaged to obtain final locations and class scores. The tIoU threshold in Soft-NMS is set to 0.5 for THUMOS14 and 0.85 for ActivityNet1.3.

Metrics We report mean Average Precision (*m*AP) in all experiments. The thresholds are $[0.3 : 0.1 : 0.7]$ for THUMOS14 and $[0.5 : 0.05 : 0.95]$ for ActivityNet1.3.

4.2. Main Results

We compare our model with state-of-the-art methods in Tab. 1 and report the backbone used by each method, including TS [34], C3D [36], P3D [29] and I3D [6]. On THUMOS14, our AFSD outperforms the strongest competitor A2Net and G-TAD on all thresholds by large margin, especially 7.7% on *m*AP@0.6. The remarkable improvement comes along with high efficiency, thus making our model more practical for real TAL scenarios. Note that while A2Net also has an anchor-free module, the performance of their merged model is far worse than ours, not to mention the sole anchor-free branch, which proves the superiority of our architecture for anchor-free methods.

On ActivityNet1.3, the results are still comparable. Our model receives the best *m*AP@0.75 and average *m*AP compared to the strongest competitor GTAN. It is noteworthy that while all of the actionness-guided methods have less average *m*AP than ours, most of them enjoy a higher *m*AP@0.95. One possible reason is that they can enumerate all potential proposals, thus the ground truth proposals are already contained in the alternative prediction set. With such an enumeration strategy the actionness-based methods would be better when dealing with harder datasets like ActivityNet1.3. Compared with actionness-guided methods, our model is more efficacious in the sense of producing less proposals and attaining better overall performance when considering multiple thresholds. Additionally, compared with THUMOS14, ActivityNet is less well annotated as ex-

Type	Model	Backbone	THUMOS14						ActivityNet1.3			
			0.3	0.4	0.5	0.6	0.7	Avg.	0.5	0.75	0.95	Avg.
Anchor-based	SSAD [19]	TS	43.0	35.0	24.6	—	—	—	—	—	—	—
	TURN [10]	C3D	44.1	34.9	25.6	—	—	—	—	—	—	—
	R-C3D [38]	C3D	44.8	35.6	28.9	—	—	—	26.8	—	—	—
	CBR [11]	TS	50.1	41.3	31.0	19.1	9.9	30.3	—	—	—	—
	TAL [7]	I3D	53.2	48.5	42.8	33.8	20.8	39.8	38.2	18.3	1.3	20.2
	GTAN [23]	P3D	57.8	47.2	38.8	—	—	—	**52.6**	34.1	8.9	34.3
Actionness	CDC [32]	—	40.1	29.4	23.3	13.1	7.9	22.8	45.3	26.0	0.2	23.8
	SSN [44]	TS	51.0	41.0	29.8	—	—	—	43.2	28.7	5.6	28.3
	BSN [20]	TS	53.5	45.0	36.9	28.4	20.0	36.8	46.5	30.0	8.0	30.0
	BMN [18]	TS	56.0	47.4	38.8	29.7	20.5	38.5	50.1	34.8	8.3	33.9
	DBG [17]	TS	57.8	49.4	42.8	33.8	21.7	41.1	—	—	—	—
	G-TAD [39]	TS	54.5	47.6	40.2	30.8	23.4	39.3	50.4	34.6	9.0	34.1
	BU-TAL [43]	I3D	53.9	50.7	45.4	38.0	28.5	43.3	43.5	33.9	9.2	30.1
	BC-GNN [2]	TS	57.1	49.1	40.4	31.2	23.1	40.2	50.6	34.8	**9.4**	34.3
Other	A2Net [40]	I3D	58.6	54.1	45.5	32.5	17.2	41.6	43.6	28.7	3.7	27.8
	G-TAD+PGCN [41]	I3D	66.4	60.4	51.6	37.6	22.9	47.8	—	—	—	—
Anchor-free	Ours	I3D	**67.3**	**62.4**	**55.5**	**43.7**	**31.1**	**52.0**	52.4	**35.3**	6.5	**34.4**

Table 1. Performance comparison with state-of-the-art methods on THUMOS14 and ActivityNet1.3, measured by *m*AP at different IoU thresholds, and average *m*AP in $[0.3 : 0.1 : 0.7]$ on THUMOS14 and $[0.5 : 0.05 : 0.95]$ on ActivityNet1.3.

plained by official report [1] showing 'it is hard to agree about the temporal boundaries', which partially attributes to the slight improvement.

4.3. Ablation Study

To further verify the efficacy of our contributions, we conduct several ablation studies on THUMOS14 for the RGB model, including each part of our model and the choice of hyper-parameters.

Effectiveness of Quality Confidence We first compare the basic prediction module introduced in Sec. 3.1 trained and inferred with and without the proposed quality confidence in Tab. 2(a). Besides, we add an extra model using the centerness proposed in FCOS. The results show that centerness leads to 1.3% drop on *m*AP@0.7, doing no help for training TAL model. In contrast, model with quality loss ℓ_q (Eq. 14) can have 1.0% average *m*AP improvement, which supports our claim that the definition of centerness is somehow inappropriate in TAL, thus cannot be directly applied. Compared with that, our quality loss ℓ_q is a more suitable objective function for suppressing low-quality action proposals.

Choice of Signal Normalization In Action Guided Learning introduced in Sec. 3.3, we adopt a *tanh* function to normalize the feature vector. We compare this one with another two instantiations. One is a hard clipping between $[0, 1]$. The other is to use simple standardization in the following form, which is denoted as **0-1 norm** in Tab. 2(b):

$$f(i) = \frac{f(i) - \min_{i=1}^{T} f(i)}{\max_{i=1}^{T} f(i) - \min_{i=1}^{T} f(i)}. \tag{16}$$

The results show that using *tanh* can have 0.9% and 0.6% average *m*AP advantage against two alternatives.

Choice of Boundary Region In Tab. 2(c) we assess choices of the boundary region used for pooling boundary features, which is composed of (1) a symmetric area with the coarse boundary as center point, denoted as $\delta_a = \delta_b$. (2, 3) two asymmetric regions either focusing on background or action. Through the results we can safely say that it is better to keep a larger proportion of region inside the coarse action than the background. The possible reason is that our naive predictor can already produce a relatively accurate prediction. As shown in Tab. 2(a), the performance of baseline can beat most of the competitors in Tab. 1 even if the competitors fuse RGB and optical flow models for final prediction and our baseline only utilizes RGB frames as input.

Effectiveness of Boundary Refinement In Tab. 2(d) we compare four forms of boundary refinement, including: (1) **Naive**: Only several temporal convolutions are directly applied to f_{loc}, f_{cls} to get another prediction. In this way, the information disposed for refinement is the stacked neighbor introduced by the convolutions. (2) **Self**: The saliency-based refinement module is utilized without f_{frame}, and thus the consistency learning only calculates losses corresponding to the FPN feature. (3) **Frame-level**: only f_{frame} is adopted in temporal refinement and FPN features are abandoned. (4) **All**: All usable features are included, which is our final model. The results demonstrate that: (1) Boundary information is more valuable for refinement than that from neighborhood of each temporal location. (2) Frame-level feature can only be taken as a complement of FPN features. Using only frame-level feature would result in 1.1% average *m*AP drop against the naive refinement model.

Instantiation of Boundary Pooling We compare our proposed boundary pooling with the following three instantia-

Model	0.5	0.6	0.7	Avg.
baseline	43.1	31.0	19.0	40.4
+centerness	43.3	31.6	17.7	40.2
+quality	**44.0**	**32.0**	**19.8**	**41.4**

(a) **Training strategy**: we compare our quality confidence with centerness loss proposed in FCOS.

Model	0.5	0.6	0.7	Avg.
0-1 clip	45.3	34.6	22.4	42.6
0-1 norm	45.4	34.9	21.6	42.9
tanh	**45.9**	**35.0**	**23.4**	**43.5**

(b) **Feature normalization**: we compare different choice of feature normalization in constraint.

Model	0.5	0.6	0.7	Avg.
$\delta_a = \delta_b$	45.0	33.4	21.2	42.3
$\delta_a > \delta_b$	45.3	34.7	21.8	42.6
$\delta_a < \delta_b$	**45.9**	**35.0**	**23.4**	**43.5**

(c) **Boundary feature extraction**: we compare different choice of boundary regions.

Model	0.5	0.6	0.7	Avg.
naive	44.9	32.5	19.9	41.6
self	44.6	33.9	22.3	42.4
frame	42.9	31.9	20.2	40.5
all	**45.9**	**35.0**	**23.4**	**43.5**

(d) **Refinement**: we compare different sources of information for refinement.

Model	0.5	0.6	0.7	Avg.
mean	45.1	34.6	22.1	42.7
conv	44.8	34.4	22.3	42.6
stack	44.9	33.6	22.3	42.9
max	**45.9**	**35.0**	**23.4**	**43.5**

(e) **Instantiations**: we compare different forms of boundary feature extraction.

Model	0.5	0.6	0.7	Avg.
w/o BCL	44.3	34.1	21.2	42.0
ℓ_{act}	45.6	34.4	22.3	42.7
ℓ_{trip}	45.5	34.8	22.4	42.7
$\ell_{act} + \ell_{trip}$	**45.9**	**35.0**	**23.4**	**43.5**

(f) **Consistency learning**: we compare models varied with Boundary Consistency Learning.

Table 2. Ablation studies of RGB model on THUMOS14, measured by *m*AP at 0.5, 0.6 and 0.7, and average *m*AP in $[0.3 : 0.1 : 0.7]$.

Model	GPU	FPS
S-CNN [33]	—	60
DAP [9]	—	134
CDC [32]	TITAN Xm	500
SS-TAD [4]	TITAN Xm	701
R-C3D [38]	TITAN Xm	569
R-C3D [38]	TITAN Xp	1030
Ours	1080 Ti	**3259**
Ours	V100	**4057**

Table 3. Comparison of inference speed between our method and other methods on THUMOS14.

tions: (1) **Mean**: The max operation is replaced with mean. (2) **Conv**: We sample three temporal positions for each region, and a 1-layer temporal convolution is applied to aggregate them. (3) **Stack**: Similar to (2), while instead of using convolution, we directly concatenate these three features into one boundary feature. Note that all models are trained with our boundary consistency learning. The results are presented in Tab. 2(e). Among all instantiations, pooling with max receives the best performance, showing 0.8%, 0.9% and 0.6% advantage of average *m*AP against mean, conv and stack respectively. It is noteworthy that the improvement of our model over others is larger than 1.0% on *m*AP@0.7, which indicates that moment-level boundary feature helps the model generate more accurate proposals.

Effectiveness of Boundary Consistency Learning We verify the proposed BCL by comparing our full model with that trained only with $\mathcal{L}$ (Eq. 11) and without ℓ_{con} (Eq. 10). Results in Tab. 2(f) suggest that when trained without any consistency guarantee, the model cannot learn good representations for boundary pooling. Therefore lack of useful information in refinement leads to worse performance. Moreover, each loss term brings 0.7% improvement on average *m*AP and by assembling ℓ_{act} (Eq. 8) and ℓ_{trip} (Eq. 9) together, our model finally achieves the best performance. To further validate the efficacy of BCL, we obtain the learned frame-level feature f_{frame} of an action instance and its neighboring background, each half part of which could represent the start and end signals in hypothesis.

Comparison on Inference Time As discussed, the proposed AFSD is highly efficient. To verify this claim we report the inference speed on THUMOS14 in terms if fps among different models in Tab. 3. Thus results show that our model is much more faster than the existing methods. The main reasons is that after features are extracted we can employ both a lighter predictor and a lighter refinement module composed of 1D convolution due the the help of anchor-free mechanism. Moreover, the number of proposals produced by our model is less than that of these methods, which also helps speed our model.

5. Conclusion

In this paper we explore the possibility of a novel form of temporal action localization model — anchor-free method. We discuss the merits over anchor-based methods, and actionness-guided methods and design a dedicated anchor-free model. Our model includes an end-to-end trainable basic predictor and a temporal refinement module. For the refinement module, we analyze the drawbacks of existing means to extract boundary features, and propose a novel boundary pooling together with a Boundary Consistency Learning strategy. We achieve remarkable results on THUMOS14 and comparable ones on ActivityNet1.3. The results indicate the strength of anchor-free model as a promising choice for solving temporal action localization.

References

[1] Humam Alwassel, Fabian Caba Heilbron, Victor Escorcia, and Bernard Ghanem. Diagnosing error in temporal action detectors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 256–272, 2018. 7

[2] Yueran Bai, Yingying Wang, Yunhai Tong, Yang Yang, Qiyue Liu, and Junhui Liu. Boundary content graph neural network for temporal action proposal generation. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020. 7

[3] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017. 6

[4] Shyamal Buch, Victor Escorcia, Bernard Ghanem, Li Fei-Fei, and Juan Carlos Niebles. End-to-end, single-stream temporal action detection in untrimmed videos. 2019. 8

[5] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015. 6

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 3, 6

[7] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1130–1139, 2018. 7

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3

[9] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *European Conference on Computer Vision*, pages 768–784. Springer, 2016. 8

[10] Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE international conference on computer vision*, pages 3628–3636, 2017. 2, 4, 7

[11] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Cascaded boundary regression for temporal action detection. *arXiv preprint arXiv:1705.01180*, 2017. 7

[12] Guoqiang Gong, Xinghan Wang, Yadong Mu, and Qi Tian. Learning temporal co-attention models for unsupervised video action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9819–9828, 2020. 3

[13] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 3

[14] 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. 6

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, pages 734–750, 2018. 3

[17] 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*, pages 11499–11506, 2020. 1, 4, 7

[18] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3889–3898, 2019. 1, 2, 3, 7

[19] Tianwei Lin, Xu Zhao, and Zheng Shou. Single shot temporal action detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 988–996, 2017. 7

[20] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*, pages 3–19, 2018. 1, 2, 3, 6, 7

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[22] Qinying Liu and Zilei Wang. Progressive boundary refinement network for temporal action detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11612–11619, 2020. 1

[23] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 344–353, 2019. 1, 2, 4, 7

[24] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Learning to localize actions from moments. *arXiv preprint arXiv:2008.13705*, 2020. 1

[25] Guillaume Lorre, Jaonary Rabarisoa, Astrid Orcesi, Samia Ainouz, and Stephane Canu. Temporal contrastive pretraining for video action recognition. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 662–670, 2020. 3

[26] Zhekun Luo, Devin Guillory, Baifeng Shi, Wei Ke, Fang Wan, Trevor Darrell, and Huijuan Xu. Weakly-supervised action localization with expectation-maximization multi-instance learning. In *European Conference on Computer Vision*, pages 729–745. Springer, 2020. 1

[27] Fan Ma, Linchao Zhu, Yi Yang, Shengxin Zha, Gourab Kundu, Matt Feiszli, and Zheng Shou. Sf-net: Single-frame

supervision for temporal action localization. In *European Conference on Computer Vision*, pages 420–437. Springer, 2020. 1

[28] Han Qiu, Yuchen Ma, Zeming Li, Songtao Liu, and Jian Sun. Borderdet: Border feature for dense object detection. *arXiv preprint arXiv:2007.11056*, 2020. 3

[29] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017. 6

[30] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 3

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 2

[32] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5734–5743, 2017. 7, 8

[33] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1049–1058, 2016. 2, 8

[34] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *arXiv preprint arXiv:1406.2199*, 2014. 6

[35] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 9627–9636, 2019. 3, 6

[36] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 6

[37] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pages 3–19, 2018. 4

[38] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE international conference on computer vision*, pages 5783–5792, 2017. 1, 2, 7, 8

[39] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020. 7

[40] Le Yang, Houwen Peng, Dingwen Zhang, Jianlong Fu, and Junwei Han. Revisiting anchor mechanisms for temporal action localization. *IEEE Transactions on Image Processing*, 29:8535–8548, 2020. 2, 7

[41] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7094–7103, 2019. 2, 7

[42] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9759–9768, 2020. 2

[43] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *European Conference on Computer Vision*, pages 539–555. Springer, 2020. 7

[44] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2914–2923, 2017. 2, 7