TriDet: Temporal Action Detection with Relative Boundary Modeling

Dingfeng Shi*
VRLab, Beihang University, China
shidingfeng@buaa.edu.cn

Yujie Zhong
Meituan Inc.
jaszhong@hotmail.com

Qiong Cao†
JD Explore Academy
mathqiong2012@gmail.com

Lin Ma
Meituan Inc.
forest.linma@gmail.com

Jia Li†
VRLab, Beihang University, China
jiali@buaa.edu.cn

Dacheng Tao
JD Explore Academy
dacheng.tao@gmail.com

Abstract

In this paper, we present a one-stage framework TriDet for temporal action detection. Existing methods often suffer from imprecise boundary predictions due to the ambiguous action boundaries in videos. To alleviate this problem, we propose a novel Trident-head to model the action boundary via an estimated relative probability distribution around the boundary. In the feature pyramid of TriDet, we propose an efficient Scalable-Granularity Perception (SGP) layer to mitigate the rank loss problem of self-attention that takes place in the video features and aggregate information across different temporal granularities. Benefiting from the Trident-head and the SGP-based feature pyramid, TriDet achieves state-of-the-art performance on three challenging benchmarks: THUMOS14, HACS and EPIC-KITCHEN 100, with lower computational costs, compared to previous methods. For example, TriDet hits an average mAP of 69.3% *on THUMOS14, outperforming the previous best by* 2.5%*, but with only* 74.6% *of its latency. The code is released to https://github.com/dingfengshi/TriDet.*

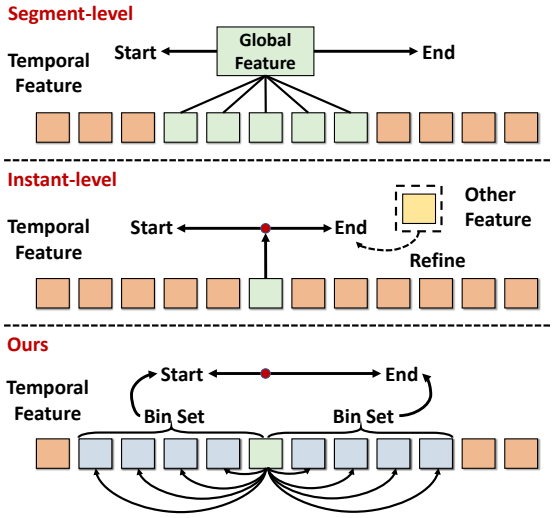


Figure 1. Illustration of different boundary modeling. **Segment-level**: these methods locate the boundaries based on the global feature of a predicted temporal segment. **Instant-level**: they directly regress the boundaries based on a single instant, potentially with some other features. **Ours**: the action boundaries are modeled via an estimated relative probability distribution of the boundary.

1. Introduction

Temporal action detection (TAD) aims to detect all start and end instants and corresponding action categories from an untrimmed video, which has received widespread attention. TAD has been significantly improved with the help of the deep learning. However, TAD remains to be a very challenging task due to some unresolved problems.

A critical problem in TAD is that action boundaries are usually not obvious. Unlike the situation in object detection where there are usually clear boundaries between the objects and the background, the action boundaries in videos can be fuzzy. A concrete manifestation of this is that the instants (*i.e.* temporal locations in the video feature sequence) around the boundary have relatively higher predicted response value from the classifier.

Some previous works attempt to locate the boundaries based on the global feature of a predicted temporal segment [21,22,29,46,51], which may ignore detailed information at each instant. As another line of work, they directly regress the boundaries based on a single instant [32,47], potentially with some other features [20,33,49], which do not consider the relation between adjacent instants (*e.g.* the rel-

*: This work is done during an internship at JD Explore Academy.
†: Corresponding authors.

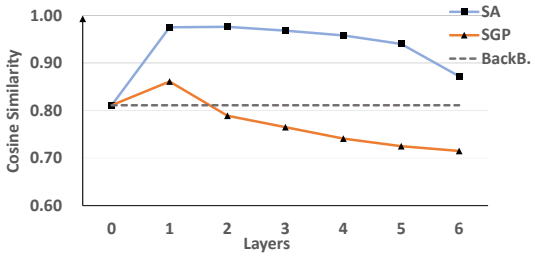


Figure 2. Within the HACS dataset and SlowFast backbone, we statistic the average cosine similarity between features at each instant and the video-level average feature for self-attention and SGP, respectively. We observe that the SA exhibits high similarity, indicating poor discriminability (*i.e.* rank loss problem). In contrast, SGP resolves the issue and exhibits stronger discriminability.

ative probability) around the boundary. How to effectively utilize boundary information remains an open question.

To facilitate localization learning, we posit that the relative response intensity of temporal features in a video can mitigate the impact of video feature complexity and increase localization accuracy. Motivated by this, we propose a one-stage action detector with a novel detection head named Trident-head tailored for action boundary localization. Specifically, instead of directly predicting the boundary offsets based on the center point feature, the proposed Trident-head models the action boundary via an estimated relative probability distribution of the boundary (see Fig. 1). The boundary offset is then computed based on the expected values of neighboring locations (*i.e.* bins).

Apart from the Trident-head, in this work, the proposed action detector consists of a backbone network and a feature pyramid. Recent TAD methods [9, 40, 47] adopt the transformer-based feature pyramid and show promising performance. However, the video features of the video backbone tend to exhibit high similarities between snippets, which is further deteriorated by SA, leading to the rank loss problem [12] (see Fig. 2). Additionally, SA also incurs significant computational overhead.

Fortunately, we discover that the success of the previous transformer-based layers (in TAD) primarily relies on their macro-architecture, namely, how the normalization layer and feed-forward network (FFN) are connected, rather than the self-attention mechanism. We therefore propose an efficient convolutional-based layer, termed Scalable-Granularity Perception (SGP) layer, to alleviate the two abovementioned problems of self-attention. SGP comprises two primary branches, which serve to increase the discrimination of features in each instant and capture temporal information with different scales of receptive fields.

The resultant action detector is termed TriDet. Extensive experiments demonstrate that TriDet surpasses all the previous detectors and achieves state-of-the-art performance across three challenging benchmarks: THUMOS14, HACS and EPIC-KITCHEN 100.

2. Related Work

Temporal action detection. Temporal action detection (TAD) involves localizing and classifying all actions from an untrimmed video. The existing methods can be roughly divided into two categories, namely, two-stage methods and one-stage methods. The two-stage methods [33, 36, 43, 46, 53] split the detection process into two stages: proposal generation and proposal classification. Most of the previous works [8, 13, 19, 21, 22, 26] put emphasis on the proposal generation phrase. Concretely, some works [8, 21, 22] predict the probability of the action boundary and densely match the start and end instants according to the prediction score. Anchor-based methods [13, 19] classify actions from specific anchor windows. However, two-stage methods suffer from a high complexity problem and can not be trained in an end-to-end manner. The one-stage methods do the localization and classification with a single network. Some previous works [20, 44, 45] build this hierarchical architecture with the convolutional network (CNN). However, there is still a performance gap between the CNN-based and the latest TAD methods.

Object detection. Object detection is a twin task of TAD. General Focal Loss [18] transforms bounding box regression from learning Dirac delta distribution to a general distribution function. Some methods [10, 15, 28] use Depthwise Convolution to model network structure and some branched designs [16, 37] show high generalization ability. They are enlightening for the architecture design of TAD.

Transformer-based methods. Inspired by the great success of the Transformer in the field of machine translation and object detection, some recent works [9, 25, 27, 35, 38, 47] adopt the attention mechanism in TAD task, which help improve the detection performance. For example, some works [27, 35, 38] detect the action with the DETR-like Transformer-based decoder [6], which models action instances as a set of learnable. Other works [9, 47] extract a video representation with a Transformer-based encoder. However, most of these methods are based on the *local* behavior. Namely, they conduct attention operation only in a local window, which introduces an inductive bias similar to CNN but with a larger computational complexity and additional limitations (*e.g.* The length of the sequence needs to be pre-padded to an integer multiple of the window size.).

3. Method

Problem definition. We first give a formal definition for TAD task. Specifically, given a set of untrimmed videos $\mathcal{D} = \{\mathcal{V}_i\}_{i=1}^{n}$, we have a set of RGB (and optical flow)

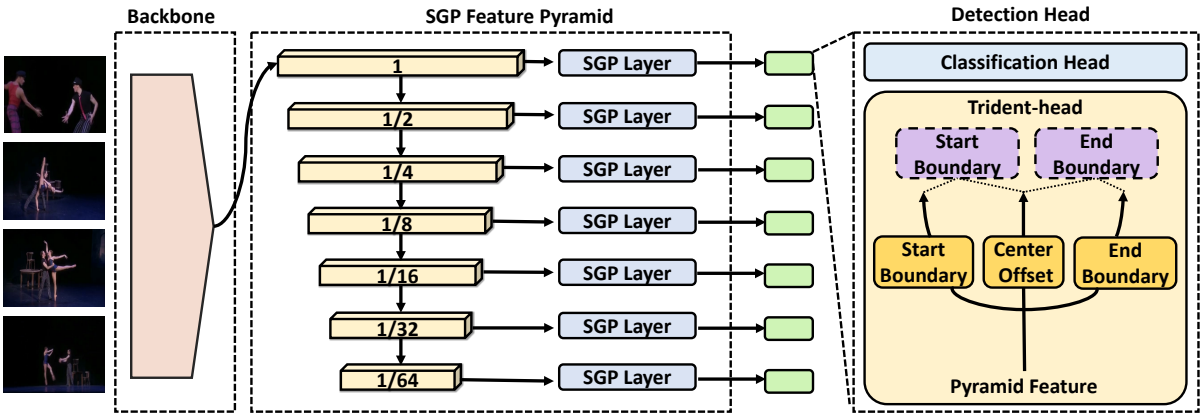


Figure 3. Illustration of TriDet. We build the pyramid features with Scalable-Granularity Perception (SGP) layer. The corresponding features in each level are fed into a shared-weight detection head to obtain the detection result, which consists of a classification head and a Trident-head. The Trident-head estimates the boundary offset based on a relative distribution predicted by three branches: Start Boundary, End Boundary and Center Offset.

temporal visual features $X_i = \{x_t\}_{t=1}^{T}$ from each video $\mathcal{V}_i$, where T corresponds to the number of instants, and K_i segment labels $Y_i = \{s_k, e_k, c_k\}_{k=1}^{K_i}$ with the action segment start instant s_k, the end instant e_k and the corresponding action category c_k. TAD aims at detecting all segments Y_i based on the input feature X_i.

3.1. Method Overview

Our goal is to build a simple and efficient one-stage temporal action detector. As shown in Fig. 3, the overall architecture of TriDet consists of three main parts: a video feature backbone, a SGP feature pyramid, and a boundary-oriented Trident-head. First, the video features are extracted using a pretrained action classification network (*e.g.* I3D [7] or SlowFast [14]). Following that, a SGP feature pyramid is built to tackle actions with various temporal lengths, similar to some recent TAD works [9, 20, 47]. Namely, the temporal features are iteratively downsampled and each scale level is processed with a proposed Scalable-Granularity Perception (SGP) layer (Section 3.2) to enhance the interaction between features with different temporal scopes. Lastly, action instances are detected by a designed boundary-oriented Trident-head (Section 3.3). We elaborate on the proposed modules in the following.

3.2. Feature Pyramid with SGP Layer

The feature pyramid is obtained by first downsampling the output features of the video backbone network several times via max-pooling (with a stride of 2). The features at each pyramid level are then processed using transformer-like layers (e.g. ActionFormer [47]).

Current Transformer-based methods for TAD tasks primarily rely on the macro-architecture of the Transformer (See supplementary material for details), rather than the self-attention mechanisms. Specifically, SA mainly encounters two issues: the rank loss problem across the temporal dimension and its high computational overhead.

Limitation 1: the rank loss problem. The rank loss problem arises because the probability matrix in self-attention (*i.e.* softmax(QK^T)) is *non-negative* and *the sum of each row is 1*, indicating the outputs of SA are *convex combination* for the value feature V. Considering that pure Layer Normalization [2] projects feature onto the unit hyper-sphere in high-dimensional space, we analyze the degree of their distinguishability by studying the maximum angle between features within the instant features. We demonstrate that the maximum angle between features after the *convex combination* is less than or equal to that of the input features, resulting in increasing similarity between features (as outlined in the supplementary material), which can be detrimental to TAD.

Limitation 2: high computational complexity. In addition, the dense pair-wise calculation (between instant features) in self-attention brings a high computational overhead and therefore decreases the inference speed.

The SGP layer. Based on the above discovery, we propose a Scalable-Granularity Perception (SGP) layer to effectively capture the action information and suppress rank loss. The major difference between the Transformer layer and SGP layer is the replacement of the self-attention module with the fully-convolutional module SGP. The successive Layer Normalization [2] (LN) is changed to Group Normalization [41] (GN).

As shown in Fig. 4, SGP contains two main branches: an instant-level branch and a window-level branch. In the

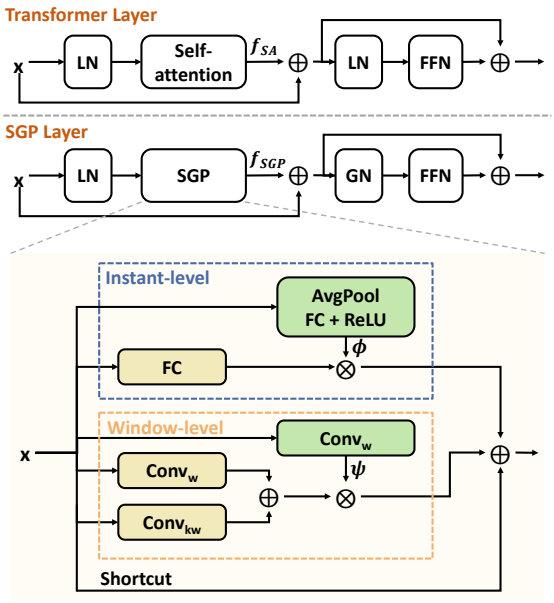


Figure 4. Illustration of the structure of SGP layer. We replace the self-attention and the second Layer Normalization (LN) with SGP and Group Normalization (GN), respectively.

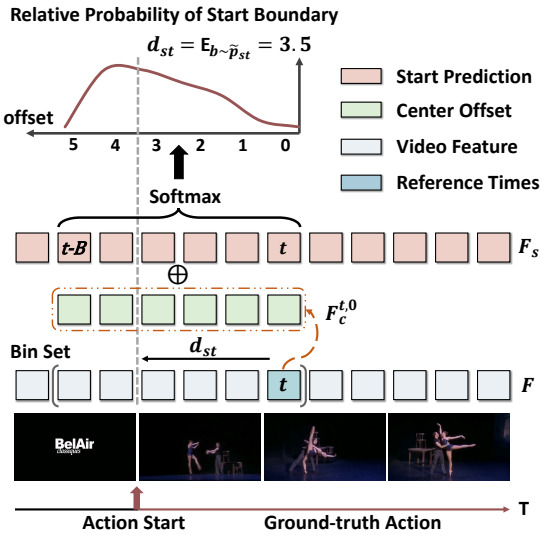


Figure 5. The boundary localization mechanism of Trident-head. We predict the boundary response and the center offset for each instant. At the instant t, the predicted boundary response in neighboring bin set is summed element-wise with the center offset corresponding to the instant t, which is further estimated as the relative boundary distribution. Finally, the offset is computed based on the expected value of the bin.

instant-level branch, we aim to increase the feature discriminability between action and non-action instant by enlarging their feature distance with the video-level average feature. The window-level branch is designed to introduce the semantic content from a wider receptive field with a branch ψ to help dynamically focus on the features of which scale. Mathematically, the SGP can be written as:

$$f_{SGP} = \phi(x)FC(x)+\psi(x)(Conv_w(x)+Conv_{kw}(x))+x, \quad (1)$$

where FC and $Conv_w$ denotes fully-connected layer and the 1-D depth-wise convolution layer [10] over temporal dimension with window size w. As a signature design of SGP, k is a scalable factor aiming at capturing a larger granularity of temporal information. The video-level average feature $\phi(x)$ and branch $\psi(x)$ are given as

$$\phi(x) = ReLU(FC(AvgPool(x))), \quad (2)$$

$$\psi(x) = Conv_w(x), \quad (3)$$

where $AvgPool(x)$ is the average pooling for all features over the temporal dimension. Here, both $\phi(x)$ and $\psi(x)$ perform the element-wise multiplication with the mainstream feature.

The resultant SGP-based feature pyramid can achieve better performance than the transformer-based feature pyramid while being much more efficient.

3.3. Trident-head with Relative Boundary Modeling

Intrinsic property of action boundaries. Regarding the detection head, some existing methods directly regress the temporal length [47] of the action at each instant of the feature and refine with the boundary feature [20, 33], or [21, 22, 46] simply predict an *actionness* score (indicating the probability of being an action). These simple strategies suffer from a problem in practice: imprecise boundary predictions, due to the intrinsic property of actions in videos. Namely, the boundaries of actions are usually not obvious, unlike the boundaries of objects in object detection. Intuitively, a more statistical boundary localization method can reduce uncertainty and facilitate more precise boundaries.

Trident-head. In this work, we propose a boundary-oriented Trident-head to precisely locate the action boundaries based on the relative boundary modeling, *i.e.* considering the relation of features in a certain period and obtaining the relative probability of being a boundary for each instant in that period. The Trident-head consists of three components: a start head, an end head, and a center-offset head, which are designed to locate the start boundary, end boundary, and the temporal center of the action, respectively. The Trident-head can be trained end-to-end with the detector.

Concretely, as shown in Fig. 5, given a sequence of features $F \in \mathcal{R}^{T\times D}$ output from the feature pyramid, we first obtain three feature sequences from the three branches (namely, $F_s \in \mathcal{R}^T$, $F_e \in \mathcal{R}^T$ and $F_c \in \mathcal{R}^{T\times 2\times(B+1)}$), where B is the number of bins for boundary prediction, F_s and F_e characterize the response value for each instant as

the starting or ending point of an action, respectively. In addition, the center-offset head aims at estimating two conditional distributions $P(b_{st}|t)$ and $P(b_{et}|t)$. They represent the probability that each instant (in its set of bins) serves as a boundary when the instant t is the midpoint of an action. Then, we model the boundary distance by combining the outputs of the boundary head and center-offset head:

$$\widetilde{P}_{st} = Softmax(F_s^{[(t-B):t]} + F_c^{t,0}), \tag{4}$$

$$d_{st} = \mathbb{E}_{b\sim\widetilde{P}_{st}}[b] \approx \sum_{b=0}^{B}(b\widetilde{P}_{stb}), \tag{5}$$

where $F_s^{[(t-B):t]} \in \mathcal{R}^{B+1}$, $F_c^{t,0} \in \mathcal{R}^{B+1}$ are the feature of the left adjacent bin set of instant t and the center offsets predicted by instant t only, respectively, and $\widetilde{P}_{st}$ is the *relative probability* which represents the probability of each instant as a start of the action within the bin set. Then, the distance between the instant t and the start instant of action instance d_{st} is given by the expectation of the adjacent bin set. Similarly, the offset distance of the end boundary d_{et} can be obtained by

$$\widetilde{P}_{et} = Softmax(F_e^{[t:(t+B)]} + F_c^{t,1}), \tag{6}$$

$$d_{et} = \mathbb{E}_{b\sim\widetilde{P}_{et}}[b] \approx \sum_{b=0}^{B}(b\widetilde{P}_{etb}) \tag{7}$$

All heads are simply modeled in three layers convolutional networks and share parameters at all feature pyramid levels to reduce the number of parameters.

Combination with feature pyramid. We apply the Trident-head in a pre-defined local bin set, which can be further improved by combining it with the feature pyramid. In this setting, features at each level of the feature pyramid simply share the same small number of bins B (*e.g.* 16) and then the corresponding prediction for each level l can be scaled by 2^{l-1}, which can significantly help to stabilize the training process.

Formally, for an instant in the l-th feature level t^l, TriDet estimates the boundary distance $\hat{d}_{st}^l$ and $\hat{d}_{et}^l$ with the Trident-head described above, then the segments $a = (\hat{s}_t, \hat{e}_t)$ can be decoded by

$$\hat{s}_t = (t - \hat{d}_{st}^l) \times 2^{l-1}, \tag{8}$$

$$\hat{e}_t = (t + \hat{d}_{et}^l) \times 2^{l-1}. \tag{9}$$

Comparison with existing methods that have explicit boundary modeling. Many previous methods improve boundary predictions. We divide them into two broad categories: the prediction based on sampling instants in segments [21, 27, 35] and the prediction based on a single instant. The first category predicts the boundary according to the global feature of the predicted instance segments. They only consider global information instead of detailed information at each instant. The second category directly predicts the distance between an instant and its corresponding boundary based on the instant-level feature [20, 33, 47, 49]. Some of them refine the segment with boundary features [20, 33, 49]. However, they do not take the relation (*i.e.* relative probability of being a boundary) of adjacent instants into account. The proposed Trident-head differs from these two categories and shows superior performance in precise boundary localization.

3.4. Training and Inference

Each layer l of the feature pyramid outputs a temporal feature $F^l \in \mathcal{R}^{(2^{l-1}T)\times D}$, which is then fed to the classification head and the Trident-head for action instance detection. The output of each instant t in feature pyramid layer l is denoted as $\hat{o}_t^l = (\hat{c}_t^l, \hat{d}_{st}^l, \hat{d}_{et}^l)$.

The overall loss function is then defined as follows:

$$\begin{aligned}\mathcal{L} = &\frac{1}{N_{pos}}\sum_{l,t}\mathbb{1}_{\{c_t^l>0\}}(\sigma_{IoU}\mathcal{L}_{cls} + \mathcal{L}_{reg}) \\ &+ \frac{1}{N_{neg}}\sum_{l,t}\mathbb{1}_{\{c_t^l=0\}}\mathcal{L}_{cls},\end{aligned} \tag{10}$$

where σ_{IoU} is the temporal IoU between the predicted segment and the ground truth action instance, and $\mathcal{L}_{cls}$, $\mathcal{L}_{reg}$ is focal loss [23] and IoU loss [34]. N_{pos} and N_{neg} denote the number of positive and negative samples. The term σ_{IoU} is used to reweight the classification loss at each instant, such that instants with better regression (*i.e.* of higher quality) contribute more to the training. Following previous methods [39,47,48], center sampling is adopted to determine the positive samples. Namely, the instants around the center of an action instance are labeled as positive and all the others are considered as negative.

Inference. At inference time, the instants with classification scores higher than threshold λ and their corresponding instances are kept. Lastly, Soft-NMS [4] is applied for the deduplication of predicted instances.

4. Experiments

Datasets. We conduct experiments on four challenging datasets: THUMOS14 [17], ActivityNet-1.3 [5], HACS-Segment [50] and EPIC-KITCHEN 100 [11]. THUMOS14 consists of 20 sport action classes and it contains 200 and 213 untrimmed videos with 3,007 and 3,358 action instances on the training set and test set, respectively. ActivityNet and HACS are two large-scale datasets and they share 200 classes of action. They have 10,024 and 37,613 videos for training, as well as 4,926 and 5,981 videos for test. The EPIC-KITCHEN 100 is a large-scale dataset in first-person vision, which have two sub-tasks: *noun* localization (*e.g.* door) and *verb* localization (*e.g.* open the door).

Table 1. Comparison with the state-of-the-art methods on THUMOS14 dataset. *: TSN backbone. †: Swin Transformer backbone. Others: I3D backbone.

Method	0.3	0.4	0.5	0.6	0.7	Avg.
BMN [21]*	56.0	47.4	38.8	29.7	20.5	38.5
G-TAD [43]*	54.5	47.6	40.3	30.8	23.4	39.3
A2Net [44]	58.6	54.1	45.5	32.5	17.2	41.6
TCANet [33]*	60.6	53.2	44.6	36.8	26.7	44.3
RTD-Net [38]	68.3	62.3	51.9	38.8	23.7	49.0
VSGN [49]*	66.7	60.4	52.4	41.0	30.4	50.2
ContextLoc [53]	68.3	63.8	54.3	41.8	26.2	50.9
AFSD [20]	67.3	62.4	55.5	43.7	31.1	52.0
ReAct [35]*	69.2	65.0	57.1	47.8	35.6	55.0
TadTR [27]	74.8	69.1	60.1	46.6	32.8	56.7
TALLFormer [9]†	76.0	-	63.2	-	34.5	59.2
ActionFormer [47]	82.1	77.8	71.0	59.4	43.9	66.8
TriDet	**83.6**	**80.1**	**72.9**	**62.4**	**47.4**	**69.3**

Table 2. Comparison with the state-of-the-art methods on HACS dataset.

Method	Backbone	0.5	0.75	0.95	Avg.
SSN [52]	I3D	28.8	18.8	5.3	19.0
LoFi [42]	TSM	37.8	24.4	7.3	24.6
G-TAD [43]	I3D	41.1	27.6	8.3	27.5
TadTR [27]	I3D	47.1	32.1	10.9	32.1
BMN [21]	SlowFast	52.5	36.4	10.4	35.8
TALLFormer [9]	Swin	55.0	36.1	11.8	36.5
TCANet [33]	SlowFast	54.1	37.2	11.3	36.8
TriDet	I3D	**54.5**	**36.8**	**11.5**	**36.8**
TriDet	SlowFast	**56.7**	**39.3**	**11.7**	**38.6**

It contains 495 and 138 videos with 67,217 and 9,668 action instances for training and test, respectively. The number of action classes for *noun* and *verb* are 300 and 97.

Evaluation. For all these datasets, only the annotations of the training and validation sets are accessible. Following the previous practice [9, 21, 46, 47], we evaluate on the validation set. We report the mean average precision (mAP) at different intersection over union (IoU) thresholds. For THUMOS14 and EPIC-KITCHEN, we report the IoU thresholds at [0.3:0.7:0.1] and [0.1:0.5:0.1] respectively. For ActivityNet and HACS, we report the result at IoU threshold [0.5, 0.75, 0.95] and the avearge mAP is computed at [0.5:0.95:0.05].

4.1. Implementation Details

TriDet is trained end-to-end with AdamW [31] optimizer. The initial learning rate is set to 10^{-4} for THUMOS14 and EPIC-KITCHEN, and 10^{-3} for ActivityNet and HACS. We detach the gradient before the start boundary head and end boundary head and initialize the CNN weights of these two heads with a Gaussian distribution $\mathcal{N}(0, 0.1)$ to stabilize the training process. The learning rate is updated with Cosine Annealing schedule [30]. We train 40, 23, 19, 15 and 13 epochs for THUMOS14, EPIC-KITCHEN **verb**, EPIC-KITCHEN *noun*, ActivityNet and HACS (containing warmup 20, 5, 5, 10, 10 epochs).

For ActivityNet and HACS, the number of bins B of the Trident-head is set to 12, 14 and the convolution window w is set to 15, 11 and the scale factor k is set to 1.3 and 1.0, respectively. For THUMOS14 and EPIC-KITCHEN, the number of bins B of the Trident-head is set to 16 and the convolution window w is set to 1 and the scale factor k is set to 1.5. We round the scaled windows size and take it up to the nearest odd number for convenience. We conduct our experiments on a single NVIDIA A100 GPU.

Table 3. Comparison with the state-of-the-art methods on EPIC-KITCHEN dataset. *V.* and *N.* denote the *verb* and *noun* sub-tasks, respectively.

	Method	0.1	0.2	0.3	0.4	0.5	Avg.
V.	BMN [21]	10.8	8.8	8.4	7.1	5.6	8.4
	G-TAD [43]	12.1	11.0	9.4	8.1	6.5	9.4
	ActionFormer [47]	26.6	25.4	24.2	22.3	19.1	23.5
	TriDet	**28.6**	**27.4**	**26.1**	**24.2**	**20.8**	**25.4**
N.	BMN [21]	10.3	8.3	6.2	4.5	3.4	6.5
	G-TAD [43]	11.0	10.0	8.6	7.0	5.4	8.4
	ActionFormer [47]	25.2	24.1	22.7	20.5	17.0	21.9
	TriDet	**27.4**	**26.3**	**24.6**	**22.2**	**18.3**	**23.8**

4.2. Main Results

THUMOS14. We adopt the commonly used I3D [7] as our backbone feature and Tab. 1 presents the results. Our method achieves an average mAP of 69.3%, outperforming all previous methods including one-stage and two-stage methods. Notably, our method also achieves better performance than recent Transformer-based methods [9, 27, 33, 35, 47], which demonstrates that the simple design can also have impressive results.

HACS. For the HACS-segment dataset, we conduct experiments based on two commonly used features: the official I3D [7] feature and the SlowFast [14] feature. As shown in Tab. 2, our method achieves an average mAP of 36.8% with the official features. It is the state-of-the-art and outperforms the previous best model TadTR by about 4.7% in average mAP. We also show that changing the backbone to SlowFast can further boost performance, resulting in a 1.8% increase in average mAP, which indicates that our method can benefit from a much more advanced backbone network.

EPIC-KITCHEN. On this dataset, following all previous methods, SlowFast is adopted as the backbone feature.

Table 4. Comparison with the state-of-the-art methods on ActivityNet-1.3 dataset.

Method	Backbone	0.5	0.75	0.95	Avg.
PGCN [46]	I3D	48.3	33.2	3.3	31.1
ReAct [35]	TSN	49.6	33.0	8.6	32.6
BMN [21]	TSN	50.1	34.8	8.3	33.9
G-TAD [43]	TSN	50.4	34.6	9.0	34.1
AFSD [20]	I3D	52.4	35.2	6.5	34.3
TadTR [27]	TSN	51.3	35.0	9.5	34.6
TadTR [27]	R(2+1)D	53.6	37.5	10.5	36.8
VSGN [49]	I3D	52.3	35.2	8.3	34.7
PBRNet [24]	I3D	54.0	35.0	9.0	35.0
TCANet+BMN [33]	TSN	52.3	36.7	6.9	35.5
TCANet+BMN [33]	SlowFast	54.3	**39.1**	**8.4**	**37.6**
TALLFormer [9]	Swin	54.1	36.2	7.9	35.6
ActionFormer [47]	R(2+1)D	**54.7**	37.8	**8.4**	36.6
TriDet	R(2+1)D	**54.7**	**38.0**	**8.4**	**36.8**

The method of our main comparison is ActionFormer [47], which has demonstrated promising performance in EPIC-KITCHEN 100 dataset. We present the results in Tab. 3. Our method shows a significant improvement in both subtasks: *verb* and *noun*, and achieves 25.4% and 23.8% average mAP, respectively. Note that our method outperforms ActionFormer with the same features by a large margin (1.9% and 1.9% average mAP in *verb* and *noun*, respectively). Moreover, our method achieves state-of-the-art performance on this challenging dataset.

ActivityNet. For the ActivityNet v1.3 dataset, we adopt the TSP R(2+1)D [1] as our backbone feature. Following previous methods [9, 20, 27, 33, 47], the video classification score predicted from the UntrimmedNet is adopted to multiply with the final detection score. Tab. 4 presents the results. Our method still shows a promising result: TriDet outperform the second best model [47] with the same feature, only worse than TCANet [33] which is a two-stage method and using the SlowFast as the backbone feature which is not available now.

4.3. Ablation Study

In this section, we mainly conduct the ablation studies on the THUMOS14 dataset.

Main components analysis. We demonstrate the effectiveness of our proposed components in TriDet: SGP layer and Trident-head. To verify the effectiveness of our SGP layer, we use a baseline feature pyramid used by [20, 47] to replace our SGP layer. The baseline consists of two 1D-convolutional layers and shortcut. The window size of convolutional layers is set to 3 and the number of channels of the intermediate feature is set to the same dimension as the intermediate dimension in the FFN in our SGP layer. All other hyperparameters (*e.g.* number of the pyramid layers, etc.) are set to the same as our framework.

Table 5. Analysis of the Effectiveness of three main components on THUMOS14.

Method	SA	SGP	Trident	0.3	0.5	0.7	Avg.
1				77.3	65.2	40.0	62.1
2	√			82.1	71.0	43.9	66.8
3		√		**83.6**	71.7	45.8	68.3
4		√	√	**83.6**	**72.9**	**47.4**	**69.3**

Table 6. Analysis of computation cost on THUMOS14. Main: All parts of the model except the detection head. *: Our method with a normal instant-level regression head.

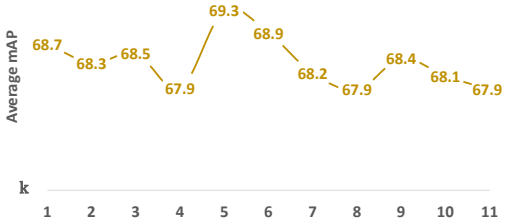
Method	mAP			GMACs			Latency
	0.3	0.7	Avg.	Main	Head	All	(ms)
ActionFormer	82.1	43.9	66.8	30.8	**14.4**	45.3	224
TriDet*	**83.6**	45.8	68.3	**14.5**	**14.4**	**28.9**	**145**
TriDet	**83.6**	**47.4**	**69.3**	**14.5**	29.1	43.7	167

As depicted in Tab. 5, compared with the baseline model we implement (Row 1), the SGP layer brings a 6.2% absolute improvement in the average mAP. Secondly, we compare the SGP with the previous state-of-the-art method, ActionFormer, which adopts a self-attention mechanism in a sliding window behavior [3] with window size 7 (Row 2). We can see our SGP layer still has 1.5% improvement in average mAP, demonstrating that the convolutional network can also have excellent performance in TAD task. Besides, we compare our Trident-head with the normal instant-level regression head, which regresses the boundary distance for each instant. We can see that the Trident-head improves the average mAP by 1.0%, and the mAP improvement is more obvious in the case of high IoU threshold (*e.g.* 1.6% average mAP improvement in IoU 0.7).

Computational complexity. We compare the computational complexity and latency of TriDet with the recent ActionFormer [47], which brings a large improvement to TAD by introducing the Transformer-based feature pyramid.

As shown in Tab. 6, we divide the detector into two parts: the main architecture and the detection heads (*e.g.* classification head and regression head). We report the GMACs for each part and the inference latency (average over five times) on THUMOS14 dataset using an input with the shape 2304×2048, following the [47]. We also report our results using the Trident-head and the normal regression head, respectively. First, from the first row, we see that GMACs of our main architecture with SGP layer is only 47.1% of

(a) The choice of w (k=1)



(b) The choice of k (w=1)

Figure 6. Effectiveness of window size w and k.

Table 7. Analysis of the number of feature pyramid layers.

#Levels	Bin	0.3	0.7	Avg.	Bin	0.3	0.7	Avg.
1		70.1	15.3	44.5	512	74.2	25.9	53.5
2		77.9	27.8	57.1	256	78.0	29.7	58.0
3		79.8	37.7	61.8	128	80.6	37.1	62.5
4	16	82.1	42.6	66.1	64	82.7	39.0	64.7
5		82.9	45.7	68.1	32	82.7	44.7	67.4
6		**83.6**	**47.4**	**69.3**	16	**83.6**	**47.4**	**69.3**
7		83.4	46.2	68.9	8	82.7	46.8	68.2

the ActionFormer (14.5 versus 30.8), and the overall latency is only 65.2% (146ms versus 224ms), but TriDet still outperforms Actionformer by 1.5% average mAP, which shows that our main architecture is much better than the local Transformer-based method. Besides, we further evaluate our method with Trident-head. The experimental result shows that our framework can be improved by the Trident-head which further brings 1.0% average mAP improvement and the GMACs is still 1.6G smaller than ActionFormer, and the latency is still only 74.6% of it, proving the high efficiency of our method.

Ablation on the window size in SGP layer. In this section, we study the effectiveness of the two hyper-parameters related to the window size in the SGP layer. Firstly, we fix $k = 1$ and vary w. Secondly, we fix the value of $w = 1$ and change k. Finally, we present the results in Fig. 6 on THUMOS14 datasets. We find that different choices of w and k produce stable results on both datasets. The optimal values are $w = 1$, $k = 5$ for THUMOS14.

The effectiveness of feature pyramid level. To study the effectiveness of the feature pyramid and its relation with the number of Trident-head bin set, we start the ablation from the feature pyramid with 16 bins and 6 levels. We conduct two sets of experiments: a fixed number of bins or a scaled number of bins for each level in the feature pyramid. As shown in Tab. 7, we can see that the detection performance rises as the number of layers increases. With fewer levels (*i.e.* level less than 3), more bins bring better performance. That is because the fewer the number of levels, the more bins are needed to predict the action with a long duration (*i.e.* higher resolution at the highest level). We achieve the best result with a level number of 6.

Table 8. Analysis of the number of bins.

Bin	THUMOS14				HACS			
	0.3	0.5	0.7	Avg.	0.5	0.75	0.95	Avg.
4	82.9	71.5	46.3	68.1	55.7	32.3	4.7	33.3
8	83.5	**72.9**	46.3	69.0	56.2	38.4	11.2	38.0
10	82.8	71.8	46.2	68.1	56.2	38.5	11.1	37.9
12	**83.6**	72.3	46.2	68.5	56.3	38.4	11.1	38.0
14	83.4	72.6	45.6	68.3	**56.7**	**39.3**	**11.7**	**38.6**
16	**83.6**	**72.9**	**47.4**	**69.3**	56.5	38.6	11.1	38.1
20	**83.6**	71.7	45.8	68.3	56.3	38.6	11.1	38.0

Ablation on the number of bins. In this section, we present the ablation results for the choice of the number of bins on the THUMOS14 and HACS datasets in Tab. 8. We observe the optimal value is obtained at 16 and 14 on the THUMOS14 and the HACS, respectively. We also find that a small bin value leads to significant performance degradation on HACS but not on THUMOS14. That is because the THUMOS14 dataset aims at detecting a large number of action segments from a long video and a small bin value can meet the requirements, but on HACS, there are more actions with long duration, thus a larger number of bins is needed.

5. Conclusion

In this paper, we aim at improving the temporal action detection task with a simple one-stage convolutional-based framework TriDet with relative boundary modeling. Experiments conducted on THUMOS14, HACS, EPIC-KITCHEN and ActivityNet demonstrate a high generalization capability of our method, which achieves state-of-the-art performance on the first three datasets and comparable results on ActivityNet. Extensive ablation studies are conducted to verify the effectiveness of each proposed component.

Acknowledgement. This work is supported by the National Natural Science Foundation of China under Grant 62132002.

References

[1] Humam Alwassel, Silvio Giancola, and Bernard Ghanem. TSP: Temporally-sensitive pretraining of video encoders for localization tasks. In *Int. Conf. Comput. Vis.*, 2021. 7

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 3

[3] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. 7

[4] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Int. Conf. Comput. Vis.*, 2017. 5

[5] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2015. 5

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Eur. Conf. Comput. Vis.* Springer, 2020. 2

[7] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 3, 6

[8] Guo Chen, Yin-Dong Zheng, Limin Wang, and Tong Lu. Dcan: improving temporal action detection via dual context aggregation. In *AAAI*, 2022. 2

[9] Feng Cheng and Gedas Bertasius. Tallformer: Temporal action localization with long-memory transformer. *Eur. Conf. Comput. Vis.*, 2022. 2, 3, 6, 7

[10] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2, 4

[11] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *Int. J. Comput. Vis.*, 2022. 5

[12] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *Int. Conf. Machine Learning*, 2021. 2

[13] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *Eur. Conf. Comput. Vis.*, 2016. 2

[14] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Int. Conf. Comput. Vis.*, 2019. 3, 6

[15] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[16] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[17] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes, 2014. 5

[18] Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *Adv. Neural Inform. Process. Syst.*, 2020. 2

[19] 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. 2

[20] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning salient boundary feature for anchor-free temporal action localization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 3, 4, 5, 6, 7

[21] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Int. Conf. Comput. Vis.*, 2019. 1, 2, 4, 5, 6, 7

[22] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *Eur. Conf. Comput. Vis.*, 2018. 1, 2, 4

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, 2017. 5

[24] Qinying Liu and Zilei Wang. Progressive boundary refinement network for temporal action detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020. 7

[25] Xiaolong Liu, Song Bai, and Xiang Bai. An empirical study of end-to-end temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20010–20019, 2022. 2

[26] Xiaolong Liu, Yao Hu, Song Bai, Fei Ding, Xiang Bai, and Philip HS Torr. Multi-shot temporal event localization: a benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12596–12606, 2021. 2

[27] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Shiwei Zhang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *IEEE Trans. Image Process.*, 2022. 2, 5, 6, 7

[28] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[29] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1

[30] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *Int. Conf. Learn. Represent.*, 2017. 6

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Int. Conf. Learn. Represent.*, 2019. 6

[32] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 563–579, 2018. 1

[33] Zhiwu Qing, Haisheng Su, Weihao Gan, Dongliang Wang, Wei Wu, Xiang Wang, Yu Qiao, Junjie Yan, Changxin Gao, and Nong Sang. Temporal context aggregation network for temporal action proposal refinement. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 4, 5, 6, 7

[34] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 5

[35] Dingfeng Shi, Yujie Zhong, Qiong Cao, Jing Zhang, Lin Ma, Jia Li, and Dacheng Tao. React: Temporal action detection with relational queries. In *Eur. Conf. Comput. Vis.*, 2022. 2, 5, 6, 7

[36] Deepak Sridhar, Niamul Quader, Srikanth Muralidharan, Yaoxin Li, Peng Dai, and Juwei Lu. Class semantics-based attention for action detection. In *Int. Conf. Comput. Vis.*, 2021. 2

[37] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017. 2

[38] Jing Tan, Jiaqi Tang, Limin Wang, and Gangshan Wu. Relaxed transformer decoders for direct action proposal generation. In *Int. Conf. Comput. Vis.*, 2021. 2, 6

[39] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Int. Conf. Comput. Vis.*, 2019. 5

[40] Yuetian Weng, Zizheng Pan, Mingfei Han, Xiaojun Chang, and Bohan Zhuang. An efficient spatio-temporal pyramid transformer for action detection. In *Eur. Conf. Comput. Vis.*, 2022. 2

[41] Yuxin Wu and Kaiming He. Group normalization. In *Eur. Conf. Comput. Vis.*, 2018. 3

[42] Mengmeng Xu, Juan Manuel Perez Rua, Xiatian Zhu, Bernard Ghanem, and Brais Martinez. Low-fidelity video encoder optimization for temporal action localization. *Adv. Neural Inform. Process. Syst.*, 2021. 6

[43] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 2, 6, 7

[44] Le Yang, Houwen Peng, Dingwen Zhang, Jianlong Fu, and Junwei Han. Revisiting anchor mechanisms for temporal action localization. *IEEE Trans. Image Process.*, 2020. 2, 6

[45] Min Yang, Guo Chen, Yin-Dong Zheng, Tong Lu, and Limin Wang. Basictad: an astounding rgb-only baseline for temporal action detection. *arXiv preprint arXiv:2205.02717*, 2022. 2

[46] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *Int. Conf. Comput. Vis.*, 2019. 1, 2, 4, 6, 7

[47] Chenlin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *Eur. Conf. Comput. Vis.*, 2022. 1, 2, 3, 4, 5, 6, 7

[48] 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 *IEEE Conf. Comput. Vis. Pattern Recog.*, 2020. 5

[49] Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In *Int. Conf. Comput. Vis.*, 2021. 1, 5, 6, 7

[50] Hang Zhao, Zhicheng Yan, Lorenzo Torresani, and Antonio Torralba. Hacs: Human action clips and segments dataset for recognition and temporal localization. *arXiv preprint arXiv:1712.09374*, 2019. 5

[51] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *Eur. Conf. Comput. Vis.*, 2020. 1

[52] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *ICCV*, 2017. 6

[53] Zixin Zhu, Wei Tang, Le Wang, Nanning Zheng, and Gang Hua. Enriching local and global contexts for temporal action localization. In *Int. Conf. Comput. Vis.*, 2021. 2, 6