

ReAct: Temporal Action Detection with Relational Queries

Dingfeng Shi^{1*}, Yujie Zhong², Qiong Cao^{3†},
Jing Zhang⁴, Lin Ma², Jia Li^{1†}, and Dacheng Tao^{3,4}

¹ State Key Laboratory of Virtual Reality Technology and Systems, School of
Computer Science and Engineering, Beihang University

² Meituan Inc

³ JD Explore Academy

⁴ The University of Sydney

Abstract. This work aims at advancing temporal action detection (TAD) using an encoder-decoder framework with action queries, similar to DETR, which has shown great success in object detection. However, the framework suffers from several problems if directly applied to TAD: the insufficient exploration of inter-query relation in the decoder, the inadequate classification training due to a limited number of training samples, and the unreliable classification scores at inference. To this end, we first propose a relational attention mechanism in the decoder, which guides the attention among queries based on their relations. Moreover, we propose two losses to facilitate and stabilize the training of action classification. Lastly, we propose to predict the localization quality of each action query at inference in order to distinguish high-quality queries. The proposed method, named ReAct, achieves the state-of-the-art performance on THUMOS14, with much lower computational costs than previous methods. Besides, extensive ablation studies are conducted to verify the effectiveness of each proposed component. The code is available at <https://github.com/sssste/React>.

1 Introduction

Temporal action detection (TAD) has been actively studied because of the deep learning era. Inspired by the advance of one-stage object detectors [22,32,10], many recent works focus on one-stage action detectors [18], which show excellent performance while having a relatively simple structure. On the other hand, DETR [4], which tackles object detection in a Transformer encoder-decoder framework, attracted considerable attention. In this work, we propose a novel one-stage action detector ReAct that is based on such a learning paradigm. Inspired by DETR, ReAct models action instances as a set of learnable action queries. These action queries are fed into the decoder as inputs, and they iteratively attend to the output features of the encoder as well as update their

* This work is done during an internship at JD Explore Academy.

† Corresponding authors: mathqiong2012@gmail.com and jiali@buaa.edu.cn.

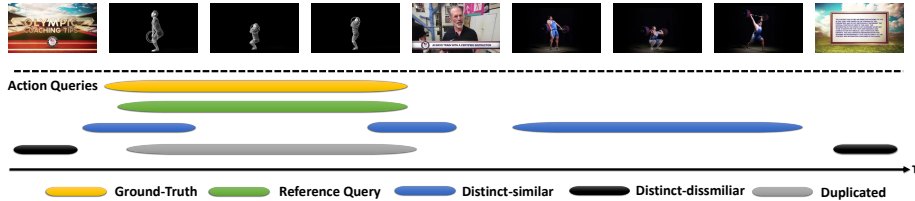


Fig. 1. The relation of queries. We choose the green one as the reference query, and the queries in a different relation to it are labeled with different colors. Only the Distinct-similar pair (Blue ones) will be kept for attention computation.

predictions. The action classification and localization are then predicted by two simple feedforward neural nets.

However, the DETR-like methods suffer from several problems when applied to TAD task. First, the inter-query relations are not fully explored by the self-attention in the decoder, which is performed densely over all the queries. Second, DETR-like methods may suffer from the inadequate training of action classification since the number of positive training samples for the classifier is relatively small compared to anchor-based/free methods. Moreover, when multiple queries fire for the same action instance at inference, queries with higher classification scores may not necessarily have better temporal localization. In the following, we elaborate on these problems and introduce the proposed methods to alleviate them in three aspects: attention mechanism, training losses, and inference.

The decoder in DETR-like methods applies the self-attention over the action queries to capture their relations, which can not fully explore the complex relations among queries. In this work, we denote the action queries that are responsible for localizing different action instances of similar or same action classes as *distinct-similar* queries, and those detecting different action classes as *distinct-dissimilar* queries. For the queries that fire for the same action instance, we regard them as *duplicate* queries. In this work, we propose a novel **attention mechanism**, named Relational Attention with IoU Decay (RAID), to explicitly handle these three types of query relations in the decoder. As Fig. 1 shows, RAID focuses on the communication among distinct-similar queries (since they are expected to provide more informative signals) and blocks the attention between distinct-dissimilar and duplicate queries. Furthermore, the proposed IoU decay encourages the duplicate queries to be slightly different from each other to enable a more diverse prediction.

Another problem is that a DETR-like approach may have a relatively low classification accuracy due to inadequate classification training. This is because the positive training samples for the classification of DETR-like methods are much fewer than those of the anchor-free methods. Namely, for DETR-like methods, the number of positives per input clip is only the same as the ground truth actions because of the bipartite-matching-based label assignment. To address this problem, we propose two **training losses**, codenamed Action Classification Enhancement (ACE) losses, to facilitate the classification learning. The first loss

ACE-*enc* is applied to the input features of the encoder and is designed to reduce the intra-class variance and inter-class similarity of action instances. This loss explicitly improves the discriminability of video features regarding acting classes, thus benefiting the classification. Meanwhile, a ACE-*dec* loss is proposed as the classification loss in the decoder, which considers both the predicted segments and the ground-truth segments for action classification. It increases the training samples and generates a stable learning signal for the classifier.

Lastly, the action queries are redundant by design compared to the actual action instances. At inference, it is a common situation where multiple action queries fire for the same action instance. Hence, it is important to focus on precise action localization queries. Nonetheless, the classification score is deficient in measuring the temporal localization quality. As a result, we propose a Segment Quality to predict the localization quality of each action query **at inference**, such that the more high-quality queries can be distinguished.

To summarize, we make the following contributions in this work:

- We approach temporal action detection using a DETR-like framework and identify three limitations of such method when directly applied to TAD.
- We propose the relational attention with IoU decay, the action classification enhancement losses, and the segment quality prediction, which alleviate the identified problems from the perspectives of attention mechanism, training losses, and network inference, respectively.
- Experiments on two action detection benchmarks demonstrate the superiority of ReAct: it achieves the state-of-the-art performance on THUMOS14, with much lower computational costs than previous methods. Extensive ablation studies are conducted to verify the effectiveness of each component.

2 Related Work

Temporal action detection. Temporal action detection (TAD) aims to detect all the start and end timestamps and the corresponding action types based on the video stream information. The existing methods can be roughly divided into two categories: two-stage methods and one-stage methods. Two-stage methods [11,28,38,43,19,21,12,17] split the detection task into two subtasks: proposal generation and proposal classification. Concretely, some methods [21,17,19] generate the proposals by predicting the probability of the start point and endpoint of the action and then selecting the proposal segments according to prediction score. In addition, PGCN [43] considers the relationship between proposals, then refines and classifies the proposals by Graph Convolutional Network. These two-stage methods can perform better by combining proposal generation networks and proposal classification networks. However, they can not be trained in an end-to-end manner and are computationally inefficient. To solve the above problems, some one-stage methods [20,6,18,26,37] are proposed. Some works [6,18,40] try to adapt to the high variance of the action duration by constructing a temporal feature pyramid, while Liu *et al.* [23] propose to dynamically sample temporal features by learnable parameters. These one-stage methods reduce the complexity

of the models, which are more computationally friendly. In this work, we mainly follow the one-stage fashion and the deformable convolution design [9,47,23] to build a efficient action detector, which will be detailed in the Section 3.

Attention-based model. Attention-based models [33] have achieved great success in machine translation and been extended to the field of computer vision [25,1,24,39,44,8] in recent years. The attention module computes a soft weight dynamically for a set of points at runtime. Concretely, DETR [4] proposes a Transformer-based image detection paradigm. It learns decoder input features shared by all input videos and detects a fixed number of outputs. Deformable DETR [47] improves DETR by reducing the number of pairs to be computed in the attention module with learnable spatial offsets. Liu *et al.* [23] propose an end-to-end framework for TAD based on Deformable DETR. This type of training paradigm is highly efficient and fast in prediction. However, there is still a performance gap between these methods and the latest methods in TAD [23,43]. Our work is built on DETR-like workflows. In contrast to the above work, our approach suppresses the flow of invalid information by constricting a computational subset for the attention module, which improves performance effectively.

Contrastive learning. Contrastive learning [7] is a method that has been widely used in unsupervised learning. NCE [13] mines data features by distinguishing between data and noise. Info-NCE [27] is proposed to extract representations from high-dimensional data with a probabilistic contrastive loss. Lin *et al.* [18] leverage contrastive learning to help network identify action boundaries. Inspired by these works, we use contrastive learning to extract a global common representation of action categories and enlarge the feature distance between action segments and noise segments.

3 Method

Problem definition. This work focuses on the problem of temporal action detection (TAD). Specifically, given a set of untrimmed videos $\mathcal{D} = \{\mathcal{V}_i\}_{i=1}^n$. A set of $\{X_i, Y_i\}$ can be extracted from each video \mathcal{V}_i , where $X_i = \{x_t\}_{t=1}^T$ corresponds to the image (and optical flow) features of T snippets and $Y_i = \{m_k, d_k, c_k\}_{k=1}^{K_i}$ is K_i segment labels for the video V_i with the action segment midpoint time m_k , the action duration d_k and the corresponding action category c_k . Temporal action detection aims at predicting all segments Y_i based on the input feature X_i .

Method overview. Motivated by DETR [4], we approach the problem of TAD by an encoder-decoder framework based on the transformer network. As Fig. 3 shows, the overall architecture of ReAct contains three parts: a video feature extractor, an action encoder, and an action decoder. First, video clip features are extracted from each RGB frame by using the widely-used 3D-CNN (*e.g.*, TSN [35] or I3D [5]). The optical flow features are also extracted using TVL1 optical flow algorithm [42]. Following that, a 1-D conv layer is used to modify the feature dimension of the clip features. The output features are then passed

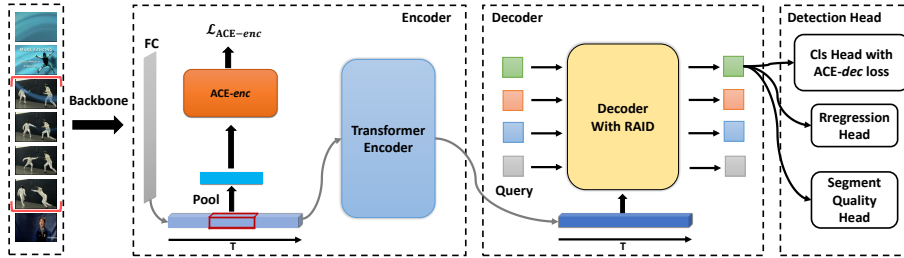


Fig. 2. Illustration of the proposed framework. The video feature is extracted by a pretrained backbone, followed by a fully-connected layer to project the feature, and is additionally supervised by the AEC-Enc loss. After enhancement by the Transformer encoder, the features are fed into the decoder and attended by L_q action queries in the decoder. The classification head is trained with the proposed ACE-Dec loss.

to the action encoder, which is a L_E -layer transformer network. The encoded clip features serve as one of the inputs to the action decoder. The decoder is a L_D -layer transformer, and it differs from the encoder in two aspects. It has action queries (which are learnable embeddings) as inputs, and the queries attend the encoder outputs in each layer of the decoder, known as Cross-attention. Essentially, ReAct maps action instances as a set of action queries. The action queries are transformed by the decoder into output embeddings which are used for both action classification and temporal localization by separate feed-forward neural nets. The details of the encoder structure are provided in the appendix.

At training, following previous works[4,47,23], the Hungarian Algorithm [16] is applied to assign labels to the action queries. The edge weight is defined by the summation of the segment IoU, the probability of classification, and the L1 norm between two coordinates. Based on the matching, ReAct applies several losses to the action queries, including the action classification loss and temporal segment regression loss.

Limitations of DETR-like methods for TAD. DETR-like methods may suffer from several problems when applied to TAD task. First, the decoder performs the self-attention densely over all the queries, which causes the inter-query relations not to be sufficiently explored. Second, compared with anchor-based/free methods, DETR-like methods may have issues in deficit training of action classification attributed to relatively smaller number of positive training samples for the classifier. Third, queries with higher classification scores may not be reliable due to multiple queries firing for the same action instance at inference.

In this work, we mitigate these problems in three aspects: (1) We propose the Relational Attention with IoU Decay which allows each action query to attend to others in the decoder based on their relations; (2) We design two Action Classification Enhancement losses to enhance the action classification learning at the encoder and decoder, respectively; (3) We introduce a Segment Quality to predict the localization quality of each action query at inference to compensate the deficiency of classification score at inference. We elaborate on these three aspects in the following.

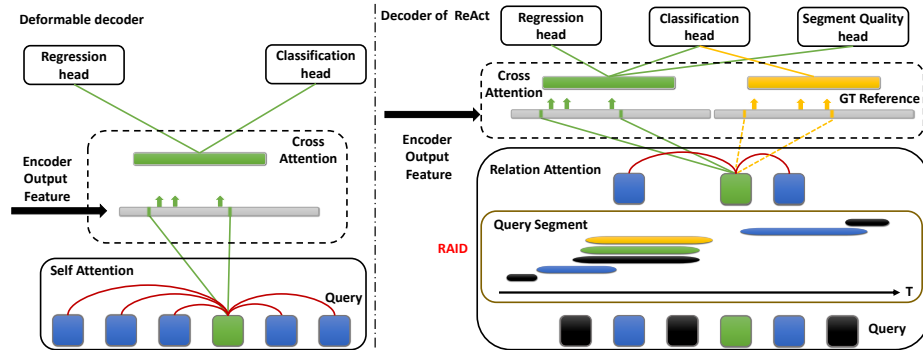


Fig. 3. Illustration of our decoder. **Left:** plain deformable decoder. Each query performs the attention operation with all other query features and sample segment features from the encoder output. **Right:** decoder of ReAct. Each query only attends to specific queries based on the inter-query relation. Besides, the ground-truth segment provides an additional loss to further supervise the classification head. Note that for clarity, the LayerNorm, FFN, and residual connection are not shown in the figure (see appendix for detailed network structure).

3.1 Relational Attention with IoU Decay

To better explore the inter-query relation in the decoder, we present the Relational Attention with IoU Decay (RAID) which replaces the self-attention in the transformer decoder. Below, we describe the proposed method in detail.

Relational attention. As a recap, we define three types of queries with respect to an action query q_i , which are differentiated by their relations to q_i . *Distinct-similar* queries are the queries that try to detect different action instances but of similar (or same) action class to q_i . *Distinct-dissimilar* queries are those which try to detect different action instances and of dissimilar action class to q_i . *Duplicate* queries are the queries that try to detect the same action instance as q_i . Intuitively, we anticipate that attending to distinct-dissimilar queries does not provide informative signals to q_i , since they focus on different action classes, and the relation between action classes may not be a reliable cue for detecting actions. On the contrary, attending to distinct-similar queries can benefit the query q by gathering some background information and cues around q_i . For example, some actions may occur multiple times in a clip, and attending to each other can increase the confidence of the detection. Moreover, duplicate queries only repeat the prediction as q_i , so they bring no extra information and should be ignored in the attention for q_i .

To find the distinct-similar queries for a query q_i , we consider two properties, namely, high context similarity and low temporal overlap. To measure context similarity, we compute a similarity matrix $A \in \mathbb{R}^{L_q \times L_q}$ (L_q is the number of

queries) based on the query features, where each element represents the cosine similarity of two queries. Then the query-pair set \mathcal{E}_{sim} is constructed by

$$\mathcal{E}_{sim} = \{(i, j) | A[i, j] - \gamma > 0\}, \quad (1)$$

where $\gamma \in [-1, 1]$ is a preset similarity threshold. To identify the queries having low temporal overlap with q , a natural strategy is using the Interaction of Union (IoU) in the time domain, which measures the overlap between two temporal segments. Therefore, we compute a pair-wise IoU matrix $B \in \mathbb{R}^{L_q \times L_q}$ for the reference segments and construct a query-pair set \mathcal{E}_{IoU} as follows:

$$\mathcal{E}_{IoU} = \{(i, j) | B[i, j] - \tau < 0\}, \quad (2)$$

where i and j denote the i -th and j -th queries respectively, and $\tau \in [0, 1]$ is a preset IoU threshold. As shown in Fig. 3, this simple strategy removes the segments which have large temporal overlap. We can then define the distinct-similar query-pair set \mathcal{E} by combining \mathcal{E}_{sim} , \mathcal{E}_{IoU} and the query itself \mathcal{E}_s . The definition is given as follow:

$$\mathcal{E} = (\mathcal{E}_{IoU} \setminus \mathcal{E}_{sim}) \cup \mathcal{E}_s. \quad (3)$$

For a query q_i and its distinct query-pair set \mathcal{E}_i , the key and value features can be written as $K_i = concatenate(\{k_j | (i, j) \in \mathcal{E}_i\})$ and $V_i = concatenate(\{v_j | (i, j) \in \mathcal{E}_i\})$. Then, the query features q_i are updated by

$$q'_i = a_i V_i^T, \quad (4)$$

where the attention weight a_i is

$$a_i = Softmax_K(q_i K_i^T). \quad (5)$$

Note that by considering both the context similarity and temporal overlap, the proposed relational attention successfully preserves the communication between q_i and useful queries while blocking that between uninformative ones.

IoU decay. Apart from relational attention, we introduce a further improvement by handling duplicate queries. Namely, we propose a regularization, termed as IoU decay, which is added to the network optimization. It is given as

$$\omega_d = \frac{1}{2} \sum_{i=1}^{L_q} \sum_{j=1}^{L_q} IoU(s_i, s_j). \quad (6)$$

During the detector training, it penalizes the IoU between queries, such that duplicate queries can be diversified and different from each other, which can increase the probability of obtaining a more precise localization for the target action instance.

3.2 Action Classification Enhancement

To combat the issue of the inadequate learning of classification when applying the DETR-like methods to TAD, we propose two Action Classification Enhancement (ACE) losses to boost the classification performance.

ACE-enc loss. We aim to enhance the features with respect to action classification in the phase of encoder by enlarging the similarity of inter-class action instances and reducing the variance between intra-class action instances. We posit that explicitly increasing the discriminability of the features on the action detection dataset in an early stage can also benefit the final action classification. Specifically, we optimize the input features of the encoder using contrastive loss.

The positive and negative action instance pairs are constructed as follows. For a given ground-truth action segment s_g and its category c_g in a video v_i , we choose its positive instances by sampling the action segments of the same category c_g from either the same or different videos. As for its negative instances, we choose them from two different sources: (1) segments of action categories different from c_g and (2) segments that are completely inside the ground-truth segment, but their IoU is less than a specific threshold ξ .

For a given segment s , we denote $x \in \mathbb{R}^{T \times D'}$ and $\tilde{x} \in \mathbb{R}^{T \times D}$ as the pre-trained video feature and feature further projected by a fully-connected layer l (i.e. $\tilde{x} = l(x)$), respectively. Then, the segment feature after temporal RoI pooling [38] can be denoted as $f = RoI(\tilde{x}, s) \in \mathbb{R}^D$. With the above definitions, the loss $\mathcal{L}_{ACE-enc}$ is given by

$$\mathcal{L}_{ACE-enc} = -\log \frac{\exp(f^T f_p)}{\sum_{j \in \mathcal{D}} \exp(f^T f_j)}, \quad (7)$$

where f_p is a positive segment of f and \mathcal{D} is the index of k random negative instances as well as a positive instance.

ACE-dec loss. Anchor-based/free methods treat all (or multiple) the temporal locations within the ground truth action segment as positives (i.e., belonging to an action class rather than backgrounds) for training the action classifiers, whereas DETR-like methods have much fewer positives due to the bipartite matching at label assignment. We, therefore, propose the ACE-dec loss to train the action classifiers.

As Fig. 3 (right) shows, in the training phase, an additional positive training sample is fed to the action classifiers for each query segment (i.e., the green one) matched with a ground-truth action instance. The additional positive is obtained by feeding the ground-truth segment (i.e., the yellow one) as a normal query segment to the cross-attention layer. The details of the cross-attention layer are described in the supplementary material.

Concretely, every decoder layer is attached a ACE-dec loss which is given by

$$\mathcal{L}_{ACE-dec} = \mathcal{L}_{foc}^q + \mathbb{1}_{y \neq 0}[\mathcal{L}_{foc}^{gt}], \quad (8)$$

where \mathcal{L}_{foc}^q and \mathcal{L}_{foc}^{gt} is the sigmoid Focal Loss [22] for the query and ground-truth classification loss respectively. Note that, only the queries that are matched to ground-truth segments will contribute the ground-truth classification loss.

3.3 Segment Quality Prediction

To remedy the problem that classification score is unreliable for selecting the best query among a set of duplicate queries, we propose a Segment Quality to predict the localization quality of each action query at inference for distinguishing high-quality queries. The proposed segment quality prediction considers both the midpoint of the segment as well as its temporal coverage on the action instance.

Concretely, given a predicted segment s_q and its query feature f_q , we define $(\zeta_1, \zeta_2) = \phi(f_q)$, where ϕ is a single fully-connected layer and $\zeta_1, \zeta_2 \in [0, 1]$. Then, a final quality value ζ is defined by $\zeta = \zeta_1 \cdot \zeta_2$. Segment Quality is supervised by a two-dimensional vector composing of the offset of the predicted midpoint and its ground truth for localizing the midpoint precisely, and the IoU between the predicted segment and its closest ground-truth segment for accurate temporal localization and coverage. The overall loss is given by

$$\mathcal{L}_\zeta = \sum \left| \phi(f_q) - \left(\exp\left(-\frac{1}{l_{gt}}|m_q - m_{gt}|\right), \text{IoU}(s_q, s_{gt}) \right) \right|_1, \quad (9)$$

where m_q is the midpoint of the predicted segment, and m_{gt}, l_{gt} are the midpoint and length of the ground-truth segment, respectively. At inference, ζ is multiplied with the classification score of the segment.

3.4 Training Losses

At training, based on the label assignment by the Hungarian Algorithm, ReAct is trained by the total loss as follow:

$$\mathcal{L} = \mathcal{L}_{ACE-enc} + \mathcal{L}_{ACE-dec} + \mathcal{L}_\zeta + \mathcal{L}_{reg}. \quad (10)$$

Here, \mathcal{L}_{reg} is the commonly used regression loss for TAD which regresses the midpoint and the duration of the detected segments using the summation of L1 distance and the generalized IoU distance [29] for the matched pair. We define each objective as follows:

$$\begin{aligned} \mathcal{L}_{reg} &= \frac{1}{N_{c_{gt} \neq \emptyset}} \sum_{j \in L_q} \mathbb{1}_{c_{gt}^{(j)} \neq \emptyset} [\gamma_1 \mathcal{L}_{L1}^{(j)} + \gamma_2 \mathcal{L}_{gIoU}^{(j)}], \\ \mathcal{L}_{L1}^{(j)} &= |m_{gt}^{(j)} - m^{(j)}| + |d_{gt}^{(j)} - d^{(j)}|, \\ \mathcal{L}_{gIoU}^{(j)} &= 1 - gIoU(s_{gt}^{(j)}, s^{(j)}), \end{aligned} \quad (11)$$

where $s^{(j)} = (m^{(j)}, d^{(j)})$ is j-th detected segment represented by midpoint and the duration. $c_{gt}^{(j)}$ is a set of the ground-truth segments that s^j is matched and

$N_{c_{gt} \neq \emptyset}$ is the number of segments in $c_{gt}^{(j)}$. $s_{gt}^{(j)} = (m_{gt}^{(j)}, d_{gt}^{(j)})$ is the matched ground-truth segment of s^j and $s_{gt}^{(j)} \in c_{gt}^{(j)}$. In addition, we follow the segment refinement fashion [47,23] to predict detections in each decoder layer, each of which will be updated by summing with the upper layer segment and re-normalizing it. In this way, each layer provides auxiliary classification loss \mathcal{L}'_{cls} and regression loss \mathcal{L}'_{reg} , which further helps the network training.

4 Experiment

We conduct experiments on two challenging datasets: THUMOS14 [14] and ActivityNet-1.3 [3].

4.1 Implementation Details

Architecture details. For THUMOS14, we set $L_q = 40$, $L_E = 2$, $L_D = 4$ for the number of queries, encoder layer and decoder layer, respectively. Each deformable attention module samples 4 temporal offsets for computing the attention. The hidden layer dimension of the feedforward network is set to 1024, and the other hidden feature dimension in the intermediate of the network is all set to 256. The pair-wise IoU threshold τ and feature similarity threshold γ in ACE module are set to 0.2 and 0.2, respectively. For ActivityNet-1.3, we set $L_q = 60$, $L_E = 3$, $L_D = 4$, $\tau = 0.9$, $\gamma = -0.2$. We sample 4 temporal offsets for the deformable module. For more implementation details including feature extraction and training details, please refer to the supplementary material.

Optimization parameters and inference. We train the ReAct with AdamW optimizer with a batch size of 16. The learning rate is set to 2×10^{-4} and 1×10^{-4} for THUMOS14 and ActivityNet-1.3 respectively. ReAct is trained for 15 epochs on THUMOS14 and 35 epochs on ActivityNet-1.3. At inference, the classification head output is activated by sigmoid. Then all the predictions will be processed with Soft-NMS[2] to remove the redundant and low-quality segments.

4.2 Main Results

On THUMOS14 (see Tab. 1), our ReAct achieves superior performance and suppresses the state-of-the-art one-stage and two-stage methods in mAP at different thresholds. In particular, ReAct achieves 55.0% in the average mAP, which outperforms TadTR by a large margin, namely about the 9.4% absolute improvement. Besides, we compare the computational performance during testing. We adopt Floating-point operations per second (FLOPs) per clip following the previous works. [23,48]. We can see that our model has FLOPs of 0.68G, which is 0.06G lower than TadTr and much lower than all the other methods. Note that the FLOPs we report in the table does not include the computation of video feature extraction with backbone. For methods like AFSD, which fine-tunes the

Table 1. Comparison with the state-of-the-art methods on THUMOS14 dataset. We report the mean Average Precision (mAP) in different thresholds and the floating-point operations (FLOPs, G).

Type	Method	0.3	0.4	0.5	0.6	0.7	Avg.	FLOPs
Two-stage	BSN[21]	53.5	45.0	36.9	28.4	20.0	36.8	3.4
	BMN[19]	56.0	47.4	38.8	29.7	20.5	38.5	171.0
	G-TAD[38]	54.5	47.6	40.3	30.8	23.4	39.3	639.8
	TAL[6]	53.2	48.5	42.8	33.8	20.8	39.8	-
	TCANet[28]	60.6	53.2	44.6	36.8	26.7	44.3	-
	CSA+BMN[30]	64.4	58.0	49.2	38.2	27.8	47.5	-
	P-GCN[43]	63.6	57.8	49.1	-	-	-	4.4
	RTD-Net[31]	68.3	62.3	51.9	38.8	23.7	49.0	-
	VSGN[45]	66.7	60.4	52.4	41.0	30.4	50.2	-
	ContextLoc[48]	68.3	63.8	54.3	41.8	26.2	50.9	3.1
One-stage	SSAD[20]	43.0	35.0	24.6	-	-	-	-
	SSN[41]	51.9	41.0	29.9	-	-	-	-
	A2Net[40]	58.6	54.1	45.5	32.5	17.2	41.6	30.4
	AFSD[18]	67.3	62.4	55.5	43.7	31.1	52.0	5.1
	TadTr[23]	62.4	57.4	49.2	37.8	26.3	46.6	0.75
	ReAct	69.2	65.0	57.1	47.8	35.6	55.0	0.68

backbone and does feature extraction during testing, we ignore the computation of feature extraction and only report the FLOPs afterward.

On ActivityNet-1.3, our method achieves comparable results to the state-of-the-art (See Tab. 2). The ReAct outperforms the other DETR-based methods while enjoying a low computational cost (*e.g.*, 0.38G). The Actionness and Anchor-based methods tend to have higher performance compared with the DETR-based methods. One possible reason is that the DETR-based methods take learnable query embedding as input, which is video-agnostic and only keeps statistical information. For a dataset with a large variance in action time, a query feature has to take both long and short action into account (See appendix for more details) and is prone to conflicts.

4.3 Ablation Study

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

Main components. We demonstrate the effectiveness of three proposed components in ReAct: RAID, ACE, and Segment Quality. From Tab. 3 (row 2 and row 5), we can see that compared with the plain deformable decoder layer, our RAID brings about a 3.7% absolute improvement in the average mAP, proving the effectiveness of the module by introducing the relational attention based on

Table 2. Comparison with the state-of-the-art methods on ActivityNet-1.3 dataset.

Type	Method	0.5	0.75	0.95	Avg.	FLOPs(G)
Actionness	BSN[21]	46.5	30.0	8.0	28.2	-
	SSN[41]	43.2	28.7	5.6	28.3	-
	BMN[19]	50.1	34.8	8.3	33.9	45.6
	G-TAD[38]	50.4	34.6	9.0	34.1	45.7
	BU-TAL[46]	43.5	33.9	9.2	34.3	-
	VSGN[45]	52.3	35.2	8.3	34.7	-
Anchor-based	TAL[6]	38.2	18.3	1.3	20.2	-
	PGCN[43]	48.3	33.2	3.3	31.1	5.0
	TCANet[28]	52.3	36.7	6.9	35.5	-
	AFSD[18]	52.4	35.2	6.5	34.3	15.3
DETR-based	RTD-Net[31]	47.2	30.7	8.6	30.8	-
	TadTr[23]	49.1	32.6	8.5	32.3	0.38
	ReAct	49.6	33.0	8.6	32.6	0.38

Table 3. Ablation study on three main components.

Method	RAID	ACE	SQ	0.3	0.4	0.5	0.6	0.7	Avg.
Our Base				66.6	59.2	49.7	38.0	25.0	47.7
		✓	✓	66.6	61.5	53.7	43.4	31.2	51.3
	✓		✓	67.0	62.6	54.4	44.0	32.2	52.1
	✓	✓		69.1	63.3	54.2	43.5	31.0	52.2
	✓	✓	✓	69.2	65.0	57.1	47.8	35.6	55.0

the defined distinct-similar, distinct-dissimilar and duplicated queries. Besides, from rows 4 and 5 of the Table, we see our ACE improves the average mAP performance by 2.9%, which shows its effectiveness by designing new losses to enhance classification learning. Finally, from rows 3 and 5, the proposed Segment Quality achieves 2.8% improvements in average mAP, which effectively estimates the predicted segments’ quality at inference.

Analysis of RAID. We study the effect of two hyperparameters γ and τ in Section 3.1 for thresholding the similarity scores and IoU values when constructing the distinct similar and dissimilar query sets. First, we set $\tau = 1$ and plot the average mAP when varying γ . From Fig. 5(a) we see that as γ increases, the mAP exhibits an increase followed by a decrease, with a peak at $\tau = 0.2$. Besides, we observe that the detection performance shows greater volatility as τ decreases further (*i.e.*, $\tau < -0.1$). Intuitively, smaller τ leads to more irrelevant query pairs communicating, thus introducing greater uncertainty. Next, we study the effect of the choice of τ by fixing $\gamma = 0.2$. From Fig. 5(b) we observe a

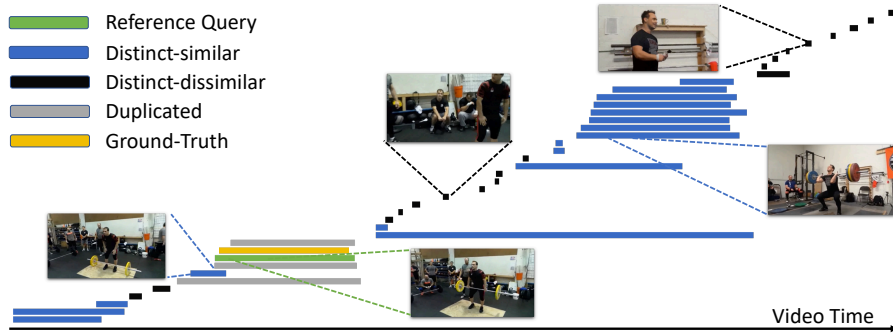


Fig. 4. Visualization of the queries for a test video in THUMOS14. Some example frames are shown for the queries, and we can see that many distinct-dissimilar queries correspond to noises (i.e. not actions).

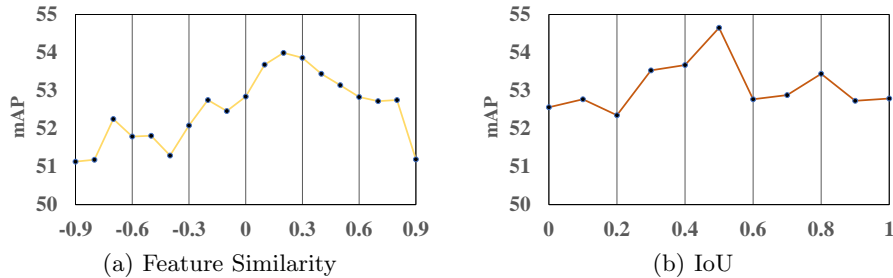


Fig. 5. (a) is visualization of the choice of the hyperparameter γ , with $\tau = 1$; (b) is visualization of the choice of the hyperparameter τ , with $\gamma = 0.2$

similar trend with the figure for similarity as τ changes, and the optimal value is obtained at 0.5. Notice that the smaller τ is, the more queries will be excluded, and when $\tau = 0$, only those that do not overlap will be retained. Intuitively, partially overlapped queries tend to be in the vicinity of the target query, which helps to perceive the information near the boundary. A visualized example of the queries is presented in Fig. 4.3 to illustrate the work of RAID.

Analysis of ACE. We analyze the effect of ACE-enc loss in the following aspects: the construction of contrastive pairs, where to apply ACE-enc loss and training losses. First, we study how contrastive pairs affect performance. In particular, to form the positive segment pairs, we randomly choose segments of the same category from either the same video or different videos, denoted by **S1** and **S2**, respectively. As for negative pairs, there are two ways: segment pairs belonging to different action classes (denoted by **N1**), and segment pairs that one completely includes the other, but their IoU is less than a threshold (denoted by **N2**), as described in 3.2. Tab. 4 presents the results using different combinations of positive and negative pairs. In Tab. 4, we see that **N2** play a

Table 4. Comparison of different settings of ACE module.

Module	Setting	0.3	0.4	0.5	0.6	0.7	Avg.
ACE-enc	No Contrastive	68.1	63.4	55.0	46.0	32.8	53.1
	{S1,S2} + {N1}	68.3	63.4	55.4	46.2	33.9	53.4
	{S1,S2} + {N2}	69.7	64.6	55.7	45.6	33.8	53.9
	{S1,S2} + {N1,N2}	69.7	64.5	56.6	45.9	34.7	54.3
	{S1} + {N1,N2}	69.1	64.4	56.3	46.2	34.6	54.1
	Before Transformer Enc.	69.7	64.3	56.1	46.4	34.2	54.1
After Transformer Enc.	66.4	61.2	53.3	43.4	32.0	51.2	
ACE-dec	\mathcal{L}_{foc}^q Only	67.5	62.6	53.9	43.3	33.2	52.1
	\mathcal{L}_{foc}^{gt} Only	66.1	61.1	53.6	44.2	30.9	51.2
	$\mathcal{L}_{foc}^q + \mathcal{L}_{foc}^{gt}$	68.3	63.4	55.4	46.2	33.9	53.4

more important role in training than **N1** (*e.g.*, average mAP 53.9 versus 53.4), and merging them can gain further promotion (*i.e.*, 54.3).

Secondly, we study the effect of where to apply ACE-enc loss. We mainly consider two positions: before the transformer encoder and after it. We train a single fully connected layer for the former to enhance the video features. For the latter, we use the encoder output. The experimental results show that a single fully connected layer is much better than a complex transformer encoder. Intuitively, after encoder processing, the features on each frame already contain local temporal information, therefore, the pooled segment features can not represent the action precisely, leading to inaccurate convergence.

Finally, to go deeper into the ACE-dec loss, we conducted three experiments: query classification loss only, ground-truth classification loss only, and the complete ACE-dec loss. For the case of the ground-truth classification loss only, we still predict and match the ground-truth segment with the input query feature, which provides the matched query position and reference ground-truth segment. However, we only update the network with ground-truth classification loss \mathcal{L}_{foc}^{gt} . From the Tab. 4, neither \mathcal{L}_{foc}^q nor \mathcal{L}_{foc}^{gt} can perform well, but when we combine them together, the result are significantly better (*e.g.*, 53.4 versus 51.2).

5 Conclusion

In this work, we consider the task of temporal action detection and propose a novel one-stage action detector ReAct based on a DETR-like learning framework. Three limitations of such a method when directly applied to TAD are identified. We propose the relational attention with IoU decay, the action classification enhancement losses, and the segment quality prediction and handle those issues from three aspects: attention mechanism, training losses, and network inference, respectively. ReAct achieves the state-of-the-art performance with much lower

computational costs than previous methods on THUMOS14. Extensive ablation studies are also conducted to demonstrate the effectiveness of each proposed component. In the future, we plan to include the video feature extractor in the action detection training to improve the performance further.

Acknowledgement. This work is supported by the Major Science and Technology Innovation 2030 "New Generation Artificial Intelligence" key project (No. 2021ZD0111700), National Natural Science Foundation of China under Grant 62132002, Grant 61922006 and Grant 62102206.

Supplementary Material For ReAct

A Encoder in Detail

To be self-contained, we provide the detailed structure of the encoder. As Fig. 1 shows, for the input video feature $F \in \mathbb{R}^{T \times D}$, a local offset position and attention weight will be predicted with two fully-connected layers, respectively. For each time step, feature are then sampled according to the K offsets with linear interpolation. The sampled features are weighted by the attention weights and summed up to produce the updated frame feature for the corresponding time step.

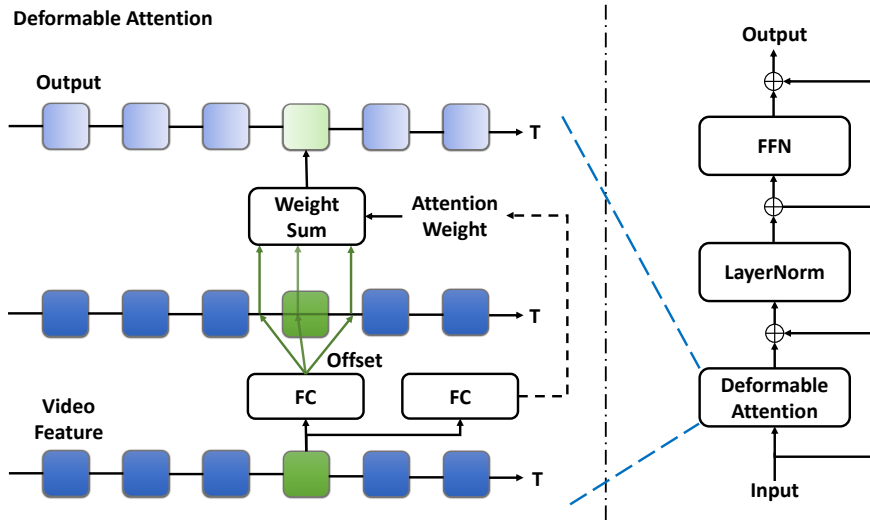


Fig. 1. Illustration of the encoder.

B Decoder in Detail

To help understand our method better, we introduce the decoder in detail. There are two attention modules in the decoder: the proposed relational attention module and a cross-attention module.

In the following, we elaborate on the deformable cross-attention module. As Fig. 2 showed, reference segment, offset position, and attention weights are predicted by three fully-connect layers, based on which the network samples sparse features to update the query feature at each decoder layer. There are two main differences in the deformable attention module between the encoder

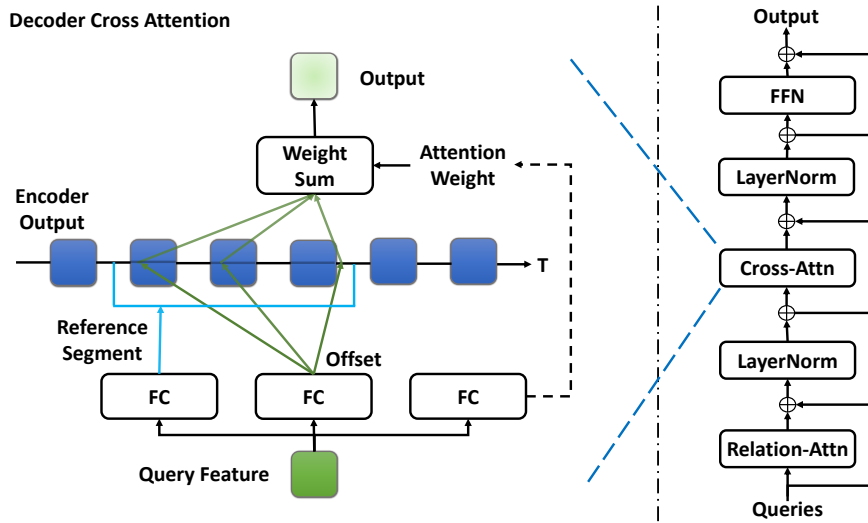


Fig. 2. Illustration of the Deformable Cross Attention module.

and decoder. First, the inputs and outputs are different. The input of the cross-attention in the decoder is the queries, while the input of the encoder is video features. The second difference is the reference segment. In the encoder, temporal offsets for each frame are sampled only around that frame. Whereas for the cross-attention module, an additional reference segment length is predicted for each query feature, and the offsets are normalized such that the sampled frames are always in the segment.

C Architecture and Training Detail

For THUMOS14, following [38], we use the TSN network [35] pre-trained on Kinetics [15] to extract features, which are then down-sampled every five frames. Each video feature is cropped in sequence with a window size 256, and two adjacent windows will have 192 overlapped features with a stride rate of 0.25. In the training phase, ground-truth cut by windows over 75% duration will be kept, and all empty windows without any ground-truth are removed. Finally, all ground-truth coordinates are re-normalized to the window coordinate system. we set $L_q = 40$, $L_E = 2$, $L_D = 4$ for the number of queries, encoder layer and decoder layer, respectively. Each deformable attention module will sample 4 temporal offsets for computing the attention. The hidden layer dimension of the feedforward network is set to 1024, and the other hidden feature dimension in the intermediate of the network is all set to 256. The pair-wise IoU threshold τ and feature similarity threshold γ in ACE module are set to 0.5 and 0.2, respectively. For ActivityNet, the pre-trained TSN network by Xiong *et al.* [36] is adopted to extract features. Then each video feature downsamples every 16 frames, and the resultant feature will be rescaled to 100 snippets using linear interpolation. We only do video-level detection instead of window-level. We set

the $L_q = 60$, $L_E = 3$, $L_D = 4$. We sample 4 temporal offsets for the deformable module. The dimension of hidden features is set to 256, and we set the pair-wise IoU threshold τ and feature similarity threshold γ to 0.9 and -0.2, respectively. Following previous works [38,43,46,40], we combined the Untrimmed-Net video-level classification results [34] with our classification score.

D Visualization of the Classification Loss

To further demonstrate the effect of ACE-*dec* loss, we compute the classification loss for the Activitynet-1.3 test set. As Fig. 3 shows, compared to the Focal Loss, the ACE-*dec* loss improves not only the convergence speed but also the accuracy.

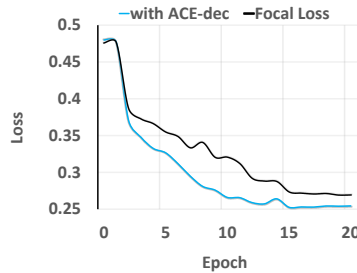


Fig. 3. Visualization of the test classification loss. We record the testing loss with or without ACE-*dec* loss during training

References

1. Arnab, A., Deghani, M., Heigold, G., Sun, C., Lučić, M., Schmid, C.: Vivit: A video vision transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6836–6846 (2021)
2. Bodla, N., Singh, B., Chellappa, R., Davis, L.S.: Soft-nms—improving object detection with one line of code. In: Proceedings of the IEEE international conference on computer vision. pp. 5561–5569 (2017)
3. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 961–970 (2015)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6299–6308 (2017)

6. Chao, Y.W., Vijayanarasimhan, S., Seybold, B., Ross, D.A., Deng, J., Sukthankar, R.: Rethinking the faster r-cnn architecture for temporal action localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1130–1139 (2018)
7. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
8. Chen, X., Cao, Q., Zhong, Y., Zhang, J., Gao, S., Tao, D.: Dearkd: Data-efficient early knowledge distillation for vision transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12052–12062 (2022)
9. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 764–773 (2017)
10. Feng, C., Zhong, Y., Gao, Y., Scott, M.R., Huang, W.: Tood: Task-aligned one-stage object detection. arXiv preprint arXiv:2108.07755 (2021)
11. Gao, J., Chen, K., Nevatia, R.: Ctap: Complementary temporal action proposal generation. In: Proceedings of the European conference on computer vision (ECCV). pp. 68–83 (2018)
12. Gao, J., Yang, Z., Chen, K., Sun, C., Nevatia, R.: Turn tap: Temporal unit regression network for temporal action proposals. In: Proceedings of the IEEE international conference on computer vision. pp. 3628–3636 (2017)
13. Gutmann, M., Hyvärinen, A.: Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics. pp. 297–304. JMLR Workshop and Conference Proceedings (2010)
14. Jiang, Y.G., Liu, J., Roshan Zamir, A., Toderici, G., Laptev, I., Shah, M., Sukthankar, R.: THUMOS challenge: Action recognition with a large number of classes. <http://csrcv.ucf.edu/THUMOS14/> (2014)
15. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
16. Kuhn, H.W.: The hungarian method for the assignment problem. *Naval research logistics quarterly* **2**(1-2), 83–97 (1955)
17. Lin, C., Li, J., Wang, Y., Tai, Y., Luo, D., Cui, Z., Wang, C., Li, J., Huang, F., Ji, R.: Fast learning of temporal action proposal via dense boundary generator. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 11499–11506 (2020)
18. Lin, C., Xu, C., Luo, D., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y.: Learning salient boundary feature for anchor-free temporal action localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3320–3329 (2021)
19. Lin, T., Liu, X., Li, X., Ding, E., Wen, S.: Bmn: Boundary-matching network for temporal action proposal generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3889–3898 (2019)
20. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. In: Proceedings of the 25th ACM international conference on Multimedia. pp. 988–996 (2017)
21. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: Bsn: Boundary sensitive network for temporal action proposal generation. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)

22. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2980–2988 (2017)
23. Liu, X., Wang, Q., Hu, Y., Tang, X., Bai, S., Bai, X.: End-to-end temporal action detection with transformer. arXiv preprint arXiv:2106.10271 (2021)
24. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)
25. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv preprint arXiv:2106.13230 (2021)
26. Long, F., Yao, T., Qiu, Z., Tian, X., Luo, J., Mei, T.: Gaussian temporal awareness networks for action localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 344–353 (2019)
27. Van den Oord, A., Li, Y., Vinyals, O.: Representation learning with contrastive predictive coding. arXiv e-prints pp. arXiv-1807 (2018)
28. Qing, Z., Su, H., Gan, W., Wang, D., Wu, W., Wang, X., Qiao, Y., Yan, J., Gao, C., Sang, N.: Temporal context aggregation network for temporal action proposal refinement. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 485–494 (2021)
29. Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 658–666 (2019)
30. Sridhar, D., Quader, N., Muralidharan, S., Li, Y., Dai, P., Lu, J.: Class semantics-based attention for action detection. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13739–13748 (2021)
31. Tan, J., Tang, J., Wang, L., Wu, G.: Relaxed transformer decoders for direct action proposal generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13526–13535 (2021)
32. Tian, Z., Shen, C., Chen, H., He, T.: Fcos: Fully convolutional one-stage object detection. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9627–9636 (2019)
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. *Advances in neural information processing systems* **30** (2017)
34. Wang, L., Xiong, Y., Lin, D., Van Gool, L.: Untrimmednets for weakly supervised action recognition and detection. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 4325–4334 (2017)
35. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks for action recognition in videos. *IEEE transactions on pattern analysis and machine intelligence* **41**(11), 2740–2755 (2018)
36. Xiong, Y., Wang, L., Wang, Z., Zhang, B., Song, H., Li, W., Lin, D., Qiao, Y., Van Gool, L., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2016. arXiv preprint arXiv:1608.00797 (2016)
37. Xu, H., Das, A., Saenko, K.: R-c3d: Region convolutional 3d network for temporal activity detection. In: Proceedings of the IEEE international conference on computer vision. pp. 5783–5792 (2017)
38. Xu, M., Zhao, C., Rojas, D.S., Thabet, A., Ghanem, B.: G-tad: Sub-graph localization for temporal action detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10156–10165 (2020)

39. Xu, Y., Zhang, Q., Zhang, J., Tao, D.: Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems* **34** (2021)
40. Yang, L., Peng, H., Zhang, D., Fu, J., Han, J.: Revisiting anchor mechanisms for temporal action localization. *IEEE Transactions on Image Processing* **29**, 8535–8548 (2020)
41. Yu, T., Ren, Z., Li, Y., Yan, E., Xu, N., Yuan, J.: Temporal structure mining for weakly supervised action detection. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 5522–5531 (2019)
42. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime tv-l 1 optical flow. In: *Joint pattern recognition symposium*. pp. 214–223. Springer (2007)
43. Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., Gan, C.: Graph convolutional networks for temporal action localization. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 7094–7103 (2019)
44. Zhang, Q., Xu, Y., Zhang, J., Tao, D.: Vitaev2: Vision transformer advanced by exploring inductive bias for image recognition and beyond. *arXiv preprint arXiv:2202.10108* (2022)
45. Zhao, C., Thabet, A.K., Ghanem, B.: Video self-stitching graph network for temporal action localization. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 13658–13667 (2021)
46. Zhao, P., Xie, L., Ju, C., Zhang, Y., Wang, Y., Tian, Q.: Bottom-up temporal action localization with mutual regularization. In: *European Conference on Computer Vision*. pp. 539–555. Springer (2020)
47. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159* (2020)
48. Zhu, Z., Tang, W., Wang, L., Zheng, N., Hua, G.: Enriching local and global contexts for temporal action localization. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 13516–13525 (2021)