PointTAD: Multi-Label Temporal Action Detection with Learnable Query Points

Jing Tan [1*] **Xiaotong Zhao** [2] **Xintian Shi** [2] **Bing Kang** [2] **Limin Wang** [1,3†]
[1]State Key Laboratory for Novel Software Technology, Nanjing University
[2]Platform and Content Group (PCG), Tencent [3]Shanghai AI Lab
jtan@smail.nju.edu.cn, {davidxtzhao,tinaxtshi,bingkang}@tencent.com, lmwang@nju.edu.cn

Abstract

Traditional temporal action detection (TAD) usually handles untrimmed videos with small number of action instances from a single label (e.g., ActivityNet, THUMOS). However, this setting might be unrealistic as different classes of actions often co-occur in practice. In this paper, we focus on the task of multi-label temporal action detection that aims to localize all action instances from a multi-label untrimmed video. Multi-label TAD is more challenging as it requires for fine-grained class discrimination within a single video and precise localization of the co-occurring instances. To mitigate this issue, we extend the sparse query-based detection paradigm from the traditional TAD and propose the multi-label TAD framework of PointTAD. Specifically, our PointTAD introduces a small set of learnable query points to represent the important frames of each action instance. This point-based representation provides a flexible mechanism to localize the discriminative frames at boundaries and as well the important frames inside the action. Moreover, we perform the action decoding process with the Multi-level Interactive Module to capture both point-level and instance-level action semantics. Finally, our PointTAD employs an end-to-end trainable framework simply based on RGB input for easy deployment. We evaluate our proposed method on two popular benchmarks and introduce the new metric of detection-mAP for multi-label TAD. Our model outperforms all previous methods by a large margin under the detection-mAP metric, and also achieves promising results under the segmentation-mAP metric. Code is available at https://github.com/MCG-NJU/PointTAD.

1 Introduction

With the increasing amount of video resources on the Internet, video understanding is becoming one of the most important topics in computer vision. Temporal action detection (TAD) [52, 23, 21, 4, 10, 19, 37] has been formally studied on traditional benchmarks such as THUMOS [15], ActivityNet [14], and HACS [50]. However, the task seems impractical because their videos almost contain non-overlapping actions from a single category: 85% videos in THUMOS are annotated with single action category. As a result, most TAD methods [23, 21, 45, 5, 35] simply cast this TAD problem into sub-problems of action proposal generation and global video classification [42]. In this paper, we shift our playground to the more complex setup of multi-label temporal action detection, which aims to detect all action instances from multi-labeled untrimmed videos. Existing works [9, 16, 40, 8] in this field formulate the problem as a dense prediction task and perform multi-label classification in a frame-wise manner. Consequently, these methods are weak in localization and fail to provide the instance-level detection results (i.e., the starting time and ending time of each instance). In analogy to image instance segmentation [22], we argue that it is necessary to redefine multi-label TAD as a

[*]Work is done during internship at Tencent PCG. [†]Corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

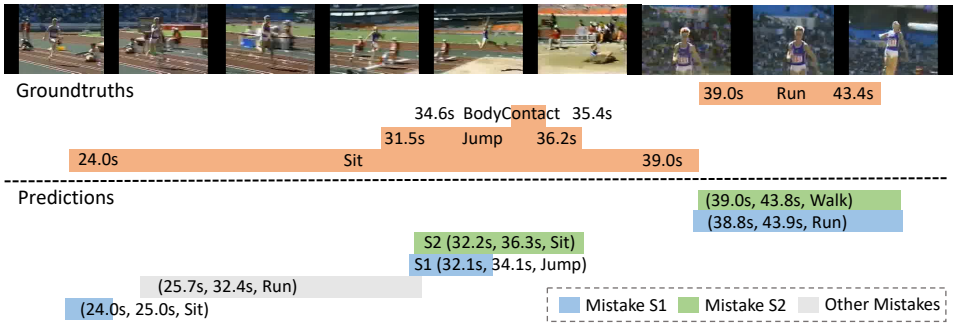

Figure 1: Illustration of action predictions by segment-based action detectors in multi-label TAD.

instance-level detection problem rather than a frame-wise segmentation task. In this sense, multi-label TAD results not only provide the action labels, but also the exact temporal extent of each instance.

Direct adaptation of action detectors is insufficient to deal with the challenges of concurrent instances and complex action relations in multi-label TAD. The convention of extracting action features from **action segments** [4, 31, 49, 43] lacks the flexibility of handling both important semantic frames inside the instance as well as discriminative boundary frames. Consider the groundtruth action of *"Jump"* in Fig. 1, segment-based action detectors mainly produce two kinds of error predictions, as in type $\mathbf{S_1}$ and type $\mathbf{S_2}$. $\mathbf{S_1}$ successfully predicts the correct action category with an **incomplete** segment of action highlights, whereas $\mathbf{S_2}$ does a better job in locating the boundaries yet get **misclassified** as *"Sit"* due to the inclusion of confusing frames. In addition, most action detectors [49, 31, 25] are inadequate in processing sampled frames and classifying fine-grained action classes. They often exploit temporal context modeling at a single level and ignore the exploration of channel semantics.

To address the above issues, we present PointTAD, a sparse query-based action detector that leverages learnable query points to flexibly attend important frames for action instances. Inspired by RepPoints [47], the query points are directly supervised by regression loss. Given specific regression targets, the query points learn to localize at discriminative frames at action boundaries as well as semantic key frames within actions. Hence, concurrent actions of different categories can yield distinctive features through the specific arrangement of query points. Moreover, we improve the action localization and semantics decoding by proposing the Multi-level Interactive Module with dynamic kernels based on query vector. Direct interpolation or pooling at each query point lacks temporal reasoning over consecutive frames. Following deformable DETR [55], we extract point-level features with deformable convolution [6, 54] from a local snippet to capture the temporal cues of action change or important movement. At instance-level, both temporal context and channel semantics are captured with frame-wise and channel-wise dynamic mixing [36, 11] to further decode the distinctive features of simultaneous actions.

PointTAD streamlines end-to-end TAD with joint optimization of the backbone network and action decoder without any post-processing technique. We validate our model on two challenging multi-label TAD benchmarks. Our model achieves the state-of-the-art detection-mAP performance and competitive segmentation-mAP performance to previous methods with RGB input.

2 Related Work

Multi-label temporal action detection. Multi-label temporal action detection has been studied as a multi-label frame-wise classification problem in the previous literature. Early methods [29, 30] paid a lot of attention on modeling the temporal relations between frames with the help of Gaussian filters in temporal dimension. Other works deployed dilated temporal kernels [9] or iterative temporal convolution and self attention blocks [8] to integrate features at different temporal scales. Recently, attention has shifted beyond temporal modeling. Coarse-Fine [16] handled different temporal resolutions in the slow-fast fashion and performed spatial-temporal attention during fusion. MLAD [40] used multi-head self-attention blocks at both spatial and class dimension to model class relations at each timestamp. In our proposed method, we study the temporal context at different semantic levels and model the channel dynamics within each instance. In addition, we employ

point-based action representation and query-based action detection framework for more accurate and precise action localization.

Segment-based representation. Following the prevailing practice of bounding boxes [20, 13, 32, 1] in object detection, existing temporal action detectors incorporated action segments heavily with three kinds of usage: as anchors, as intermediate proposals, and as final predictions. Segments as anchors are explored mainly in anchor-based frameworks. These methods [28, 26, 49, 31] used sliding windows or proposal generation results as anchors. Most TAD methods [49, 19, 45, 4, 52, 51] use segments as intermediate proposals. Uniform sampling or pooling are commonly used to extract features from these segments. P-GCN [49] applied max-pooling within local segments for proposal features. G-TAD [45] uniformly divided segments into bins and average-pooled each bin to obtain proposal features. AFSD [19] proposed boundary pooling in boundary region to refine action feature. Segments as final predictions are employed among all three types of TAD frameworks, because segments generally facilitate the computation of action overlaps and loss calculation. Instead, in this paper, we do not need segments as anchors and directly employ learnable query points as intermediate proposals with iterative refinement. The learnable query points represent the important frames within action and action feature is extracted only from these keyframes rather than using RoI pooling.

Point-based representation. Several existing works have used point representations to describe keyframes [12, 38], objects [47, 11], tracks [53], and actions [18]. [12, 38] tackled keyframe selection by operating greedy algorithm on spatial SIFT keypoints [12] or clustering on local extremes of image color/intensity [38]. These methods followed a bottom-up strategy to choose keyframes based on local cues. In contrast, PointTAD represents action as a set of temporal points (keyframes). We follow RepPoints [47] to handle the important frames of actions with point representations and refine these points by action feature iteratively. Our method directly regresses keyframes from query vectors in a top-down manner for more flexible temporal action detection. Note that PointTAD tackles different tasks from RepPoints [47]. We also built PointTAD upon a query-based detector, where a small set of action queries is employed to sparsely attend the frame sequence for potential actions, resulting in an efficient detection framework.

Temporal context in videos. Context aggregation at different levels of semantics is crucial for temporal action modeling [41] and has been discussed in previous TAD methods. G-TAD [45] treated each snippet input as graph node and applied graph convolution networks to enhance snippet-level features with global context. ContextLoc [56] handled action semantics in hierarchy: it updated snippet features with global context, obtained proposal features with frame-wise dynamic modeling within each proposal and modeled the inter-proposal relations with GCNs. Although we considered the same levels of semantic modeling, our method is different from ContextLoc. PointTAD focuses on aggregating temporal cues at multiple levels, with deformable convolution as well as *frame* and *channel* attentions. We also apply multi-head self-attention for inter-query relation modeling.

3 PointTAD

We formulate the task of multi-label temporal action detection (TAD) as a set prediction problem. Formally, given a video clip with T consecutive frames, we predict a set of action instances $\Psi = \{\psi_n = (t_n^s, t_n^e, c_n)\}_{n=1}^{N_q}$, N_q is the number of learnable queries, t_n^s, t_n^e are the starting and ending timestamp of the n-th detected instance, c_n is its action category. The groundtruth action set to detect are denoted $\hat{\Psi} = \{\hat{\psi}_n = (\hat{t_n^s}, \hat{t_n^e}, \hat{c}_n)\}_{n=1}^{N_g}$, where $\hat{t_n^s}, \hat{t_n^e}$ are the starting and ending timestamp of the n-th action, $\hat{c}_n$ is the groundtruth action category, N_g is the number of groundtruth actions.

The overall architecture of PointTAD is depicted in Fig. 2. PointTAD consists of a **video encoder** and an **action decoder**. The model takes three inputs for each sample: RGB frame sequence of length T, a set of learnable query points $\mathbf{P} = \{\mathcal{P}_i\}_{i=1}^{N_q}$, and query vectors $\mathbf{q} \in \mathbb{R}^{N_q \times D}$. Learnable query points explicitly describe the action locations by positioning themselves around action boundaries and semantic key frames as well, and the query vectors decode action semantics and locations from the sampled features. The video encoder extracts video features $X \in \mathbb{R}^{T \times D}$ from RGB frames. The action decoder contains L stacked decoder layers and takes query points $\mathbf{P}$, query vectors $\mathbf{q}$ and video features X as input. Each decoder layer contains two parts: 1) the multi-head self-attention block models the pair-wise relationship of query vectors and establishes inter-proposal modeling for action detection; 2) the **Multi-level Interactive Module** models the point-level and instance-level

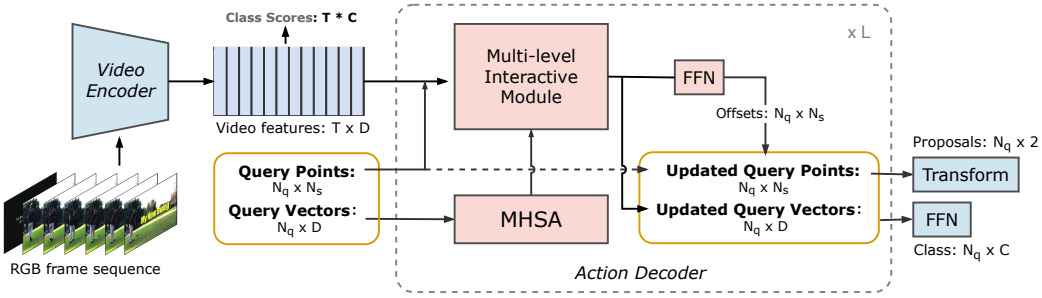

Figure 2: **Pipeline of PointTAD**. It consists of a backbone network that extracts video features from consecutive RGB frames and an action decoder of L layers that directly decodes actions from video features. PointTAD enables end-to-end training of backbone and action decoder without any post-processing of predictions.

semantics with dynamic weights based on query vector. Overall, the action decoder accomplishes context modeling at **point-wise**, **intra-proposal** and **inter-proposal** levels. Finally, we use two linear projection heads to decode action labels from query vectors, and transform query points to detected segments for loss computation and prediction output.

3.1 Video Encoder

We use the I3D backbone [3] as the video encoder in our framework. The video encoder is trained end-to-end with the action decoder and optimized by action detection loss to bridge the domain gap between action recognition and detection. For easy deployment of our framework in practice, we avoid the usage of optical flow due to its cumbersome preparation procedure. In order to achieve good performance on par with two-stream features by only using RGB input, we follow [46] to remove the temporal pooling at $\mathrm{Mixed_5c}$ and fuse the features from $\mathrm{Mixed_5c}$ with features from $\mathrm{Mixed_4f}$ as in [24]. As a result, the temporal stride of encoded video features is 4. Spatial average pooling is performed to squeeze the spatiotemporal representations from backbone to temporal features.

3.2 Learnable Query Points

Segment-based action representation (i.e., representing each action instance simply with a starting and ending time) is limited in describing its boundary and content at the same time. To increase the representation flexibility, we present a novel point-based representation to automatically learn the positions of action boundary as well as its semantic key frames inside the instance. Specifically, the point-based representation is denoted by $\mathcal{P} = \{t_j\}_{j=1}^{N_s}$ for each query, where t_j is the temporal location of jth query point, and the point quantity per query is N_s and set to 21 empirically. We explain the updating strategy and the learning of query points below.

Iterative point refinement. During training, the query points are initially placed at the midpoint of the input video clip. Then, they are refined by query vectors $\mathbf{q}$ through iterations of decoder layers to reach final positions. To be specific, at each decoder layer, the query point offsets are predicted from updated query vector (see Sec. 3.3) by linear projection. We design a self-paced updating strategy with adaptive scaling for each query at each layer to stabilize the training process. At decoder layer l, the query points for one query are represented by $\mathcal{P}^l = \{t_j^l\}_{j=1}^{N_s}$. The N_s offsets are denoted $\{\Delta t_j^l\}_{j=1}^{N_s}$. The refinement can be summarized as:

$$\mathcal{P}^{l+1} = \{(t_j^l + \Delta t_j^l \cdot s^l \cdot 0.5)\}_{j=1}^{N_s}, \tag{1}$$

where $s^l = max(\{t_j^l\}) - min(\{t_j^l\})$ is the scaling parameter and describes the span of query points at layer l. As a result, the updated step size gets smaller for shorter action, which helps with the localization of short actions. Updated query points from previous layer are inputs to the next layer.

Learning query points. The training of query points is directly supervised by regression loss at both intermediate and final stages. We follow [47] to transform query points to pseudo segments for regression loss calculation. The resulted pseudo segments participate in the calculation of L1-loss and tIoU loss with groundtruth action segments in both label assignment and loss computation.

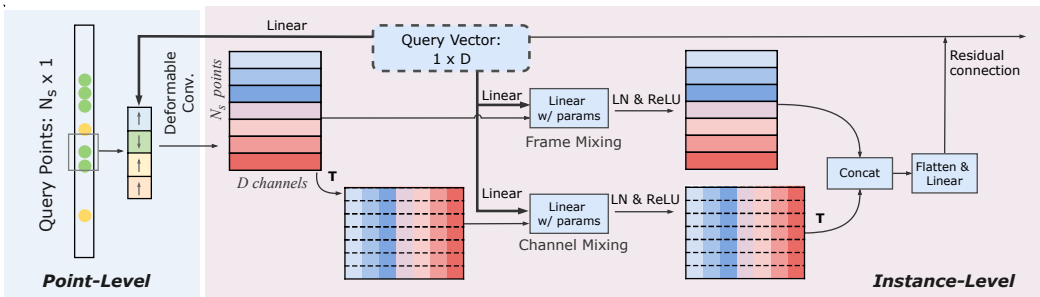

Figure 3: **Multi-level Interactive Module** aggregates action semantics at point-level and instance-level with dynamic parameters.

The transformation function is denoted by $\mathcal{T} : \mathcal{P} \rightarrow \mathcal{S} = (t^s, t^e)$. We experiment with two kinds of functions: Min-max $\mathcal{T}_1$ and Partial min-max $\mathcal{T}_2$. *Min-max* is to take the minimum and maximum location from the set of query points as the starting and ending timestamp of the pseudo segment, $\mathcal{T}_1 : \mathcal{P} \rightarrow min(\{t_j\}_{j\in\mathcal{P}}), max(\{t_j\}_{j\in\mathcal{P}})$. With Min-max transformation, the query points are strictly bounded within the local segment of the target action instance. *Partial min-max* function is to select a subset of query points $\mathcal{P}_{local}$ and perform the min-max function on them to determine a pseudo segment, $\mathcal{T}_2 : \mathcal{P} \rightarrow min(\{t_j\}_{j\in\mathcal{P}_{local}}), max(\{t_j\}_{j\in\mathcal{P}_{local}})$. It allows several query points to aggregate information from outside the action region. Empirically, we choose partial min-max by default. For each query, we randomly take $\frac{2}{3}N_s$ points from the query point set to form $\mathcal{P}_{local}$.

3.3 Multi-level Interactive Module

Apart from the limitation in the segment-based representation, previous temporal action detectors are also insufficient in decoding the sampled frames. These methods seldom consider semantic aggregation from different levels and at multiple aspects. Accordingly, we present a multi-level interactive module that consider both **local temporal cues at point-level** and **intra-proposal relation modeling at instance-level**, depicted in Fig. 3. To achieve distinct representations for each query, these parameters at both levels are dynamically generated based on the query vector.

Point-level local deformation. Paired with the refined point representation, we employ deformable convolutions to extract point features within the local neighborhood. For the jth query point, we predict four temporal offsets $\{o_k\}_{k=1}^4$ from the point location and corresponding weights $\{w_k\}_{k=1}^4$. The query point at frame t_j acts as the center point and is added with temporal offsets to achieve four deformable sub-points. These sub-points characterize the local neighborhood of center points. The features at sub-points are extracted by bilinear interpolation, then multiplied with weights and combined together to get point-level feature $x(j)$. This process can be expressed as:

$$x(j) = \sum_{k=1}^{4} X(t_j + o_k) \cdot w_k. \tag{2}$$

Both the offsets and weights are generated from query vectors $\mathbf{q}$ by linear projection,

$$\mathbf{o} = \text{Linear}(\mathbf{q}) \in \mathbb{R}^{N_q \times 4}, \quad \mathbf{w} = \text{Softmax}(\text{Linear}(\mathbf{q})) \in \mathbb{R}^{N_q \times 4}, \tag{3}$$

where the weights are additionally normalized by softmax for each query.

Instance-level adaptive mixing. Faced with the challenge of simultaneous actions, with only temporal modeling, actions with large overlap may result in similar representation and harms the classification. To tackle this problem, we propose adaptive mixing at both frames and channels by using dynamic convolutions. Specifically, the stacked features of query points is denoted by $x \in \mathbb{R}^{N_s \times D}$. Given the query vector $\mathbf{q}$, we generate dynamic parameters for frame mixing and channel mixing:

$$M_f = \text{Linear}(\mathbf{q}) \in \mathbb{R}^{N_s \times N_s}, \tag{4}$$

$$M_{c,1} = \text{Linear}(\mathbf{q}) \in \mathbb{R}^{D \times D'}, \tag{5}$$

$$M_{c,2} = \text{Linear}(\mathbf{q}) \in \mathbb{R}^{D' \times D}. \tag{6}$$

Frame mixing is carried out with dynamic projection followed by LayerNorm and ReLU activation on N_s points to explore intra-proposal frame relations:

$$x_f = \mathrm{ReLU}(\mathrm{LayerNorm}(x^T M_f)) \in \mathbb{R}^{D \times N_s}. \tag{7}$$

Similar to frame mixing, channel mixing uses two bottle-necked layers of dynamic projection on the channel dimension to enhance action semantics:

$$x_c = \mathrm{ReLU}(\mathrm{LayerNorm}(\mathrm{ReLU}(\mathrm{LayerNorm}(xM_{c,1}))M_{c,2})) \in \mathbb{R}^{N_s \times D}. \tag{8}$$

These two mixed features are concatenated along channel and squeezed by linear operations to the size of query vector. The query vector $\mathbf{q}^l$ at the lth layer is then updated with residual connection:

$$\mathbf{q}^{l+1} = \mathbf{q}^l + \mathrm{Linear}(\mathrm{Concat}(x_f^T, x_c)). \tag{9}$$

Finally, the query point offsets and the action labels are decoded from the updated vector by two linear projection heads to produce detection results at each layer.

3.4 Training and Inference

Label assignment. Similar to all query-based detectors [2, 55, 11, 37], we apply hungarian matcher on (detected) pseudo segments to search for the optimal permutation $\sigma(\cdot)$ for label assignment. The groundtruth set $\hat{\Psi}$ on each video clip is extended with no action $\varnothing$ to the size of N_q. The matching cost is formulated as:

$$\mathcal{C} = \sum_{n:\sigma(n) \neq \varnothing} \alpha_{L1} \cdot \mathcal{L}_{L1}(\mathcal{T}(\mathcal{P}_n), \hat{\psi}_{\sigma(n)}) - \alpha_{iou} \cdot \mathrm{tIoU}(\mathcal{T}(\mathcal{P}_n), \hat{\psi}_{\sigma(n)}) - \alpha_{cls} \cdot c_n. \tag{10}$$

By minimizing the matching cost, the bipartite matching algorithm finds the optimal permutation $\sigma_*(\cdot)$ that assigns each prediction with a target. $\alpha_{L1}, \alpha_{iou}, \alpha_{cls}$ is set to $1, 5, 5$, respectively.

Loss functions. PointTAD is jointly optimized by localization loss and classification loss. We use L_1 loss and iou loss as localization loss:

$$\mathcal{L}_{loc} = \sum_{n:\sigma_*(n) \neq \varnothing} \mathcal{L}_{L_1}(\mathcal{T}(\mathcal{P}_n), \hat{\psi}_{\sigma_*(n)}) + (1 - \mathrm{tIoU}(\mathcal{T}(\mathcal{P}_n), \hat{\psi}_{\sigma_*(n)})). \tag{11}$$

The cross-entropy loss between query labels and target labels is used as classification loss. In addition, to improve the performance under segmentation-mAP, we generate dense classification scores $S \in \mathbb{R}^{T \times C}$ by linear projection from video features X. Cross-entropy loss is enforced on S with dense groundtruth $\hat{S} \in \mathbb{R}^{T \times C}$. Therefore, the classification loss is composed of the set prediction loss and the dense per-frame loss:

$$\mathcal{L}_{cls} = \sum_n \mathcal{L}_{ce}(c_n, \hat{c}_{\sigma_*(n)}) + \lambda_{seg} \mathcal{L}_{ce}(S, \hat{S}). \tag{12}$$

The overall loss function is formulated as follows:

$$\mathcal{L} = \lambda_{loc} \cdot \mathcal{L}_{loc} + \lambda_{cls} \cdot \mathcal{L}_{cls}. \tag{13}$$

λ_{cls}, λ_{loc} and λ_{seg} are hyper-parameters that are set to $5, 10, 0.5$, respectively.

Inference. During inference, our PointTAD uses a single linear projection layer followed by LayerNorm and ReLU activation to predict class labels from query vectors. As for localization, we use pseudo segments transformed from query points as the final predictions. These sparse predictions are then evaluated under the detection-mAP metric. Additional dense scores S could be generated at video features for the segmentation-mAP calculation. The sparse predictions are filtered with threshold γ and processed with Gaussian kernels to approximate dense scores at each frame. Then, the approximated scores are added with weight β to the predicted dense scores for segmentation-mAP calculation. The final dense scores are generated as:

$$S_{final} = \beta \cdot \sum_{n=1}^{N_q} \mathbb{1}_{c_n > \gamma} \cdot \mathrm{Gaussian}(\psi_n) + (1 - \beta) \cdot S. \tag{14}$$

Note that the sparse predictions are adjusted by dense scores only for segmentation-mAP.

Table 1: **Comparison with the state of the art** on the MultiTHUMOS test set and Charades test set, under detection-mAP (%) and segmentation-mAP(%).

Methods	Modality	MultiTHUMOS		Charades	
		Det-mAP	Seg-mAP	Det-mAP	Seg-mAP
R-C3D [44]	RGB	-	-	-	17.6
Super-event [29]	RGB	-	36.4	-	18.6
TGM [30]	RGB	-	37.2	-	20.6
TGM [30]	RGB+OF	-	44.3	-	21.5
PDAN [9]	RGB	17.3/17.1*	40.2	8.5	23.7
Coarse-Fine [16]	RGB	-	-	6.1	25.1
MLAD [40]	RGB	14.2/13.9*	42.2	-	18.4
MLAD [40]	RGB+OF	-	51.5	-	23.7
CTRN [7]	RGB	-	**44.0**	-	25.3
CTRN [7]	RGB+OF	-	51.2	-	27.8
AGT [27]	RGB+OF	-	-	-	28.6
MS-TCT [8]	RGB	16.2/16.0*	43.1	7.9	**25.4**
Ours	RGB	**21.5/21.4***	39.8	**11.1**	21.0

* *indicates detection results excluding NoHuman class.*

4 Experiments

4.1 Datasets and Setup

Datasets. We conduct experiments on two popular multi-label action detection benchmarks: MultiTHUMOS [48] and Charades [34]. **MultiTHUMOS** is a densely labeled dataset extended from THUMOS14. It includes 413 sports videos of 65 classes. The dataset has an average number of distinctive action categories per video of 10.5, compared with 1.1 in THUMOS14. **Charades** is a large Multi-label TAD dataset that contains 9848 videos of daily in-door activities. The annotations are spread over 157 action classes, with an average of 6.8 instances per video.

Implementation details. With I3D backbone network, we encode videos at 10 fps for MultiTHUMOS and 12 fps for Charades. The spatial resolution is set to 192×192 for both datasets. The video sequence is pre-processed with the widely-used sliding window mechanism. To accommodate most of the actions, the window size is set to 256 frames for MultiTHUMOS (99.1% actions included), and 400 frames for Charades (97.3% actions included). Short videos in Charades are padded with the last input frame. Its overlap ratio is 0.75 at training, and 0 at inference. The number of queries N_q is set to 48 for both benchmarks. The number of query points per query N_s is 21. The number of deformable sub-points is set to 4 according to the number of sampling points in TadTR [25].

Appropriate **initialization** is required for backbone, query points and point-level deformable convolutions for stable training. Following common practice, the I3D backbone are initialized with Kinetics400 [17] pre-trained weights for MultiTHUMOS and Charades pre-trained weights for Charades. Query points are initialized with constant 0.5 in training and with learned weights in inference. Other possible initializations are explored in ablations. The linear layer to produce deformable offsets are initialized as follows: zeroing for weights and $[1, 2, 3, 4]$ for biases. The weights and biases to generate deformable weights are initialized as zero.

We adopt AdamW as optimizer with 1e-4 weight decay. The network is trained on a server with 8 V100 GPUs. The batch size is 3 per GPU. The learning rate is set to 2e-4 and drops by half at every 12 epochs. Backbone learning rate is additionally multiplied with 0.1 for stable training.

Evaluation metrics. The default evaluation metric for multi-label TAD is segmentation-mAP, which is the frame-wise mAP. In addition, we extend the detection-mAP metric from traditional TAD to further evaluate the completeness of predicted action instances. The detection-mAP is the instance-wise mAP of action predictions under different tIoU thresholds. We report the average mAP as well as mAPs at tIoU threshold set $\{0.1, 0.2, 0.3\}$ for both datasets. The average detection-mAP is calculated with tIoU thresholds set $[0.1 : 0.1 : 0.9]$. We argue that detection-mAP is more reasonable for an instance detection task.

Table 2: **PointTAD Ablation experiments** on MultiTHUMOS. Default setting is colored gray.

(a) **Segments** vs. **Query Points** in query-based action detectors

	0.1	0.2	0.3	Avg
Segment	35.4	33.1	30.0	19.4
Point	**39.1**	**36.6**	**33.0**	**21.5**

(b) **Initialization** of query points at training.

Init.	0.1	0.2	0.3	Avg
Uniform(0,1)	39.7	36.9	33.2	21.3
Norm(0.5,0.3)	39.9	37.0	33.1	21.2
Const.(0.5)	**39.1**	**36.6**	**33.0**	**21.5**

(c) **Point2Segment Transformation** $\mathcal{T} : \mathcal{P} \rightarrow \mathcal{S}$.

$\mathcal{T}$	0.1	0.2	0.3	Avg
Min-max	39.0	36.6	33.1	21.6
Partial Min-max	**39.1**	**36.6**	**33.0**	**21.5**

(d) **Number of Query Points** N_s per query.

N_s	0.1	0.2	0.3	Avg
9	38.6	36.2	32.4	21.0
15	38.9	36.6	33.2	21.4
21	**39.1**	**36.6**	**33.0**	**21.5**
27	39.0	36.6	32.9	21.4

(e) **Point-Level**: Deformable Convolution.

Deform. Conv.	0.1	0.2	0.3	Avg
✓	**39.1**	**36.6**	**33.0**	**21.5**
	38.1	35.7	32.2	20.8

(f) **Instance-Level**: different variants of mixing strategy.

Mixing	0.1	0.2	0.3	Avg
Frame Only	38.5	35.7	32.3	20.9
Channel Only	36.8	34.2	30.8	20.1
Frame→Channel	37.0	34.3	30.9	19.9
Channel→Frame	33.2	30.7	27.4	17.1
Parallel Mixing	**39.1**	**36.6**	**33.0**	**21.5**

4.2 Comparison with the State-of-the-Art Methods

In Tab. 1, we compare the performance of PointTAD with previous multi-label TAD methods under both segmentation-mAP and detection-mAP. The sparse prediction tuples of PointTAD are converted to dense segmentation scores with Eq. (14) for segmentation-mAP. In order to compare the detection-mAP, we reproduce several previous multi-lable temporal action localization methods and convert their dense segmentation results to sparse predictions following [48]. The prediction confidence score of an action with L_f consecutive frames is calculated as:

$$\text{score}(C, p_1 ... p_{L_f}) = (\sum_{i=1}^{L_f} p_i) \times \exp(\frac{-0.01(L_f - \mu_C)^2}{\sigma_C^2}), \tag{15}$$

where p_i is the probability for class c at frame i, μ_C and σ_C is the mean and standard derivation of action duration of class C in the training set.

Our PointTAD surpasses all previous multi-label TAD methods by a large margin under detection-mAP, indicating our ability to predict complete actions is beyond previous dense segmentation models. As for segmentation-mAP, we achieve encouraging and comparable results to the previous methods on both benchmarks with a sparse detection framework.

In Fig. 4, we show the qualitative results of PointTAD on the MultiTHUMOS test set compared to the segment-based baseline, PDAN [9] and MS-TCT [8]. PointTAD detects more instances at harder categories, such as "Fall" and "SoccerPenalty". MS-TCT and PDAN perform better at "NoHuman" category. We argue that this is because "NoHuman" class is not a well-defined action category with precise paired action boundaries, whereas PointTAD have to leverage pair relations of boundaries for localization.

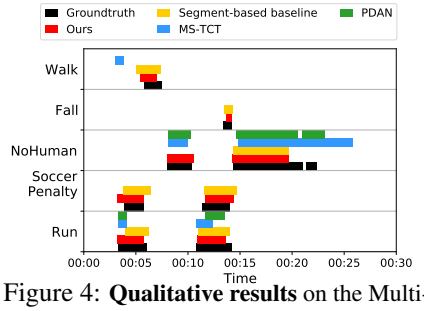

Figure 4: **Qualitative results** on the MultiTHUMOS test set.

4.3 Ablation Study

Segment-based representation vs point-based representation. In Tab. 2a, we compare the performance between segment-based representation and point-based representation. In the segment-based baseline, actions are represented by segments of paired start-end timestamps. We apply temporal RoI align to the segments as in [43, 51] to retrieve action features. Temporal RoI align divides segments into uniform bins and apply average pooling on each bin to transform frame features for query vector mixing. The number of bins for RoI align is set to N_s, the same as query point quantity. The parallel dynamic mixing is also carried out in implementations to ablate only frame sampling strategies with segments and points. Results show that point-based representation significantly outperforms segment-based representation, demonstrating the strength of adaptive sampling over segment grid alignment of frame features.

Table 3: Ablation study w.r.t. **fusion parameter** β and **scaling parameter** s on MultiTHUMOS.

(a) **Result fusion parameter** β: 1 indicates full sparse detection and 0 indicates full dense results. The results are reported under segmentation-mAP.

β	0	0.2	0.4	0.6	0.8	0.96	1
MultiTHUMOS	33.0	**39.8**	39.2	38.1	37.3	36.8	35.9
Charades	13.8	14.3	15.1	16.6	19.2	**21.0**	18.7

(b) **Offset scaling parameter** s: scale to window size vs. scale to action duration. The results are reported under detection-mAP@tIoU.

s	0.1	0.2	0.3	Avg
scale to window size	38.8	36.4	32.6	21.0
scale to action duration	**39.1**	**36.6**	**33.0**	**21.5**

Study on query point initialization. We consider three different initialization for query points in training: uniform distribution in $[0, 1]$, normal distribution with mean=0.5 and std = 0.3, and initialization with constant 0.5. Results in Tab. 2b indicates that all three initialization methods are beneficial for the training, with little performance gap in avg-mAP. In addition, normal distribution and uniform distribution achieve higher detection-mAP with lower tIoU threshold, which shows these two initialization methods are weaker at accurate localization under higher tIoU threshold. Empirically, we set constant initialization by default.

Study on pseudo segment transformation functions. Tab. 2c shows two alternatives of transformation function $\mathcal{T}$. Min-max and Partial min-max achieve similar performance under average detection-mAP. We choose Partial min-max as the default function because it offers a more relaxed constraint for query points with higher flexibility.

Study on the number of query points. We ablate with different numbers of query points N_s in Tab. 2d. In general, the performance is in proportion to the point quantity, although the increase of N_s only benefits the performance a little. N_s reaches the sweet spot at 21, where the performance no longer increases with larger N_s. To balance the model complexity and performance, we set N_s to 21.

Study on point-level deformation. In Tab. 2e, we compare point-level deformable convolution and direct interpolation for point-level feature extraction. Using deformable operator achieves higher performance than direct interpolation at query point location, demonstrating that it is beneficial to consider local context with dynamic modeling in action detection.

Study on instance-level mixing. The effect of frame-wise and channel-wise mixing is studied in Tab. 2f. We first ablate with the single application of each mixing, i.e. frame-only or channel-only. Results show that the performance degrades more without frame mixing, demonstrating the importance of frame mixing over channel mixing in multi-label TAD. Furthermore, we explore different combinations of frame and channel mixing with two cascaded alternatives, i.e. frame mixing $\rightarrow$ channel mixing and channel mixing $\rightarrow$ frame mixing. Tab. 2f shows that compared with frame-only and channel-only performances respectively, cascaded designs backfire at the performance with a decrease of 1% avg-mAP at frame $\rightarrow$ channel and 3% avg-mAP at channel $\rightarrow$ frame. To tackle action detection in video domain, intra-proposal channel mixing does not help with subsequent frame mixing and vice versa.

Study on the result fusion parameter β. Combining sparse detection results with dense segmentation scores provides smoother frame-level scores for segmentation-mAP. We ablate with choices of β on both datasets in Tab. 3a. β is set to 0.2 for MultiTHUMOS and 0.96 for Charades based on empirical results.

Study on the offset scaling parameter s. This scaling parameter is conventional in box object detectors (Faster R-CNN [32], Cascade R-CNN [1] and Sparse R-CNN [36], which is to scale regression offsets with respect to the box size instead of the image size. We extend this design to our PointTAD. In Tab. 3b, we compare the regression offsets predicted with respect to action duration (a.k.a offset scaled by duration) and with respect to window size (offset without scaling) on MultiTHUMOS. It demonstrates the effectiveness of this scaling strategy in multi-label TAD.

4.4 Query Points Visualization

Fig. 5 illustrates the learned query points and the corresponding action target on a sample from MultiTHUMOS. Partial mix-max transformation divides the query points into confined local points (blue) and open-ended global points (orange). Local points are learned to attend action boundaries

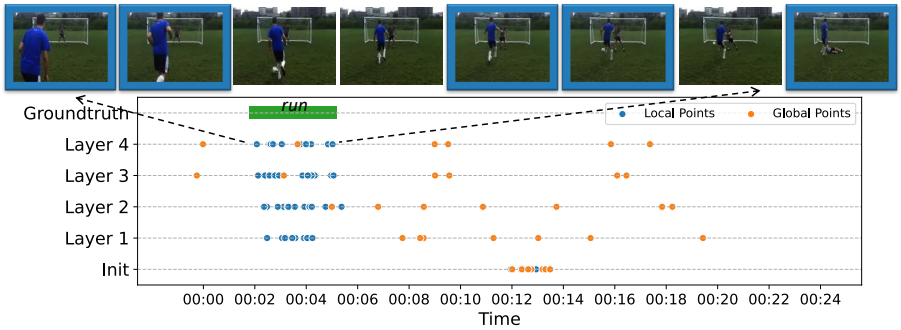

Figure 5: Visualization of **learned query points** and the corresponding action groundtruth on a sample from MultiTHUMOS.

with two extreme points and semantic key frames with interior points. Global points sparsely distribute over the entire video for context integration. Specifically, we observe that the interior local points effectively capture the most distinguished characteristic of target action *"Run"*, by capturing rapid forward movement of *both* legs and neglecting frames of similar yet disruptive movements: hopping and scoring. Moreover, the iterative refining process of query points are depicted from initial positions to the final positions in the last decoder layer. Query points are learned to kick off from the median position in the window. After the first decoder layer, the query points are already able to coarsely locate the action target. Then, these query points automatically converge to finer localization along the decoder layers.

5 Limitations and Future Work

PointTAD is proposed to solve the complex problem of multi-label TAD, by leveraging learnable query points for flexible and distinct action representation. Currently, we validate our model on two popular multi-label TAD benchmarks which include sport events and daily indoor activities. PointTAD achieves superior performance to all previous multi-label TAD methods as well as the state-of-the-art single-label TAD methods under detection-mAP metric. We have not demonstrated our model's ability in general action detection in more diverse scenarios, including important tasks such as action spotting, sentence grounding, and so on. In the future, we would continue to explore the advantages of point-based action representation in broader scope for video understanding. Meanwhile, our PointTAD training still relies on the intermediate supervision and we hope to design more effective training paradigm for the query-based detection pipeline.

6 Conclusion

In this paper, we have studied the complex multi-label TAD that aims to detect all actions from a multi-label untrimmed video. We formulate this problem as a sparse detection task and extend the traditional query-based action detection framework from single-label TAD. Faced with the challenge of concurrent instances and complex action relations in multi-label TAD, we introduce a set of learnable query points to effectively capture action boundaries and characterize action semantics for fine-grained action modeling Moreover, to facilitate the decoding process, we propose the Multi-level Interactive Module that integrates action semantics at both point level and instance level, by using dynamic kernels based on query vector. Finally, PointTAD yields an end-to-end trainable architecture by using only RGB inputs for easy deployment in practice. Our PointTAD surpasses all previous methods by a large margin under the detection-mAP and achieves promising results under the segmentation-mAP on two popular multi-label TAD benchmarks.

Acknowledgments and Disclosure of Funding

This work is supported by National Natural Science Foundation of China (No. 62076119, No. 61921006), the Fundamental Research Funds for the Central Universities (No. 020214380091), and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: delving into high quality object detection. In *CVPR*, pages 6154–6162, 2018.

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020.

[3] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *CVPR*, pages 4724–4733, 2017.

[4] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A. Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster R-CNN architecture for temporal action localization. In *CVPR*, pages 1130–1139, 2018.

[5] Guo Chen, Yin-Dong Zheng, Limin Wang, and Tong Lu. DCAN: improving temporal action detection via dual context aggregation. In *AAAI*, pages 248–257, 2022.

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, pages 764–773, 2017.

[7] Rui Dai, Srijan Das, and François Brémond. CTRN: class-temporal relational network for action detection. In *BMVC*, page 224, 2021.

[8] Rui Dai, Srijan Das, Kumara Kahatapitiya, Michael S. Ryoo, and François Brémond. MS-TCT: multi-scale temporal convtransformer for action detection. In *CVPR*, pages 20009–20019, 2022.

[9] Rui Dai, Srijan Das, Luca Minciullo, Lorenzo Garattoni, Gianpiero Francesca, and François Brémond. PDAN: pyramid dilated attention network for action detection. In *WACV*, pages 2969–2978, 2021.

[10] Jialin Gao, Zhixiang Shi, Guanshuo Wang, Jiani Li, Yufeng Yuan, Shiming Ge, and Xi Zhou. Accurate temporal action proposal generation with relation-aware pyramid network. In *AAAI*, 2020.

[11] Ziteng Gao, Limin Wang, Bing Han, and Sheng Guo. AdaMixer: A fast-converging query-based object detector. In *CVPR*, pages 5374–5373, 2022.

[12] Genliang Guan, Zhiyong Wang, Shiyang Lu, Jeremiah Da Deng, and David Dagan Feng. Keypoint-based keyframe selection. *IEEE Trans. Circuits Syst. Video Technol.*, 23(4):729–734, 2013.

[13] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(2):386–397, 2020.

[14] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, pages 961–970, 2015.

[15] Haroon Idrees, Amir Roshan Zamir, Yu-Gang Jiang, Alex Gorban, Ivan Laptev, Rahul Sukthankar, and Mubarak Shah. The THUMOS challenge on action recognition for videos "in the wild". *Comput. Vis. Image Underst.*, 155:1–23, 2017.

[16] Kumara Kahatapitiya and Michael S. Ryoo. Coarse-fine networks for temporal activity detection in videos. In *CVPR*, pages 8385–8394, 2021.

[17] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[18] Yixuan Li, Zixu Wang, Limin Wang, and Gangshan Wu. Actions as moving points. In *ECCV*, 2020.

[19] 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 *CVPR*, pages 3320–3329, 2021.

[20] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017.

[21] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: boundary-matching network for temporal action proposal generation. In *ICCV*, pages 3888–3897, 2019.

[22] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV*, pages 740–755, 2014.

[23] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. BSN: boundary sensitive network for temporal action proposal generation. In *ECCV*, pages 3–21, 2018.

[24] 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*, 2022.

[25] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *CoRR*, abs/2106.10271, 2021.

[26] Pascal Mettes, Jan C. van Gemert, Spencer Cappallo, Thomas Mensink, and Cees G. M. Snoek. Bag-of-fragments: Selecting and encoding video fragments for event detection and recounting. In *ICMR*, pages 427–434, 2015.

[27] Megha Nawhal and Greg Mori. Activity graph transformer for temporal action localization. *CoRR*, abs/2101.08540, 2021.

[28] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. Action and event recognition with fisher vectors on a compact feature set. In *ICCV*, pages 1817–1824, 2013.

[29] A. J. Piergiovanni and Michael S. Ryoo. Learning latent super-events to detect multiple activities in videos. In *CVPR*, pages 5304–5313, 2018.

[30] A. J. Piergiovanni and Michael S. Ryoo. Temporal gaussian mixture layer for videos. In *ICML*, pages 5152–5161, 2019.

[31] 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 *CVPR*, pages 485–494, 2021.

[32] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015.

[33] Rachid Riad, Olivier Teboul, David Grangier, and Neil Zeghidour. Learning strides in convolutional neural networks. *ICLR*, 2022.

[34] Gunnar A. Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *ECCV*, 2016.

[35] Haisheng Su, Weihao Gan, Wei Wu, Yu Qiao, and Junjie Yan. BSN++: complementary boundary regressor with scale-balanced relation modeling for temporal action proposal generation. In *AAAI*, 2021.

[36] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, and Ping Luo. Sparse R-CNN: end-to-end object detection with learnable proposals. In *CVPR*, pages 14454–14463, 2021.

[37] Jing Tan, Jiaqi Tang, Limin Wang, and Gangshan Wu. Relaxed transformer decoders for direct action proposal generation. In *ICCV*, pages 13506–13515, 2021.

[38] Hao Tang, Hong Liu, Wei Xiao, and Nicu Sebe. Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion. *Neurocomputing*, 331:424–433, 2019.

[39] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: fully convolutional one-stage object detection. In *ICCV*, pages 9626–9635. IEEE, 2019.

[40] Praveen Tirupattur, Kevin Duarte, Yogesh S. Rawat, and Mubarak Shah. Modeling multi-label action dependencies for temporal action localization. In *CVPR*, pages 1460–1470, 2021.

[41] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. TDN: temporal difference networks for efficient action recognition. In *CVPR*, pages 1895–1904, 2021.

[42] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. UntrimmedNets for weakly supervised action recognition and detection. In *CVPR*, pages 6402–6411, 2017.

[43] Jiannan Wu, Peize Sun, Shoufa Chen, Jiewen Yang, Zihao Qi, Lan Ma, and Ping Luo. Towards high-quality temporal action detection with sparse proposals. *CoRR*, abs/2109.08847, 2021.

[44] Huijuan Xu, Abir Das, and Kate Saenko. R-C3D: region convolutional 3d network for temporal activity detection. In *ICCV*, pages 5794–5803, 2017.

[45] Mengmeng Xu, Chen Zhao, David S. Rojas, Ali K. Thabet, and Bernard Ghanem. G-TAD: sub-graph localization for temporal action detection. In *CVPR*, pages 10153–10162, 2020.

[46] Min Yang, Guo Chen, Yin-Dong Zheng, Tong Lu, and Limin Wang. BasicTAD: an astounding rgb-only baseline for temporal action detection. *CoRR*, abs/2205.02717, 2022.

[47] Ze Yang, Shaohui Liu, Han Hu, Liwei Wang, and Stephen Lin. Reppoints: Point set representation for object detection. In *ICCV*, pages 9656–9665, 2019.

[48] Serena Yeung, Olga Russakovsky, Ning Jin, Mykhaylo Andriluka, Greg Mori, and Li Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *Int. J. Comput. Vis.*, 126(2-4):375–389, 2018.

[49] Runhao Zeng, Wenbing Huang, Chuang Gan, Mingkui Tan, Yu Rong, Peilin Zhao, and Junzhou Huang. Graph convolutional networks for temporal action localization. In *ICCV*, pages 7093–7102, 2019.

[50] Hang Zhao, Antonio Torralba, Lorenzo Torresani, and Zhicheng Yan. HACS: human action clips and segments dataset for recognition and temporal localization. In *ICCV*, pages 8667–8677, 2019.

[51] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *ECCV*, pages 539–555, 2020.

[52] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. *Int. J. Comput. Vis.*, 128(1):74–95, 2020.

[53] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In *ECCV (4)*, volume 12349 of *Lecture Notes in Computer Science*, pages 474–490, 2020.

[54] Xizhou Zhu, Han Hu, Stephen Lin, and Jifeng Dai. Deformable convnets V2: more deformable, better results. In *CVPR*, pages 9308–9316, 2019.

[55] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ICLR*, 2021.

[56] Zixin Zhu, Wei Tang, Le Wang, Nanning Zheng, and Gang Hua. Enriching local and global contexts for temporal action localization. In *ICCV*, pages 13496–13505, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Sec. 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix A.6
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] No theoretical results.
   (b) Did you include complete proofs of all theoretical results? [N/A] No theoretical results.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] The code and data are proprietary.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Sec. 4.1 and the supplementary material
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Appendix A.7
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Sec. 4.1
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Sec. 4.1
   (b) Did you mention the license of the assets? [Yes] See the supplementary material.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] See the supplementary material for details.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A] The datasets we use are open datasets.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Appendix

A.1 Comparison with the State-of-the-Art Single-label TAD Methods

In this sub-section, we report the detection-mAP results of some state-of-the-art single-label TAD methods (i.e. P-GCN [49], AFSD [19], ContextLoc [56]) reproduced on multi-label TAD task. Since MultiTHUMOS share similar data preparation with THUMOS14, the reproductions are conducted on MultiTHUMOS to get the best performance out of these models without additional hyper-parameter tuning. P-GCN [49] and ContextLoc [56] are both two-staged action detector that take in proposal generation results (e.g. BSN [23] proposals) and perform relation modeling based on coarse proposals to refine and classify each action candidate. AFSD [19] follows FCOS [39] to employ anchor-free architecture for TAD task. It is able to generate action detection results directly from the network, without pre-defined anchors or external video-level class labels.

The results in Tab. 4 show that direct application of single-label TAD models on multi-label TAD is deficient to achieve good detection performance, hence it is non-trivial to extend action detectors to multi-label TAD. Our PointTAD surpasses all of these state-of-the-art single-label TAD methods by a large margin, demonstrating the advance of our model to deal with concurrent instances and complex action relations as an action detector. Moreover, the performance of state-of-the-art single-label TAD methods are significantly worse at multi-label TAD benchmarks, indicating clear room for improvement on these benchmarks.

In addition, we follow the practice in [33] to compute the **error bar** over 3 runs, and report the mean and standard derivation in Tab. 4. The error bar for Charades results is as follows: mAP@0.1 = 16.72%$\pm$0.26%, mAP@0.2= 16.11%$\pm$0.28%, mAP@0.3= 15.17%$\pm$0.26%, mAP@0.4= 14.00%$\pm$0.26%, mAP@0.5= 12.68%$\pm$0.29%, avg-mAP = 11.36%$\pm$0.20%.

Table 4: **Comparison with the state-of-the-art single-label TAD models** on MultiTHUMOS test set with under detection-mAP (%). The single-label TAD methods are reproduced with RGB input only.

Methods	0.1	0.2	0.3	0.4	0.5	Average
BSN [23]+P-GCN [49]	22.21	19.96	16.66	12.45	8.53	9.95
BSN [23]+ContextLoc [56]	22.90	21.04	18.04	14.62	10.81	10.96
AFSD [19]	30.53	27.36	23.65	18.95	14.01	14.67
Ours	**39.20$\pm$0.10**	**36.66$\pm$0.04**	**33.32$\pm$0.03**	**28.51$\pm$0.07**	**22.90$\pm$0.36**	**21.51$\pm$0.06**

A.2 Evaluation on THUMOS14

Following RTD (query-based TAD method), we use the same feature representation and place our PointTAD head on top to build a direct TAD detector. Note that our TAD detector does not reply on the video-level classifier for action recognition and directly produce the action labels with our own PointTAD head. The result on the THUMOS14 dataset is reported in the Tab. 5. We obtain better performance on this single-label TAD dataset, demonstrating the generalization ability of PointTAD to various TAD datasets.

Table 5: **Evaluation on standard TAD benchmark THUMOS14** under detection-mAP (%).

Methods	0.3	0.4	0.5	0.6	0.7	Average
RTD + UNet	58.5	53.1	45.1	36.4	25.0	43.6
PointTAD	62.6	55.9	46.2	35.3	22.8	44.6

A.3 Comparison with DETR-based Baselines

In Table 2a of main paper, we have shown the comparison between PointTAD and a Sparse-RCNN based baseline (segment-based variant), which proves the effectiveness of point representation. We have implemented another DETR based baseline on the MultiTHUMOS dataset. The performance

comparison is reported in Tab. 6, and our PointTAD obtains better results thanks to our more flexible point-based representation.

Table 6: **Comparison with DETR-based baseline** under detection-mAP (%).

Methods	0.1	0.2	0.3	Avg
DETR-alike baseline	28.2	26.1	23.6	15.5
Sparse R-CNN alike baseline	35.4	33.1	30.0	19.4
PointTAD	**39.1**	**36.6**	**33.0**	**21.5**

Table 7: Ablation experiments w.r.t. **E2E training** on MultiTHUMOS. Default setting is colored gray.

(a) **Input spatial resolution**: from 96^2 to 192^2.

$H \times W$	0.1	0.2	0.3	Avg
96×96	30.5	28.1	24.9	16.0
128×128	36.4	33.8	30.5	19.9
160×160	38.4	35.8	32.4	20.9
192×192	**39.1**	**36.6**	**33.0**	**21.5**

(b) **Number of decoder layers** L.

L	0.1	0.2	0.3	Avg
3	37.7	35.1	31.6	20.2
4	**39.1**	36.6	**33.0**	21.5
5	39.0	**36.7**	33.0	**21.6**
6	Out of Memory			

A.4 Other Training Details

In this sub-section, we share some of our observations in building an end-to-end trainable architecture for multi-label TAD via ablations on input image resolution and the number of decoder layers L.

Study on input image resolution. In Tab. 7a, we show the detection performance with different input image resolution. According to [24], cropping images to 128^2 is adequate to tackle single-label TAD. However, we observe from the experiments that handling multi-label TAD requires more spatial information to distinguish concurrent instances from different categories and 128^2 image resolution is far less sufficient to solve the task. Our detection performance improves greatly by the increase of image resolution: from 96^2 to 128^2 the detection-mAP improves by absolute 6% at mAP@0.1, from 128^2 to 160^2 the performance improves by absolute 2% at mAP@0.1. The performance gain slows down with the increase of spatial resolution: from 160^2 to 192^2, the performance increases by 0.7% at mAP@0.1. We settled at 192^2 to balance memory consumption and model performance.

Study on the number of decoder layers. The choice of L is influenced by end-to-end training due to memory consumption by backbone. We carefully decrease L from the common setting of $L = 6$, as shown in Tab. 7b. Results indicate that 4-layer and 5-layer designs are quite similar in performance, yet from 4-layer design down, the decrease of L leads to obvious performance degrade. Hence, L is set to 4 empirically.

A.5 Visualization

To further exhibit our ability in building distinct representations for concurrent action instances, we take two concurrent actions as example and illustrate their corresponding query point predictions in Fig. 6. According to the figure, action *Run* and *SoccerPenalty* happen almost simultaneously in the video. To learn the distinct semantics for each action, two sets of query points *Pred1* and *Pred2* attend at different frames within the highly overlapping time ranges. *Pred1* aims at action *Run* and places the local points on frames that include iterative forward movements of both legs. *Pred2* detects action *SoccerPenalty* and positions the local points on frames that describe the movement of passing the ball as well as the movement of a kick. With query points, PointTAD successfully learns the authentic action semantics based on actor movements, making big process from previous action detectors that sometimes reply on background information to decide action category.

A.6 Societal Impacts

This paper proposes PointTAD, a solution with learnable query points to tackle multi-label TAD. PointTAD is the first to introduce points/keyframes for segment-level video representation. Such

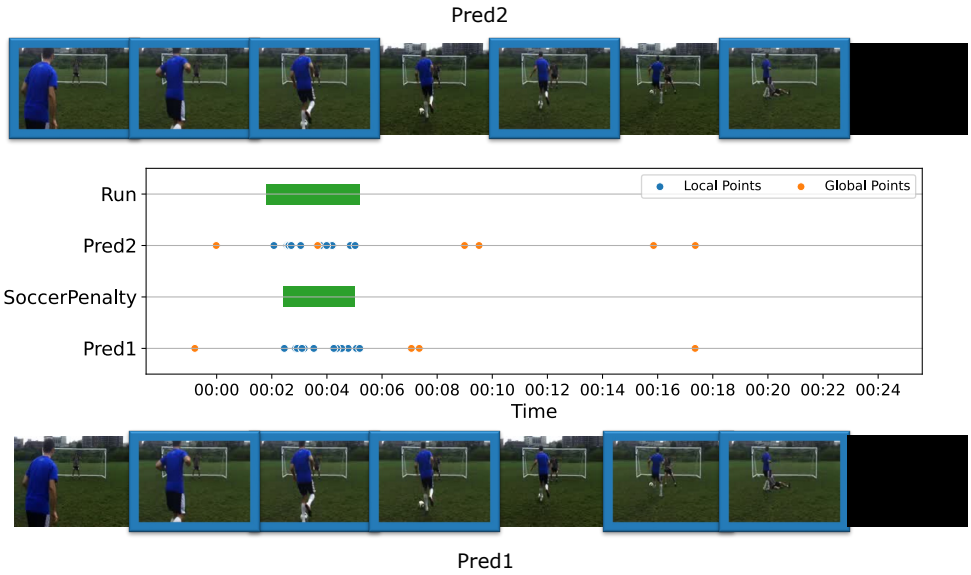

Figure 6: Visualization of predictions to **concurrent action instances**.

practice addresses the non-uniform temporal structure of videos well and could potentially drive the development of general video understanding systems for finer point-based representation. The potential applications includes video editing, anomaly event detection, etc. PointTAD enables end-to-end inference with raw video input, which benefits the deployment of automated online services for batch video processing, saving lots of human effort from offline, video-per-video handling. As the model is data-driven, any bias in training data could be captured in the algorithm. Apart from this aspect, there are no known ethical issues in the real-world applications of this technology.

A.7 Code and License

Our codebase is built upon RTD-Net[‡] [37] protected by Apache-2.0 License and Sparse R-CNN[§] [36] protected by MIT license. MultiTHUMOS [48] dataset and Charades [34] dataset are restricted to non-commercial use only.

[‡] https://github.com/MCG-NJU/RTD-Action
[§] https://github.com/PeizeSun/SparseR-CNN