A Simple and Efficient Pipeline to Build an End-to-End Spatial-Temporal Action Detector

Lin Sui[2*] Chen-Lin Zhang[1†] Lixin Gu[3] Feng Han[3]
[1] 4Paradigm Inc., Beijing, China
[2] State Key Laboratory for Novel Software Technology, Nanjing University, China
[3] DataElem Inc., Beijing, China
{suilin0432, zclnjucs}@gmail.com {gulixin, hanfeng}@dataelem.com

Abstract

Spatial-temporal action detection is a vital part of video understanding. Current spatial-temporal action detection methods mostly use an object detector to obtain person candidates and classify these person candidates into different action categories. So-called two-stage methods are heavy and hard to apply in real-world applications. Some existing methods build one-stage pipelines, But a large performance drop exists with the vanilla one-stage pipeline and extra classification modules are needed to achieve comparable performance. In this paper, we explore a simple and effective pipeline to build a strong one-stage spatial-temporal action detector. The pipeline is composed by two parts: one is a simple end-to-end spatial-temporal action detector. The proposed end-to-end detector has minor architecture changes to current proposal-based detectors and does not add extra action classification modules. The other part is a novel labeling strategy to utilize unlabeled frames in sparse annotated data. We named our model as SE-STAD. The proposed SE-STAD achieves around 2% mAP boost and around 80% FLOPs reduction. Our code will be released at `https://github.com/4paradigm-CV/SE-STAD`.

1. Introduction

Spatial-temporal action detection (STAD), which aims to classify multiple persons' actions in videos, is a vital part of video understanding. The computer vision community has drawn much attention in the field of STAD [36, 7, 42].

In previous methods, STAD is often divided into two sub-tasks: actor localization and action classification. Previous methods mostly utilize a pre-trained object detector [33, 45] and finetune it on the target dataset to obtain

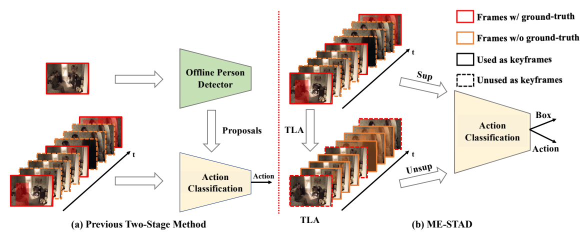

Figure 1. **Comparison between previous two-stage methods and our SE-STAD.** (a). Previous two-stage STAD methods use a heavy offline person detector, which also relies on additional data, to perform actor localization and they only use annotations of keyframes to train the action classifier. (b). Our SE-STAD trains an end-to-end spatial-temporal action detector in which actor localization part only occupies a small part of the computation. We also propose temporal label assignment (TLA) to utilize unlabeled frames in large-scale sparsely annotated datasets like AVA [13].

person candidates. Then, proposals are fed into the action classifier network to obtain the final action prediction. However, those two-stage methods are heavy and often need extra data (such as MS-COCO [24]). They need separate models and heavy computational resources. This prevents the current methods from real-world applications. A recent work [5] has shown that there is a dilemma between actor localization and action classification. Actor localization only needs a single image while action detection needs the whole input sequence. Thus, [5] proposes an end-to-end method WOO, which uses a unified backbone to perform actor localization and action detection. However, they still have a significant performance drop with the vanilla structure and need to introduce an extra attention module into the classification head to enhance the performance.

In this paper, we propose a new method named `Simple and Effective Spatial-Temporal Action Detection`, in short for SE-STAD. The general pipeline of SE-STAD is in Fig. 1. SE-STAD is composed by two parts. One is a strong one-stage spatial-temporal action

*The work was done when Lin Sui was an intern at 4Paradigm.
†Corresponding author

detector architecture. Focusing on the localization ability of detector, the proposed architecture has minor architect modifications to existing two-stage methods. Thus, our methods can be applied to many existing STAD methods with comparable performance and less computational burden. With minor added components and effective training strategies, we empower the ability to conduct actor localization and action classification simultaneously without losing accuracy. Compared to existing works, our work is light-weighted and jointly optimized, which avoids the separate learning dilemma. Besides, we are the first to explore the strategy about building an end-to-end STAD from the perspective of localization ability. SE-STAD can also get benefits from other methods, such as adopting attention-based classification heads [5, 29].

The second part is a new paradigm to utilize every possible information in sparsely annotated spatial-temporal datasets. Sparse annotation is an efficient and effective way to build a large-scale STAD dataset, only the keyframes will be annotated (e.g. 1fps in AVA [13]). A huge amount of frames do not have annotations. Hence, we propose to utilize these unlabeled data to provide more clear temporal action boundaries and help the detector learn fine-grained information. Considering the distinctiveness of unlabeled data in sparse-annotated STAD datasets, we propose a novelty pseudo labeling strategy: temporal label assignment (TLA) to generate pseudo labels. With the help of TLA, end-to-end spatial-temporal action detectors successfully enjoy performance gains from the neglected unlabeled data of sparse-annotated datasets.

Our contributions are listed as follows:

- We propose a simple and effective pipeline to build end-to-end spatial-temporal action detection methods. The proposed pipeline can be applied to many existing spatial-temporal action detection methods.
- We build a simple architecture for end-to-end action detection with effective training methods, which avoids extra offline person detector and achieves comparable performance with two-stage methods.
- A novel semi-supervised learning strategy with a pseudo temporal labeling strategy is proposed to utilize every possible information in sparsely annotated data. With the proposed pipeline, we achieve a 2.2%/1.5% mAP boost and around 80%/20% FLOPs reduction compared to both proposal-based methods and one stage methods with extra proposals.

2. Related Works

In this section, we will introduce works related to our SE-STAD, including spatial-temporal action detection, object detection and semi-supervised learning.

Spatial-Temporal Action Detection Spatial-temporal action detection (STAD) aims to detect the action of different persons in the input video clips. Thus, STAD models needs to be aware of spatial and temporal information. After large-scale datasets are annotated and introduced [13, 19], researchers have paid much attention to STAD.

Most existing works often follow a traditional Fast-RCNN [10] pipeline with pre-extracted proposals to perform STAD [13, 20, 42, 29, 6]. A previous work [43] shows the original R-CNN [11] pipeline works better for spatial-temporal action detection. However, those works are heavy and inefficient.

Besides those two-stage methods, researchers also proposed some single-stage methods for action detection. Some works [13, 8] also employ the Faster-RCNN [33] pipeline but with low performance. Early works including YOWO [16], ACRN [36] and Point3D [28] combine pre-trained 2D and 3D backbones to build pseudo end-to-end detectors. Recently, WOO [5] proposes a single-stage, unified network for end-to-end action detection. WOO first utilizes Sparse-RCNN [37] along with the key frame to generate action candidates. Then action candidates will be fed into the classifier to obtain final results. With the vanilla structure, WOO has a major performance gap with the proposal-based methods. Thus, WOO utilizes an extra attention module to boost performance. In contrast, SE-STAD has a simple modification to the current proposal-based architecture. We only add a simple object detector and utilize better training strategies, and we achieve better results than WOO and proposal-based methods.

Apart from the detection structure side, Many researchers also propose new modules to enhance the performance including feature banks modules [42, 29], attention modules [29, 5] and graph-based methods [9, 36, 48]. However, we want to build a simple and strong model for end-to-end spatial-temporal action detection. Thus, we do not add any extra modules to our SE-STAD.

Object Detection Actor localization needs to detect the location of persons in the input image. Thus, object detection is needed. Object detection has been a popular area in the computer vision community. Early works often use a two-stage pipeline with pre-defined anchors [11, 10, 33]. One-stage methods, especially anchor-free detectors are proposed to reduce the computational burden for object detection [23, 32, 25, 39, 50]. Anchor-free detectors are easy to use in real-world applications. More recent methods want to train object detectors in an end-to-end manner [3, 37].

In this paper, we adopt the one-stage anchor-free detector FCOS [39], as the person detector in SE-STAD. FCOS is a simple and effective choice for the person detector.

Semi-Supervised Learning Semi-supervised learning (SSL) aims to achieve better performance with the help of additional unlabeled data. In brief, recent semi-supervised learning follows two main ways: introducing consistency regularization [31, 2, 44] or performing pseudo-labeling [18, 38, 40]. Some other works, such as [35] also combine these two ways into a single method.

Semi-supervised object detection (SSOD), which has received lots of attention recently, is an important subdomain in SSL. CSD [14] used the consistency of predictions and proposed background elimination. Some other works [26, 47, 46] built variants of the Mean Teacher [38] framework and achieved promising performance gains.

However, in traditional semi-supervised learning tasks, the unlabeled data is introduced additionally without restrictions. Whereas, in large-scale sparsely annotated STAD datasets such as AVA [13], the unlabeled frames have high correlations with the nearby labeled frames. Temporal restriction between the labeled part and the unlabeled part has not been explored in the STAD field yet.

3. Methods

In this section, we will give a detailed description of our SE-STAD.

3.1. Notations

We will first define the notations used in this paper.

Given a spatial-temporal action detection dataset $\mathcal{D}$, which is composed of a total number of m videos: $\mathcal{D} = \{V_1, \dots, V_m\}$. For simplicity, we suppose all videos in $\mathcal{D}$ have same height h, width w and number of frames n. Thus, $V_i \in \mathbb{R}^{n\times h\times w\times 3}$. Spatial temporal action detection needs to detect the action category of the person in the specific input frame. For a frame F_j in V_i, we need to detect a tuple $(x_1, y_1, x_2, y_2, \mathbf{cls})$ for each person in F_j. (x_1, y_1, x_2, y_2) is the spatial location of the person, and $\mathbf{cls} \in [0,1]^C$ is the action category where C is the pre-defined set of action classes. In widely used sparse-annotated datasets such as AVA [13], ground-truth annotations are annotated at one frame per second. For such a dataset $\mathcal{D}$, We denotes the labeled part as $\mathcal{D}^l = \{V_1^l, \cdots, V_m^l\}$ with annotations $Y^l = \{A_1^l, \cdots, A_m^l\}$ and the unlabeled part as $\mathcal{D}^u = \{V_1^u, \cdots, V_m^u\}$.

3.2. Motivation

As shown in Sec. 2, previous STAD methods often follow a two-stage pipeline and utilize two networks: First conducting person detection with an offline object detector, then detected area of interests (RoIs) will be fed into a traditional Fast-RCNN style network to obtain the final action predictions. The two-stage networks are not efficient. Besides, they always need extra data (such as MS-COCO [24]) to train the additional person detector. WOO [5] uses a unified backbone to perform actor localization and action classification simultaneously. However, their unified models result in a large performance drop and WOO [5] proposes an extra embedding interaction head to boost the performance.

In contrast to those extra modules, we want to build a unified, end-to-end and simple method for spatial-temporal action detection. Thus, we want to make minimal modifications to current proposal-based methods, to perform spatial-temporal action detection effectively and efficiently. We name our proposed model as `Simple and Effective Spatial Temporal Action Detection (SE-STAD)`.

3.3. SE-STAD

Our proposed SE-STAD is composed of three parts: feature extraction part, actor localization part and action classification part. We unified the three parts into a single network. We will introduce these three parts step by step.

3.3.1 Feature Extraction

In this part, we directly use existing action classification backbones, i.e., SlowFast [7] for feature extraction. Moreover, SE-STAD can utilize any modern backbones to boost performance, including the recent Transformer-based models, i.e., Video-Swin [27], ViViT [1] and MViT [6].

3.3.2 Actor Localization Part

We need to perform actor localization in our SE-STAD. Previously, separate pre-trained object detectors are adopted on actor localization, the most commonly used are Faster-RCNN [33] with a ResNeXt-101 [45] backbone. However, the separate object detector has an extra heavy computational burden, which is inefficient. A recent work WOO [5] proposes to integrate an existing object detection head, i.e., Sparse R-CNN [37] into the current action classification backbone. In this paper, we follow the suggestions in WOO [5], performing actor localization with the spatial feature of keyframes. Regarding the low input resolution and efficiency issue, we choose a popular one-stage anchor-free object detector FCOS [39]. Thus, the loss for actor localization is:

$$\mathcal{L}_{al} = \mathcal{L}_{cls}(\mathbf{c}_i, \mathbf{b}_i) + \mathcal{L}_{iou}(\mathbf{c}_i, \mathbf{b}_i) + \mathcal{L}_{centerness}(\mathbf{c}_i, \mathbf{b}_i) \tag{1}$$

where $\mathbf{c}_i$ indicates the video clip, $\mathbf{b}_i$ means ground-truth bounding boxes of the keyframe of $\mathbf{c}_i$. $\mathcal{L}_{cls}$, $\mathcal{L}_{iou}$ and $\mathcal{L}_{centerness}$ are the Focal loss [23] for binary classification (existing of actor), GIoU [34] loss for bounding box regression and centerness prediction, respectively. Compared to Sparse-RCNN [37], FCOS has dense output proposals (before post-processing), and later we show that dense outputs (without post-processing) matters in STAD performance. In Sec. 4, We ablate different heads for actor localization including anchor-based heads and anchor-free heads to verify the effectiveness of different heads, and the training strategy. Our models use a simple actor localization head and perform better than vanilla WOO [5], even comparable to or better than WOO with extra attention modules.

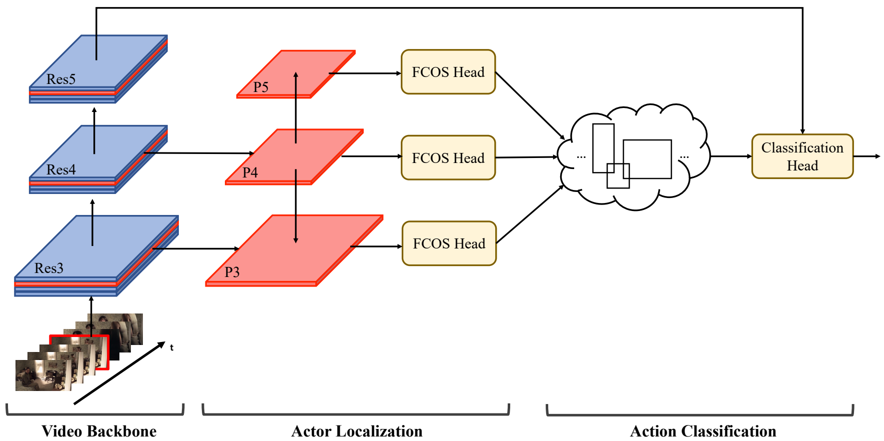

Figure 2. **Overview of our SE-STAD.** The whole pipeline consists of three parts: video backbone, actor localization part and action classification part. In the actor localization part, we build the feature pyramid on top of keyframes feature from Res3 and Res4 layers. After performing actor localization, proposals generated by FCOS heads will be used to extract features from Res5 and perform action classification.

3.3.3 Action Classification Part

For action classification, since we want to build an end-to-end spatial-temporal action detection network with minimum efforts, we follow the common practice to build an action classification head: we use the traditional ROIAlign [23] layer with temporal pooling to get the feature for each actor proposal, then a simple linear layer is attached to get the final action predictions, we use the binary cross entropy loss to train the action classification head. Thus, the loss of the action classification head becomes:

$$\mathcal{L}_{ac} = \mathcal{L}_{bce}(\mathbf{c}_i, \mathbf{b}_i, \mathbf{l}_i) \tag{2}$$

where $\mathbf{l}_i$ denotes the classification annotation and $\mathcal{L}_{bce}$ is binary cross entropy loss. In order to balance the scale of localization loss $\mathcal{L}_{al}$ and classification loss $\mathcal{L}_{ac}$, we introduce loss weight λ_{cls} for action classification which is set to 10 as default. Experiments in Sec. 4 show the model is robust to different λ_{cls}.

The overall structure of our SE-STAD is quite simple. We only add a simple FCOS head to perform actor localization. However, a simple model achieves comparable results to proposal-based methods [7] and the recently unified backbone method [5] which introduces additional attention modules.

Besides the model structure, we propose a novel semi-supervised training strategy to better utilize every possible piece of information in the training video. With the semi-supervised training stage, our model can achieve better results than originally trained models.

3.4. Semi-Supervised Action Detection for SE-STAD

It's well known that sparse annotation is an efficient and effective way to build large-scale spatial-temporal action detection datasets. However, as large parts of data are unlabeled, sparse annotations fail to provide clear temporal action boundaries. This phenomenon has been shown by previous literature [22]. Utilizing the unlabeled part is a natural way to help the detector to learn fine-grained information. Hence, we propose a new semi-supervised training method for sparsely annotated datasets in spatial-temporal action detection.

When performing semi-supervised training in SE-STAD, we adopt the online updating paradigm. Besides, in order to avoid the inductive bias introduced in semi-supervised training, following the widely used Mean Teacher [38] pipeline, we also build a teacher-student mutual-learning paradigm. Firstly, to get a good initialization for the detector, we do not perform semi-supervised training directly at first. That means we only use data with annotations to warm up the detector D by Eq. 3.

$$\mathcal{L}_{sup} = \frac{1}{N}\sum_i \mathcal{L}_{ac}(\mathbf{c}_i^s, \mathbf{b}_i^s, \mathbf{l}_i^s) + \lambda_{cls}\mathcal{L}_{al}(\mathbf{c}_i^s, \mathbf{b}_i^s) \tag{3}$$

After warming up the spatial-temporal action detector D, weights of D will be copied to the teacher model $D_{teacher}$ and student model $D_{student}$ as the initialization weights. Then we use both the labeled data and unlabeled data to further train the detector. The student model is updated via gradient back-propagation, but the gradient back-propagating to the teacher model is stopped. The teacher model is maintained by exponential moving average so as to eliminate the influence of inductive bias and provide more accurate person proposals for the student at the beginning. Loss function

Algorithm 1 Temporal Label Assignment (TLA)

Input: video clip $\mathbf{c}^u$, boxes and labels of nearest former keyframe $\mathbf{b}^{left}, \mathbf{l}^{left}$ and nearest later keyframe $\mathbf{b}^{right}, \mathbf{l}^{right}$, detector D

Output: pseudo bounding boxes $\mathbf{b}^u$, pseudo lables $\mathbf{l}^u$

$\mathbf{b}^u, \mathbf{s} = D(\mathbf{c}^u)$

$\mathbf{b}^{gt} = \mathbf{b}^{left} \bigcup \mathbf{b}^{right}, \mathbf{l}^{gt} = \mathbf{l}^{left} \bigcup \mathbf{l}^{right}$

for $i = 1, \ \ldots, \ N$

  for $j = 1, \ \ldots, \ M$

    $Cost_{ij} = \mathcal{L}_{bce}(\mathbf{s}_i, \mathbf{l}_j^{gt}) + \mathcal{L}_{L1}(\mathbf{b}_i^u, \mathbf{b}_j^{gt}) + \mathcal{L}_{iou}(\mathbf{b}_i^u, \mathbf{b}_j^{gt})$

Assignment $\hat{\pi} = \arg\min_{\pi \in \prod_N^M} \sum_i^N Cost_{i,\pi(i)}$

$inds = [\hat{\pi}(1), \cdots, \hat{\pi}(N)]$

$\mathbf{l}^u = \mathbf{l}_{gt}[inds]$

in this stage consists of losses on labeled data $\mathcal{L}_{sup}$ (Eq. 3) and unlabeled data $\mathcal{L}_{unsup}$ (Eq. 5).

$$\mathcal{L} = \mathcal{L}_{sup} + \lambda_{unsup}\mathcal{L}_{unsup} \tag{4}$$

$$\mathcal{L}_{unsup} = \frac{1}{N}\sum_i \mathcal{L}_{ac}(\mathbf{c}_i^u, \mathbf{b}_i^u, \mathbf{l}_i^u) + \lambda_{cls}\mathcal{L}_{al}(\mathbf{c}_i^u, \mathbf{b}_i^u) \tag{5}$$

where $\mathbf{b}_i^u$ and $\mathbf{l}_i^u$ are pseudo ground-truth annotations dynamically generated by temporal label assignment (TLA) which will be discussed later.

As the spatial-temporal action detection tasks are always accompanied by multi-label and long-tail classification problems, pseudo-labels have a high risk of missing tags and inaccuracies, especially for the rare categories with poor classification performance. Besides, we found that the temporal constraints are strong in spatio-temporal data. Therefore, we propose temporal label assignment (TLA) to assign classification labels for unlabeled data. Because the temporal actions are highly bounded by the temporal restrictions, we propose TLA to assign pseudo labels to detected person proposals by utilizing the neighbour annotated keyframes. The TLA procedure is detailed in Algorithm 1. Firstly, $D_{teacher}$ generates person proposals $\mathbf{b}^u$ and classification scores $\mathbf{s}$ for a video clip $\mathbf{c}^u \in \mathcal{D}^u$ with unlabeled center frame. We fetch the ground-truth bounding boxes $\mathbf{b}^{left}, \mathbf{b}^{right}$ and classification labels $\mathbf{l}^{left}, \mathbf{l}^{right}$ of the neighbour annotated keyframes which are nearest to center frame of $\mathbf{c}^u$ to perform TLA. Then we assign pseudo-labels to person proposals by resorting to the help of Hungarian algorithm. Following [3], we consider both classification and regression factors and build the cost function between i-th prediction and j-th annotation as Eq. 6.

$$Cost_{ij} = \mathcal{L}_{bce}(\mathbf{s}_i, \mathbf{l}_j^{gt}) + \mathcal{L}_{L1}(\mathbf{b}_i^u, \mathbf{b}_j^{gt}) + \mathcal{L}_{iou}(\mathbf{b}_i^u, \mathbf{b}_j^{gt}) \tag{6}$$

where $\mathcal{L}_{bce}, \mathcal{L}_{L1}$ and $\mathcal{L}_{iou}$ are binary cross entropy loss, smooth-L1 loss and GIoU loss. Weights of loss functions are set to 1. Then we use Hungarian algorithm [17] to calculate the optimal label assignment policy $\hat{\pi}$ to minimize Eq. 7.

$$\hat{\pi} = \arg\min_{\pi \in \prod_N^M} \sum_i^N Cost_{i,\pi(i)} \tag{7}$$

Finally, we can use $\hat{\pi}$ to assign pseudo classification label $\mathbf{l}_{\hat{\pi}(i)}^{gt}$ to i-th person boxes $\mathbf{b}_i^u$. Each ground-truth bounding boxes could only be assigned to one person proposal. If the number of proposals N is larger than the number ground-truth bounding boxes M, additional background objects will be added. The cost between one prediction and one background object only contains the classification part (i.e. the binary cross entropy loss).

4. Experiments

In this section, we will provide the experiments settings, results, and ablations.

4.1. Experimental Setup

4.1.1 Datasets

We mainly use AVA [13] and JHMDB [15] to conduct all our experiments.

AVA [13] is a major dataset for benchmarking the performance of spatial-temporal action detection. It contains about 211k training clips and 57k validating video clips. The labels are annotated at 1FPS. Following standard evaluation protocol [13, 8, 7], we evaluate 60 classes among the total 80 classes. We evaluate on both versions (v2.1 and v2.2) of annotations on AVA.

JHMDB [15] consists 21 action classes and 928 clips. JHMDB is a densely annotated dataset with per-frame annotations. Following previous works [43, 5], we report the frame-level mean average precision (frame-mAP) with IoU threshold of 0.5,

4.1.2 Training Details

We use a server with eight 3090 GPUs to conduct all our experiments. We use PyTorch [30] to implement our SE-STAD. To conduct a fair comparison, we adopt the commonly used backbone, SlowOnly and SlowFast [7] network as our backbone. We use SlowOnly ResNet50, SlowFast ResNet50 and SlowFast ResNet101 with non-local [41] modules to perform experiments. For the actor localization head, we use an improved version of FCOS [39], i.e., FCOS with center sampling as our actor localization head.

Following previous works [8, 7], we use SGD with momentum as our optimizer. The hyperparameters are listed as follows: The batch size is 48 with 8 GPUs (6 clips per

AVA	Model	Backbone	Frames	E2E	Pretrain	val mAP	GFLOPs
AVA v2.1	VAT [9]	I3D	64	✗	K400	25.2	N/A
	I3D [8]	I3D	64	✓	K600	21.9	N/A
	Context-RCNN [43]	R50-NL	64	✗	K400	28.0	N/A
	LFB [42]	R50-NL	64	✗	K400	25.8	N/A
	LFB [42]	R101-NL	64	✗	K400	27.1	N/A
	SlowFast [7]		32	✗	K400	24.7	97.5+406.5
	WOO* [5]	R50	32	✗	K400	25.2	141.6
	SE-STAD	8×8	32	✓	K400	25.0	111.3
	SE-STAD + TLA		32	✓	K400	**26.5**	111.3
	SlowFast [7]		32	✗	K600	27.3	151.5+406.5
	WOO* [5]	R101-NL	32	✓	K600	28.0	245.8
	SE-STAD	8×8	32	✓	K600	27.7	165.2
	SE-STAD + TLA		32	✓	K600	**28.8**	165.2
	SE-STAD + TLA*	R101	32	✓	K700	**31.8**	192.7
	TubeR [49]	8×8	32	✓	K700	31.6	240
AVA v2.2	SlowOnly [7]		4	✗	K400	20.3	41.8+406.5
	WOO* [5]	R50	4	✓	K400	21.3	68.0
	SE-STAD	4×16	4	✓	K400	21.5	55.5
	SE-STAD + TLA		4	✓	K400	**22.0**	55.5
	SlowFast [7]		32	✗	K400	24.7	97.5+406.5
	WOO* [5]	R50	32	✓	K400	25.4	147.5
	SE-STAD	8×8	32	✓	K400	25.5	111.3
	SE-STAD + TLA		32	✓	K400	**26.9**	111.3
	SlowFast [7]		32	✗	K600	27.4	151.5+406.5
	WOO* [5]	R101-NL	32	✓	K600	28.3	251.7
	SE-STAD	8×8	32	✓	K600	28.5	165.2
	SE-STAD + TLA		32	✓	K600	**29.3**	165.2

Table 1. **Results on AVA dataset.** We report the FLOPs of action classification network plus the FLOPs of person detector for proposal-based methods. We calculate the FLOPs of person detector according to the official configure file provided by [42]. * means the method reports the performance by testing with 320 resolution.

GPU) for the burn-in (baseline) stage, and 96 in the semi-supervised action detection (SSAD) stage. the ratio of labeled/unlabeled data is 1:1 in the SSAD stage. We use an initial learning rate of 0.075 and the cosine decay schedule. We train the model with 20000 iterations (around 5 epochs) in the burn-in stage, and we train the model with 40000 iterations (around 10 epochs) in the SSAD stage. For models without SSAD, we train the model with 40000 iterations. Longer training schedules (60000 or 80000 iterations) will decrease the performance by around 0.3% mAP when adopting SlowFast R50 backbone. The backbone is initialized with the pre-trained weights on Kinetics-400 or Kinetics-600 [4]. The actor localization head uses the initialization schedule in the original FCOS [39] paper. For other layers, we initialize the layer with Xavier [12]. We perform random scaling to the video clip input, we random resize the shortest edge to $[256, 320]$. and then we random crop a 256×256 video clip to feed into the model.

For the actor localization head, we will use a post-processing step with 0.3 scoring threshold and maximum number of 100 proposals during training. We `do not` perform non-maximum suppression (NMS) for actor localization head in the training stage. Then those proposals will be fed into the action classification head. The loss is showed in Sec. 3. The loss weight for $\mathcal{L}_{al}$ is 1 and 10 for $\mathcal{L}_{ac}$. Generated proposals which have at least 50% intersection-over-union (IoU) with the ground-truth boxes will be treated as positive proposals in the action classification stage, otherwise those proposals will be ignored.

4.1.3 Testing Details

Our inference steps are somehow simple. With an input video clip, we will first resize the shortest edge to 256, then directly feed into the model. We will use a post-processing step with 0.4 scoring threshold and NMS step with IoU threshold 0.3 to get the testing proposals. Then those proposals will be fed into the action classification head. We set the final action threshold as 0.002 and limit the maximum output of actors to 10 per image. During inference, we always use a single view instead of applying multi-scale testing to our models.

4.2. Results on AVA

In this section, we will provide results and analyses of results on AVA. The results are listed in Table 1. From the table, we can have the following observations:

- The extra person detector will bring a huge computational burden to the spatial-temporal action detection

Method	Backbone	JHMDB mAP
Context-Aware RCNN [43]	I3D R50-NL 8x8	79.2
WOO [5]	SlowFast R101-NL 8x8	80.5
SE-STAD	SlowFast R50 8x8	80.7
SE-STAD	SlowFast R101-NL 8x8	**82.5**

Table 2. **Results on JHMDB dataset.**

model. The FLOPs of the person detector is 406.5G FLOPs. which is around 7 times larger than the FLOPs of SlowOnly R50 (4×16), and more than 2 times larger than the heaviest backbone SlowFast R101-NL (8×8). The large FLOPs come from the high input resolution in the person detector. The high input resolution along with the high FLOPs makes proposal-based methods hard to apply in real-world scenarios.

- With a simply added component, i.e., FCOS head, our model can have roughly comparable or better performance than proposal-based methods, even than the recent WOO [5] and we do not use extra SSAD techniques. This is quite encouraging because we have around 70∼90% FLOPs drop with proposal-based SlowFast, and we have around 20∼35% FLOPs drop with WOO [5]. This shows the effectiveness of our simple models. We will dive into the model details part to figure out what makes the simple model work so well.
- With the extra SSAD techniques (the semi-supervised learning stage and the temporal labeling assignment), our model can have an extra performance boost with no extra modules and no computational cost. For example, SlowFast R50 can have an extra 1.4% mAP performance boost on AVA v2.2 and 1.5% mAP boost on AVA v2.1. Similar performance gaps are observed on SlowFast R101. However, SSAD stage can only have a 0.5% performance gain on SlowOnly R50. We conjecture that it may be due to the input capacity. SlowOnly R50 only has 4 frames as input. The low number of input frames prevents SlowOnly R50 to have better performance. We can achieve 31.8 mAP on AVA v2.1, which is 0.2 mAP higher than TubeR [49], and our model has 20% fewer FLOPs than TubeR. TubeR uses extra encoder-decoder structure with Transformers to perform end-to-end STAD. Our SE-STAD, have comparable or better performance and simple design.

4.3. Results on JHMDB

To verify the effectiveness of SE-STAD, we further evaluate our model on JHMDB [15]. Since JHMDB is densely-annotated, we directly apply the basic SE-STAD model. The results are in Table 2. From the table, we can observe that: SE-STAD models can achieve 82.5% mAP with SlowFast R101 8x8 backbone, which is 2.0% higher than WOO [5]. Even with a weaker backbone, SE-STAD can still achieve 80.7% mAP, which is still 0.2% higher than WOO. These results show the effectiveness of SE-STAD.

Actor Heads	Type	mAP
RPN+RCNN [33]	Anchor-Based	21.0
RetinaNet [23]	Anchor-Based	19.7
GFocalV2 [21]	Anchor-Based	23.7
FCOS- [39]	Anchor-Free	24.9
WOO* [5]	Anchor-Free	25.4
FCOS [39]	Anchor-Free	25.5

Table 3. **Ablation study on different heads for actor localization**. We try different heads with SlowFast R50 backbone. We apply the anchor-based version of GFocalV2 [21]. FCOS- is the original FCOS [39] version without tricks. We do not use self/semi-training for all methods. WOO [5] is listed only for comparison.

4.4. Ablation Study

In this section, we will provide ablations of our model, including the head choice for actor localization, loss coefficients, input resolutions and the methods to train the classification head. In this section, unless specified, all experiments use SlowFast R50 (8×8) as the backbone network.

4.4.1 Head Choices for Actor Localization

In this section, we vary the head of actor localization for SE-STAD. We try different heads, including the popular anchor-based heads: RPN + RCNN [33], RetinaNet [23] and GFocalV2 [21]. The ablation results are in Table 3. From Table 3 we can observe that, Anchor-based heads perform significantly worse than anchor-free heads, i.e., FCOS [39]. Two-stage RPN+RCNN [33] and RetinaNet [23] have a large performance drop. Even the most recent GFocalV2 head (anchor-based version) will have a 1.8% mAP gap with the FCOS head. Besides, tricks on FCOS will improve around 0.6% mAP, and the original FCOS head will still achieve a 24.9% mAP. It may be due to the low input resolution and pre-defined anchor shape. Moreover, with the simple FCOS head, our model performs slightly better than WOO [5]. WOO has an extra attention module. In contrast, our SE-STAD keeps a simple architectural design and has a good performance.

4.4.2 Different Strategies to Train Action Classification

The action classification part is the other important part for SE-STAD. We will use different strategies to train and test our models to ablate our FCOS head. We vary the input of action classification head between training and testing, and verify the performance of our model.

The results are in Table 4. We can find that:

- When testing with pre-extracted proposals, our model can have better performance than FCOS generated boxes. It is not surprising because we are performing actor localization with low-resolution inputs.
- However, our model still performs better than proposal-based methods. Also, our model trained with sparse in-

Training input	Testing input	*m*AP
Proposal	Proposal	24.7
GT Only	Proposal	24.5
GT Only	FCOS Output	23.7
FCOS Output (Sparse)	FCOS Output	24.3
FCOS Output (Dense)	FCOS Output	25.5
FCOS Output (Dense)	Proposal	26.2

Table 4. **Ablation study on different training inputs for action classification**. We try different inputs for both training and testing of our models. For "GT Only", we only feed the ground-truth boxes into the action classification head. For "FCOS Output (Sparse)", we perform NMS to FCOS generated proposals in the training stage. For "FCOS Output (Dense)", we do not perform NMS in the training stage.

Method	λ_{cls}	λ_{unsup}	val mAP
SE-STAD	1	-	25.2
SE-STAD	10	-	25.5
SE-STAD	20	-	24.7
SE-STAD+TLA	10	0.2	26.8
SE-STAD+TLA	10	0.5	26.9
SE-STAD+TLA	10	1.0	26.5

Table 5. **Ablation study on λ_{unsup} and λ_{cls}**. We study the effect of λ_{unsup} and λ_{cls} to verify the robustness of SE-STAD.

puts (GT, Sparse FCOS outputs) performs worse than dense inputs with a more than 1% mAP gap. This result shows that we should use dense inputs to boost the classification performance. This can be an explanation of why WOO performs badly with Sparse-RCNN [37].

4.4.3 Ablations on Loss coefficients

As stated in Sec. 3, we introduce λ_{cls} to balance the actor localization and action classification loss. We also introduce λ_{unsup} to balance labeled and unlabeled losses. Here, we make ablation experiments to show the robustness of each coefficient. Results in Table 4 support the robustness of λ_{cls} and λ_{unsup}. $\lambda_{cls} = 10$ achieves best performance. Besides, when utilizing the unlabeled data, ablation experiments show that it's better to set the ratio of loss weight between the labeled part and unlabeled part to 2:1.

4.4.4 Computational Efficiency

In this section, we will show the computational efficiency of our model under different input resolutions.

We vary the input resolution for our model during testing. The results are in Table 7. We can observe that with the default 256 input resolution, we perform slightly better than WOO [5] and proposal-based SlowFast [7]. When we use a larger input resolution, i.e., 320, we can get a 0.6% performance boost and slightly higher FLOPs than WOO [5] but much lower than proposal-based SlowFast.

4.4.5 Ablations on Pseudo Label Generation

For the semi-supervised action detection part, pseudo label generation is the critical part for this part. It's a natural way to perform pseudo label generation by predicting the classification labels directly. However, as stated before, multi-

Pseudo Label	Performance
None	25.5
Interpolation	24.8
EMA	26.0
Hard Threshold	26.0
Per Class Threshold	26.2
TLA	26.9

Table 6. **Ablation study on different strategies to generate pseudo labels for the SSAD stage on AVA v2.2**. We also report a strong baseline: training on annotated frames with EMA.

Models	Backbone	Input Res	Performance	FLOPs
SlowFast	R50, 8×8	256	24.7	97.5+406.5
WOO		320	25.4	147.5
Ours		256	25.5	111.3
Ours		320	26.1	173.8

Table 7. **Ablation study on input resolutions on AVA v2.2**. We use the square input, e.g., 256×256 to calculate the FLOPs for all our models. For WOO, we directly report the result from [5].

label and long-tail classification problems make the SSAD part hard. In order to show the excellence of TLA, we explore different strategies to generate pseudo labels:

- Hard Threshold. We apply a hard threshold for all classes to filter generated pseudo boxes.
- Per Class Threshold: We apply an independent threshold for each class to filter pseudo boxes. Thresholds are calculated from the model on the training set.
- TLA: The method we proposed in Sec. 3.

For the former two strategies, we apply a temporal labeling restriction additionally to boost the performance: we remove classes that are not in the union set of surrounding annotated frames. Besides those semi-supervised techniques, we also try a strong but simple baseline: We discard the teacher model, and the model is learned with EMA on the annotated subset. The results are in Table 6.

From the table, we can observe: the commonly used hard threshold strategy does not work on the AVA dataset if we consider the influence of EMA. Even if we consider the multi-label and long-tail problems in the dataset, and have a stronger per class threshold baseline, it still has a minor performance gain over the baseline model. In contrast, our TLA along with SSAD has a better performance, which shows the effectiveness of TLA.

5. Conclusion

In this paper, we presented SE-STAD, an end-to-end method for spatial-temporal action detection. SE-STAD has a simple design and small computational burden, yet achieves good results across the major spatial-temporal action detection dataset. The performance gain comes from two parts: one is the powerful anchor-free detector head. The other is the proposed novel semi-supervised training schema along with the label assignment strategy. We hope that our model, notwithstanding its simplicity, can en-light the broader problem of video understanding. We will continue to explore the multi-label and long-tailed problems that existed in spatial-temporal action detection.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViViT: A video vision transformer. In *Int. Conf. Comput. Vis.*, pages 6836–6846, 2021.

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Adv. Neural Inform. Process. Syst.*, 32, 2019.

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

[4] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6299–6308, 2017.

[5] Shoufa Chen, Peize Sun, Enze Xie, Chongjian Ge, Jiannan Wu, Lan Ma, Jiajun Shen, and Ping Luo. Watch only once: An end-to-end video action detection framework. In *Int. Conf. Comput. Vis.*, pages 8178–8187, 2021.

[6] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Int. Conf. Comput. Vis.*, pages 6824–6835, 2021.

[7] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast networks for video recognition. In *Int. Conf. Comput. Vis.*, pages 6202–6211, 2019.

[8] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. A better baseline for AVA. *arXiv preprint arXiv:1807.10066*, 2018.

[9] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 244–253, 2019.

[10] Ross Girshick. Fast R-CNN. In *Int. Conf. Comput. Vis.*, pages 1440–1448, 2015.

[11] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 580–587, 2014.

[12] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Int. Conf. Arti. Intell. Stat.*, pages 249–256, 2010.

[13] Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. AVA: A video dataset of spatio-temporally localized atomic visual actions. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 6047–6056, 2018.

[14] Jisoo Jeong, Seungeui Lee, Jeesoo Kim, and Nojun Kwak. Consistency-based semi-supervised learning for object detection. *Adv. Neural Inform. Process. Syst.*, 32, 2019.

[15] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *Int. Conf. Comput. Vis.*, pages 3192–3199, 2013.

[16] Okan Köpüklü, Xiangyu Wei, and Gerhard Rigoll. You only watch once: A unified CNN architecture for real-time spatiotemporal action localization. *arXiv preprint arXiv:1911.06644*, 2019.

[17] Harold W Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[18] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Int. Conf. Mach. Learn. Workshops*, page 896, 2013.

[19] Ang Li, Meghana Thotakuri, David A Ross, João Carreira, Alexander Vostrikov, and Andrew Zisserman. The AVA-Kinetics localized human actions video dataset. *arXiv preprint arXiv:2005.00214*, 2020.

[20] Dong Li, Zhaofan Qiu, Qi Dai, Ting Yao, and Tao Mei. Recurrent tubelet proposal and recognition networks for action detection. In *ECCV*, pages 303–318, 2018.

[21] Xiang Li, Wenhai Wang, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss v2: Learning reliable localization quality estimation for dense object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 11632–11641, 2021.

[22] Yixuan Li, Lei Chen, Runyu He, Zhenzhi Wang, Gangshan Wu, and Limin Wang. MultiSports: A multi-person video dataset of spatio-temporally localized sports actions. In *Int. Conf. Comput. Vis.*, pages 13536–13545, 2021.

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Int. Conf. Comput. Vis.*, pages 2980–2988, 2017.

[24] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Eur. Conf. Comput. Vis.*, volume 8693 of *LNCS*, pages 740–755, 2014.

[25] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single shot multibox detector. In *Eur. Conf. Comput. Vis.*, pages 21–37, 2016.

[26] Yen-Cheng Liu, Chih-Yao Ma, Zijian He, Chia-Wen Kuo, Kan Chen, Peizhao Zhang, Bichen Wu, Zsolt Kira, and Peter Vajda. Unbiased teacher for semi-supervised object detection. In *Int. Conf. Learn. Represent.*, pages 1–13, 2021.

[27] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021.

[28] Shentong Mo, Jingfei Xia, Xiaoqing Tan, and Bhiksha Raj. Point3D: tracking actions as moving points with 3d cnns. In *Brit. Mach. Vis. Conf.*, pages 1–14, 2021.

[29] Junting Pan, Siyu Chen, Mike Zheng Shou, Yu Liu, Jing Shao, and Hongsheng Li. Actor-context-actor relation network for spatio-temporal action localization. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 464–474, 2021.

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. PyTorch: An imperative style, high-performance deep learning library. In *Adv. Neural Inform. Process. Syst.*, volume 32, pages 1–12, 2019.

[31] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. *Adv. Neural Inform. Process. Syst.*, 28, 2015.

[32] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 779–788, 2016.

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Adv. Neural Inform. Process. Syst.*, 28:1–9, 2015.

[34] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 658–666, 2019.

[35] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Adv. Neural Inform. Process. Syst.*, 33:596–608, 2020.

[36] Chen Sun, Abhinav Shrivastava, Carl Vondrick, Kevin Murphy, Rahul Sukthankar, and Cordelia Schmid. Actor-centric relation network. In *ECCV*, pages 318–334, 2018.

[37] Peize Sun, Rufeng Zhang, Yi Jiang, Tao Kong, Chenfeng Xu, Wei Zhan, Masayoshi Tomizuka, Lei Li, Zehuan Yuan, Changhu Wang, et al. Sparse R-CNN: End-to-end object detection with learnable proposals. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 14454–14463, 2021.

[38] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Adv. Neural Inform. Process. Syst.*, 30, 2017.

[39] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *Int. Conf. Comput. Vis.*, pages 9627–9636, 2019.

[40] Guo-Hua Wang and Jianxin Wu. Repetitive reprediction deep decipher for semi-supervised learning. In *AAAI*, pages 6170–6177, 2020.

[41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018.

[42] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krahenbuhl, and Ross Girshick. Long-term feature banks for detailed video understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 284–293, 2019.

[43] Jianchao Wu, Zhanghui Kuang, Limin Wang, Wayne Zhang, and Gangshan Wu. Context-aware RCNN: A baseline for action detection in videos. In *Eur. Conf. Comput. Vis.*, pages 440–456, 2020.

[44] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Adv. Neural Inform. Process. Syst.*, 33:6256–6268, 2020.

[45] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1492–1500, 2017.

[46] Mengde Xu, Zheng Zhang, Han Hu, Jianfeng Wang, Lijuan Wang, Fangyun Wei, Xiang Bai, and Zicheng Liu. End-to-end semi-supervised object detection with soft teacher. In *Int. Conf. Comput. Vis.*, pages 3060–3069, 2021.

[47] Qize Yang, Xihan Wei, Biao Wang, Xian-Sheng Hua, and Lei Zhang. Interactive self-training with mean teachers for semi-supervised object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5941–5950, 2021.

[48] Yubo Zhang, Pavel Tokmakov, Martial Hebert, and Cordelia Schmid. A structured model for action detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 9975–9984, 2019.

[49] Jiaojiao Zhao, Yanyi Zhang, Xinyu Li, Hao Chen, Bing Shuai, Mingze Xu, Chunhui Liu, Kaustav Kundu, Yuanjun Xiong, Davide Modolo, et al. TubeR: Tubelet transformer for video action detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 13598–13607, 2022.

[50] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.