Action-Agnostic Point-Level Supervision for Temporal Action Detection

Shuhei M. Yoshida[1], **Takashi Shibata**[1], **Makoto Terao**[1], **Takayuki Okatani**[2,3], **Masashi Sugiyama**[3,4]

[1]Visual Intelligence Research Laboratories, NEC Corporation, Kanagawa 211-8666, Japan
[2]Graduate School of Information Sciences, Tohoku University, Miyagi 980-8579, Japan
[3]RIKEN Center for Advanced Intelligence Project, Tokyo 103-0027, Japan
[4]Graduate School of Frontier Sciences, The University of Tokyo, Chiba 277-8561, Japan
{s_m_yoshida, takashi-shibata, m-terao}@nec.com, okatani@vision.is.tohoku.ac.jp, sugi@k.u-tokyo.ac.jp

Abstract

We propose action-agnostic point-level (AAPL) supervision for temporal action detection to achieve accurate action instance detection with a lightly annotated dataset. In the proposed scheme, a small portion of video frames is sampled in an unsupervised manner and presented to human annotators, who then label the frames with action categories. Unlike point-level supervision, which requires annotators to search for every action instance in an untrimmed video, frames to annotate are selected without human intervention in AAPL supervision. We also propose a detection model and learning method to effectively utilize the AAPL labels. Extensive experiments on the variety of datasets (THUMOS '14, FineAction, GTEA, BEOID, and ActivityNet 1.3) demonstrate that the proposed approach is competitive with or outperforms prior methods for video-level and point-level supervision in terms of the trade-off between the annotation cost and detection performance.

Extended version — https://arxiv.org/abs/2412.21205
Code — https://github.com/smy-nec/AAPL

1 Introduction

Temporal action detection is a vital research area in computer vision and machine learning, primarily focusing on recognizing and localizing human actions and events in untrimmed video sequences (Xia and Zhan 2020; Vahdani and Tian 2022). With the rapid growth of video data available online, developing algorithms capable of understanding and interpreting such a wealth of information is critical for a wide range of applications, including anomalous event detection in surveillance videos (Vishwakarma and Agrawal 2013; Sultani, Chen, and Shah 2018) and sports activity analysis (Giancola et al. 2018; Cioppa et al. 2020). The existing literature generally tackles action detection problems through fully supervised approaches (Lin et al. 2019; Xu et al. 2020; Zhang, Wu, and Li 2022), which require training data with complete action labels and their precise temporal boundaries. Despite significant progress in recent years, these methods confront considerable challenges due to the high annotation cost to predict actions in complex and diverse video settings accurately. To reduce the annotation cost for temporal action detection, weak supervision, such as video-level supervision (Sun et al. 2015; Wang et al. 2017; Baraka and Mohd Noor 2022; Li, Zhang, and Zhang 2024) and point-level labels (Moltisanti, Fidler, and Damen 2019; Ma et al. 2020), has been studied. However, these types of supervision have their own difficulty in practice.

Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

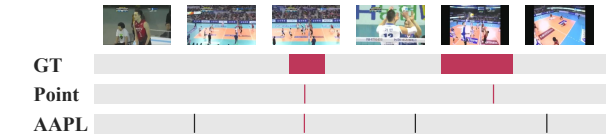

Figure 1: Illustration of ground-truth (full supervision), point-level supervision, and AAPL supervision. The red boxes and lines represents the frames labeled as "Volleyball Spiking", and the black lines represents those labeled as "Background". The images are from a video in THUMOS '14 (Jiang et al. 2014).

Video-level supervision only uses the action classes present in the video as labels. Various approaches have been proposed, such as multiple instance learning-based (Wang et al. 2017; Paul, Roy, and Roy-Chowdhury 2018), feature erasing-based (Singh and Lee 2017), and attention-based (Nguyen et al. 2018; Lee, Uh, and Byun 2020; Liu, Jiang, and Wang 2019), but they ultimately reduce action detection learning to video classification. Behind this strategy is the assumption that the discriminative intervals contributing to classification are where the actions occur, which is not always true. In addition, when a video contains multiple action classes, it becomes a multi-label classification problem, which is extremely difficult. These limitations severely limit the range of applications of video-level supervision.

Point-level supervision specifies for each action instance a single arbitrary time point in the instance and the action class and has been actively studied recently (Ju et al. 2021; Lee and Byun 2021; Li, Cao, and Ye 2023; Li, Abu Farha, and Gall 2021). While the point-level labels convey partial information about the location of action instances, they do not tell where the actions are *not*. This is a fundamental difficulty in point-level supervised action detection because localization is to distinguish actions from non-actions. In addition, the requirement that action instances must be exhaustively labeled makes the annotation process expensive.

	Instance-level localization				Video-level classes
	Available	Foreground	Background	Exhaustive	Exhaustive
Full	✓	Complete	Complete	✓	✓
Video	✗	—	—	—	✓
Point	✓	Single point	✗	✓	✓
AAPL (ours)	✓	Point(s)	Point(s)	✗	✗

Table 1: Comparison of full, video-level, point-level, and AAPL supervision. The first set of columns compares them from the perspective of four aspects of instance-level localization: whether information localizing action instances is available, what type of localization is given for foreground and background, and whether all the action instances are exhaustively annotated. The last column shows whether these types of supervision exhaustively give action classes appearing in each video.

To achieve better trade-off between the annotation costs and detection accuracy, we propose action-agnostic point-level (AAPL) supervision, a novel form of weak supervision for temporal action detection (Fig. 1). In producing AAPL labels, a small portion of video frames is sampled and presented to human annotators, who label the frames with action categories. Unlike point-level supervision, the frames to annotate are selected without human intervention. We also propose a baseline learning protocol exploiting the AAPL labels. To demonstrate the utility of AAPL supervision in various use cases, we empicically evaluate our approach using five dataests with different characteristics, including BEOID (Damen et al. 2014), GTEA (Fathi, Ren, and Rehg 2011), THUMOS '14 (Jiang et al. 2014), FineAction (Liu et al. 2022), and ActivityNet 1.3 (Heilbron et al. 2015). The results show that the proposed method is competitive with or outperforms prior methods for video-level and point-level supervision. We also find that even training only with annotated frames can achieve competitive results with the previous studies. This suggests the inherent effectiveness of AAPL supervision.

The contributions of this paper are as follows:

- We propose AAPL supervision, a novel form of weak supervision for temporal action detection, which achieves good cost-accuracy trade-offs.
- We design an action detection model and loss functions that can leverage the AAPL-labeled datasets.
- Comprehensive experiments on a wide range of action detection benchmarks demonstrate that the proposed approach is competitive with or outperforms previous methods using video-level or point-level supervision in terms of the trade-off between the annotation cost and detection performance.

2 Action-Agnostic Point-Level Supervision

We first explain the annotation pipeline for AAPL supervision (Sec. 2.1). Then, we compare AAPL supervision with other forms of weak supervision qualitatively (Sec. 2.2) and in terms of annotation time (Sec. 2.3). Some notations for AAPL labels are also introduced (Sec. 2.4).

2.1 Annotation Pipeline

AAPL supervision is characterized by the two-step annotation pipeline consisting of action-agnostic frame sampling and manual annotation. Action-agnostic frame sampling determines which frames in the training videos to annotate. This can be an arbitrary method that, without any human intervention, selects video frames to annotate. Then, human annotators label the sampled frames with action categories.

Action-agnostic frame sampling is what distinguishes AAPL supervision from conventional point-level supervision. The previous scheme requires that every action instance in a video be annotated with a single time point. This is a challenging task because human annotators need to search videos for every action instance. By contrast, for AAPL supervision, annotators just annotate the sampled frames with action categories, but they do not need to search for action instances.

The simplest examples of action-agnostic frame sampling are regularly spaced sampling and random sampling. The former picks up frames at regular intervals, while the latter selects frames randomly. A strength of these methods is that they are easy to implement, computationally light, and free from any assumption on the videos. As we will see in Sec. 4.4, the regular sampling is more preferable than the random one because the latter can result in multiple frames in temporal proximity being selected, leading to redundant annotations. We can also consider more sophisticated sampling strategies that take into account the content of the video. For example, we can use a pre-trained feature extractor to compute the feature representations of the frames and then cluster the frames based on the features. The representative frames of each cluster can be selected for annotation. See Sec. 4.4 for performance comparison.

Because sampling at regular intervals is computationally free, it might be suitable as an initial choice. In addition, it involves only one hyper-parameter, the interval length, which can be sensibly determined by using prior knowledge about the dataset, *e.g.*, the duration and frequency of action instances. If one has the computational resources, sophisticated methods like clustering-based sampling can be a good alternative, because it can adapt to the dataset characteristics and potentially provide better performance.

2.2 Qualitative Comparison

Here, we contrast AAPL supervision with other types of supervision. Table 1 compares four supervision schemes for temporal action detection: full, video-level, point-level, and AAPL supervision.

				AAPL			
	Full	Video	Point	3 sec.	5 sec.	10 sec.	30 sec.
BEOID	3.72	1.11	2.44	2.09	1.43	0.94	0.45
GTEA	4.49	0.93	3.03	1.98	1.60	1.09	0.53
THUMOS '14	1.92	0.45	1.10	1.31	0.95	0.64	0.36

Table 2: Annotation time relative to the video duration. "Full", "Video", and "Point" represent the full segment-level supervision, video-level supervision, and point-level supervision, respectively. "T sec." stands for the intervals for AAPL supervision.

Information about instance-level localization is partially available in the AAPL labels, which do not give the exact starting and ending times of action instances but do include timestamps on them. This is similar to point-level supervision, but AAPL supervision can have multiple labels on a single action instance, conveying more complete information about the action location. It also has labels on background frames, which is crucial for temporal localization because localizing an action entails finding the boundaries between the action and the background. Conventional point-level supervision contains timestamps of foreground frames only, and previous work resorts to a self-training strategy to mine background frames, assuming there is at least one background frame between two point-level labels (Lee and Byun 2021). This assumption is plausible but has a minor practical meaning for rare actions because in such cases point-level labels are distributed so sparsely that two point-level labels cannot effectively narrow down the location of the action boundaries.

Action-agnostic frame sampling is not guaranteed to find all the action instances in a video, and some action instances might not have labels. This is a potential weakness of AAPL supervision. This is true even at the video level; action-agnostic frame sampling might miss all the instances of an action class that is indeed present in the video. This makes it challenging to apply popular methods such as a video-level loss function, which is known to be effective both for video-level (Paul, Roy, and Roy-Chowdhury 2018) and point-level supervision (Ma et al. 2020; Lee and Byun 2021). However, this problem can be mitigated by a simple modification to the video-level loss introduced in Sec. 3.2.

2.3 Measurement of Annotation Time

We measured the annotation time for full, video-level, point-level, and AAPL supervision, using a modified version of the VGG Image Annotator (VIA) (Dutta, Gupta, and Zisserman 2016; Dutta and Zisserman 2019). For AAPL supervision, we sampled frames to annotate at regular intervals of 3, 5, 10, and 30 seconds. We had eight workers annotate the videos in BEOID (Damen et al. 2014), GTEA (Fathi, Ren, and Rehg 2011), and THUMOS '14 (Jiang et al. 2014). More details of this measurement are available in the extended version (Yoshida et al. 2024).

Table 2 shows the measured annotation time relative to the duration of videos, *i.e.*, the minutes it took for one annotator to annotate a 1-minute video. The previous methods ("Full", "Video", and "Point") exhibit the expected ordering that full supervision costs the most, and that video-level supervision costs the least. On the other hand, the annotation time for AAPL supervision varies with the intervals and is well-approximated by a linear function of the number of labeled frames. This modeling assumes that the annotation time per frame is constant, which is reasonable because the annotation time per frame is dominated by the time to select the action category and is not sensitive to the number of frames to annotate.

The annotation time depends on the dataset's characteristics, such as the density (*i.e.*, the number per unit length of a video) of action instances and the number of action classes occurring in one video. Indeed, these numbers are much larger in BEOID and GTEA than in THUMOS '14. As a result, annotating videos in BEOID and GTEA takes over twice as long as annotating those in THUMOS '14 for full, video-level, and point-level supervision. By contrast, the variation in the annotation time is relatively small for AAPL supervision because AAPL annotation involves local segments around the frames to label and is insensitive to global characteristics like density. This property makes it easy to apply AAPL supervision in a variety of datasets.

2.4 Notations

We introduce notations for AAPL labels. Let $\mathcal{V}$ be a set of videos. AAPL labels for a video $V \in \mathcal{V}$ are a set $\mathcal{L}^V = \{(t_i, \boldsymbol{y}_i)\}_{i \in [N^V]}$ of pairs of a time stamp t_i and an action label $\boldsymbol{y}_i \in \{0,1\}^C$. Here, N^V is the number of annotated frames, C is the number of action categories, an action label $\boldsymbol{y}_i$ is a 0/1-valued vector with the c-th component indicating the presence or absence of action of the c-th class at the time t_i, and $[K]$ is the set $\{1, 2, \ldots, K\}$. An annotated frame might not belong to any action instance. Such a frame is called a background and labeled $\boldsymbol{y} = \boldsymbol{0}$. Also, if multiple action instances of different categories overlap, the frames in the intersection are annotated with a multi-hot vector representing all the action categories present there.

3 AAPL Supervised Learning Method

This section explains our approach to temporal action detection under AAPL supervision. This includes the action detection pipeline predicting action instances from an input video (Sec. 3.1), the training objectives for the prediction model (Sec. 3.2), and the pseudo-labeling strategy to make more effective use of the training data (Sec. 3.3).

3.1 Action Detection Model

Our action detection pipeline consists of preprocessing, snippet scoring, and action instance generation, following previous studies (Li, Zhang, and Zhang 2024), At the preprocessing stage, we divide an input video into T^V non-overlapping segments of τ frames called snippets and apply the transformation to make them fit the snippet scoring model. The snippet scoring model processes an input video V into a prediction score sequence $P^V \in \mathbb{R}^{C \times T^V}$. The prediction score P^V_{ct} indicates the likelihood of an action of the

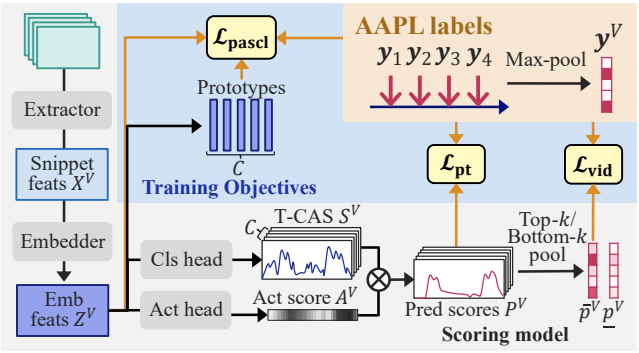

Figure 2: Illustration of the model and the loss functions.

c-th class occurring at time t. The action instance generator then converts the score sequence into a set of the scored action instances $\{(s_i, e_i, c_i, p_i)\}_{i\in[M^V]}$ in the video. Here, M^V is the number of action predictions in the video, s_i is the starting time, e_i is the ending time, c_i is the action category, and p_i is the confidence score of the i-th prediction.

The snippet scoring model comprises a feature extractor, a feature embedder, and two scoring heads, as illustrated in Fig. 2. The feature extractor is a pretrained 3D CNN converting each preprocessed snippet into a D-dimensional snippet feature. We denote the sequence of the snippet features by $X^V \in \mathbb{R}^{D\times T^V}$. The feature sequence is further fed into the feature embedder, a temporal convolution layer of the kernel size three followed by the rectified linear unit activation, which outputs the embedded feature sequence $Z^V = (\boldsymbol{z}_1^V, \ldots, \boldsymbol{z}_T^V) \in \mathbb{R}^{D\times T^V}$. The embedded features are split into $Z_S^V \in \mathbb{R}^{D/2\times T^V}$ and $Z_A^V \in \mathbb{R}^{D/2\times T^V}$, which are then input to the two scoring heads. The classification head classifies each snippet and outputs class-specific classification scores $S^V \in \mathbb{R}^{C\times T^V}$ called the temporal class activation sequence (T-CAS). The actionness head calculates class-agnostic scores called the actionness sequence $A^V \in \mathbb{R}^{T^V}$, which represents the likelihood of a snippet being in an action instance. Both heads combine a pointwise temporal convolution layer and the sigmoid function. The final prediction scores are the product of the two score sequences: $P_{ct}^V = A_t^V S_{ct}^V$.

Given the prediction scores for an action category, the action instance generator first upsamples the score sequence to match the frame rate of the input video. It then generates a set of action candidates by collecting the intervals over which the prediction scores are above a threshold θ_{pred}. This process is repeated with several different thresholds. Then, for each action candidate, it calculates the outer-inner contrastive score (Shou et al. 2018) as the initial confidence score. Finally, soft non-maximum suppression (Bodla et al. 2017) removes duplicate predictions and we calculate the final confidence scores.

3.2 Training Objectives for the Scoring Model

Our training objective is the weighted sum of three terms:

$$L = L_{\text{pt}} + \lambda_{\text{vid}} L_{\text{vid}} + \lambda_{\text{pascl}} L_{\text{pascl}}, \tag{1}$$

where L_{pt} is the point-level classification loss, L_{vid} is the video-level classification loss, and L_{pascl} is the prototype-anchored supervised contrastive loss (see Fig. 2). For brevity, we also call them the point loss, the video loss, and the contrastive loss, respectively. Unless otherwise stated, the averaging over a mini-batch of videos is implied in the expressions of the loss functions below.

The **point-level classification loss** quantifies the classification error on labeled snippets. We adopt the focal loss (Lin et al. 2020) for this purpose. We separate the contributions from the foreground and background snippets, $L_{\text{pt}} = L_{\text{pt,fg}} + L_{\text{pt,bg}}$, to handle the class imbalance between them:

$$L_{\text{pt,fg}} = \frac{-1}{|\mathcal{L}_{\text{fg}}^V|} \sum_{(t,\boldsymbol{y})\in\mathcal{L}_{\text{fg}}^V} \Bigg\{ (1-A_t^V)^2 \log A_t^V + \sum_{c=1}^{C} \left[y_c (1-S_{ct}^V)^2 \log S_{ct}^V + (1-y_c)(S_{ct}^V)^2 \log(1-S_{ct}^V) \right] \Bigg\}, \tag{2}$$

$$L_{\text{pt,bg}} = \frac{-1}{|\mathcal{L}_{\text{bg}}^v|} \sum_{(t,\boldsymbol{y})\in\mathcal{L}_{\text{bg}}^v} \left[(A_t^V)^2 \log(1-A_t^V) + \sum_{c=1}^{c} (s_{ct}^v)^2 \log(1-s_{ct}^v) \right]. \tag{3}$$

Here, $\mathcal{L}_{\text{fg}}^V$ and $\mathcal{L}_{\text{bg}}^V$ represent the subsets of AAPL labels on the foreground ($\boldsymbol{y} \neq \boldsymbol{0}$) and background ($\boldsymbol{y} = \boldsymbol{0}$) snippets, respectively. Importantly, we can calculate both $L_{\text{pt,fb}}$ and $L_{\text{pt,bg}}$ by using human-generated AAPL labels only because AAPL labels have labels on background snippets. By contrast, previous point-level methods (Ma et al. 2020; Lee and Byun 2021) require pseudo-labeling to calculate the background point loss. This is a significant advantage of AAPL supervision over previous point-level methods because action localization involves distinguishing foreground actions from the background and having reliable labels on the background snippets is crucial for learning this task.

The **video-level classification loss** measures the agreement between the video-level labels and predictions. In the AAPL-supervised setting, however, the video-level labels might be incomplete. In other words, the absence of AAPL labels of an action class does not necessarily imply that the class is absent in the video. Consequently, we cannot simply apply the video loss as used in the video-level (Wang et al. 2017; Paul, Roy, and Roy-Chowdhury 2018) and point-level (Ma et al. 2020; Lee and Byun 2021) scenarios.

To handle this incompleteness, we introduce the positive and negative parts of the video loss. The positive part of the video loss is expressed as

$$L_{\text{vid,pos}} = -\sum_{c\in[C]} y_c^V \log \bar{p}_c^V, \tag{4}$$

where $\bar{p}_c^V$ is the video-level prediction score,

$$\bar{p}_c^V = \sigma\left(\frac{1}{k_{\text{pos}}} \max_{\mathcal{T}\subset[T^V], |\mathcal{T}|=k_{\text{pos}}} \sum_{t\in\mathcal{T}} \sigma^{-1}\left(P_{ct}^V\right) \right), \tag{5}$$

y_c^V is the video-level label,

$$y_c^V = \max_{(t,\boldsymbol{y})\in\mathcal{L}^V} y_c, \tag{6}$$

and σ and σ^{-1} represent the sigmoid function and its inverse function. The terms of the form $(1 - y_c^V)\log(1 - \bar{p}_c^V)$ are excluded in our loss because $y_c^V = 0$ does not necessarily mean that the c-th class is absent in the video. This exclusion can lead to a biased estimation of the video-level prediction scores. To compensate this bias, we introduce the negative part of the video loss, $L_{\text{vid,neg}}$. To this end, we define $\underline{\boldsymbol{p}}^V$ by the "bottom-k" pooling:

$$\underline{p}_c^V = \sigma\left(\frac{1}{k_{\text{neg}}} \min_{\mathcal{T}\subset[T^V],|\mathcal{T}|=k_{\text{neg}}} \sum_{t\in\mathcal{T}} \sigma^{-1}\left(P_{ct}^V\right)\right). \quad (7)$$

This represents the average scores of frames that are not likely in the c-th class. Then, the negative part is written as

$$L_{\text{vid,neg}} = -\sum_{c\in[C]} \log\left(1 - \underline{p}_c^V\right). \quad (8)$$

The total video loss is the simple sum of the two parts: $L_{\text{vid}} = L_{\text{vid,pos}} + L_{\text{vid,neg}}$.

Following recent work (Huang et al. 2020; Liu et al. 2023; Lee and Byun 2021; Li, Cao, and Ye 2023) demonstrating that enhancing the discriminative power of embedded features improves action detection performance, we also introduce the **prototype-anchored supervised contrastive loss**. This loss was inspired by the SupCon loss (Khosla et al. 2020) and utilizes AAPL labels to enhance the embedded features. The anchors in the SupCon loss are replaced with the prototypes. This modification makes our loss computationally more efficient.

To formulate the prototype-anchored supervised contrastive loss, we first introduce the prototype $\boldsymbol{q}_c$ for the c-th action class. The prototype is the running estimate of the average embedded features for snippets belonging to action instances of the c-th class. The prototype $\boldsymbol{q}_c$ is initialized as

$$\boldsymbol{q}_c = \frac{1}{|\mathcal{L}_c|} \sum_{(V,t,\boldsymbol{y})\in\mathcal{L}_c} \boldsymbol{z}_t^V, \quad (9)$$

where $\mathcal{L}_c = \{(V,t,\boldsymbol{y}) \mid \forall V \in \mathcal{V}, (t,\boldsymbol{y}) \in \mathcal{L}^V, y_c = 1\}$ is the set of all the AAPL labels attached to c-th class action snippets in the training dataset. During the training, $\boldsymbol{q}_c$ is updated at every iteration as

$$\boldsymbol{q}_c \leftarrow (1-\mu)\boldsymbol{q}_c + \frac{\mu}{|L_c^{\mathcal{B}}|} \sum_{(V,t,\boldsymbol{y})\in\mathcal{L}_c^{\mathcal{B}}} \boldsymbol{z}_t^V, \quad (10)$$

where $\mathcal{L}_c^{\mathcal{B}}$ is a subset of $\mathcal{L}_c$ from the videos in the mini-batch for that iteration.

Using the prototypes, the prototype-anchored supervised contrastive loss for a mini-batch $\mathcal{B}$ is expressed as

$$L_{\text{pascl}} = \sum_{c\in[C]} \frac{-1}{|\mathcal{L}_c^{\mathcal{B}}|} \sum_{(V,t,\boldsymbol{y})\in\mathcal{L}_c^{\mathcal{B}}} \log\left[\frac{e^{\boldsymbol{q}_c\cdot\boldsymbol{z}_t^V/\tau}}{\sum_{(V',t',\boldsymbol{y}')\in\mathcal{L}^{\mathcal{B}}} e^{\boldsymbol{q}_c\cdot\boldsymbol{z}_{t'}^{V'}/\tau}}\right], \quad (11)$$

where $\mathcal{L}^{\mathcal{B}} = \cup_{c\in[C]}\mathcal{L}_c^{\mathcal{B}}$. Because L_{pascl} is calculated using all the videos in the mini-batch, we do not apply mini-batch averaging to L_{pascl}. This loss function pulls the embedded features of the c-th action class to $\boldsymbol{q}_c$ while repelling the others from it. We do not use a prototype for background features, and therefore, such features are repelled by all the prototype vectors.

3.3 Ground-Truth Anchored Pseudo-Labeling

Among the loss functions in the previous section, the point loss L_{pt} and the contrastive loss L_{pascl} do not involve unlabeled snippets, which constitute the majority of the snippets in the training dataset. Pseudo-labeling offers a convenient way of exploiting these underutilized data by generating pseudo labels from the predictions and using them in calculating the losses. To obtain a better outcome, the quality of the pseudo labels is crucial.

Here, we adopt the ground-truth anchored pseudo-labeling strategy, inspired by Ma et al. (2020) and Li, Cao, and Ye (2023). Under this strategy, pseudo-labels of the c-th action class are assigned to the snippets on an interval if (i) the prediction scores P_c^V over the interval are above a threshold θ_{fg}, (ii) at least one of the snippets is annotated with an AAPL label, and (iii) every AAPL label $(t,\boldsymbol{y})$ on the interval satisfies $y_c = 1$. Put differently, pseudo-labels are given to intervals with highly confident predictions consistent with AAPL labels. Similarly, pseudo-background labels are assigned to an interval if (a) the actionness scores are below a threshold θ_{bg} over the interval, and (b) there is at least one background label and no foreground action label on the interval. When calculating the point loss and the contrastive loss, we replace the AAPL labels with the pseudo labels.

4 Experiments

In this section, we empirically evaluate the effectiveness of AAPL supervision for temporal action detection. As action-agnostic frame sampling, we use the regularly spaced sampling, except in the part of Sec. 4.4 that compares different sampling schemes. We also analyze the effects of our design choices. We defer details of implementation and hyperparameters to the extended version (Yoshida et al. 2024).

4.1 Datasets

To demonstrate the usefulness in various usecases, we use five benchmark datasets with different characteristics. Here, we provide a brief overview of the datasets. More dataset statistics are shown in the extended version (Yoshida et al. 2024).

BEOID (Damen et al. 2014) is a dataset of egocentric activity videos, containing diverse activities ranging from cooking to work-outs. We adopt the training-validation split from Ma et al. (2020).

GTEA (Fathi, Ren, and Rehg 2011) also consists egocentric videos but focuses fine-grained daily activities in a kitchen. The median number of action instances per video is 18 in an about 60-second video. This number is by far the largest among the datasets used in this paper.

THUMOS '14 (Jiang et al. 2014) has significant variations in the lengths and the number of occurrences of action instances. Following the convention (Wang et al. 2017; Nguyen et al. 2018), we use the validation set for training and the test set for evaluation.

FineAction (Liu et al. 2022) is a large scale dataset for fine-grained action detection. The fine-grained nature of action categories and the sparsity of action instances make this dataset extremely challenging for action detection.

Supervision	Method	mAP@IoU [%] (BEOID) 0.1	0.3	0.5	0.7	Avg	mAP@IoU [%] (GTEA) 0.1	0.3	0.5	0.7	Avg
Point	Ma et al. (2020)	62.9	40.9	16.7	3.5	30.9	58.0	37.9	19.3	11.9	31.0
	Li, Abu Farha, and Gall (2021)	71.5	40.3	20.3	5.5	34.4	60.2	44.7	28.8	12.2	36.4
	Lee and Byun (2021)	76.9	61.4	42.7	25.1	51.8	63.9	55.7	33.9	20.8	43.5
	Li, Cao, and Ye (2023)	**78.7**	63.3	44.1	**26.9**	53.3	65.2	**56.8**	34.3	21.2	44.9
AAPL (3 sec.)	*Ours*	75.5	**67.6**	**48.5**	26.3	**55.2**	**70.3**	54.4	**37.7**	**23.4**	**46.3**

Table 3: Detection performance on GTEA and BEOID. Each column shows the mAP at a specific IoU threshold (0.1, 0.3, 0.5, and 0.7) and the average mAP (Avg) over the thresholds.

ActivityNet 1.3 (Heilbron et al. 2015) is a large-scale video dataset for action recognition and detection of 200 diverse action categories. The majority of videos in this dataset have only one action instance, and the duration of each action instance is much longer than that of the other datasets.

4.2 Evaluation Metrics

As evaluation metrics, we report mean average precision (mAP) at various thresholds for temporal intersection over union (IoU) (see Jiang et al. (2014) for the formal definition). Following convention (Lee and Byun 2021; Li, He, and Xu 2022), when calculating the average mAP (Avg mAP), we average mAP's at the thresholds from 0.1 to 0.7 with a step 0.1 for BEOID, GTEA, and THUMOS '14, and from 0.5 to 0.95 with a step 0.05 for FineAction and ActivityNet 1.3. All the reported results of our method are the average of eight runs with different random seeds.

4.3 Main Results

Table 3 provides the experimental results on BEOID and GTEA, demonstrating that our method outperforms the point-level methods in terms of the average mAP. The intervals of the AAPL labels are three seconds, which incurs less annotation costs than that for the point-level labels, as shown in Sec. 2.3. Therefore, the results in Tab. 3 not just show the better accuracy of the proposed method but also indicate the superiority of our approach regarding the trade-off between detection performance and annotation time.

Figure 3 shows the trade-off between detection performance and annotation time for AAPL-supervised learning on THUMOS '14, including the results with our full objective and with L_{pt} only. The intervals between AAPL labels are 3, 5, 10, and 30 seconds, which are converted to the annotation times using Tab. 2. For comparison, it also shows the results of previous one-stage training methods for video-level and point-level supervision. For a fixed budget of annotation time, our full objective for AAPL supervision is competitive with the state-of-the-art methods for the other types of supervision. In addition, our baseline using L_{pt} only already outperforms many of the previous methods. Even the baseline with 30-second-interval AAPL labels achieves the average mAP comparable to that of Ma et al. (2020), even though such sparse AAPL labels can be produced in one-third of the annotation time for point-level labels. This strength of the simple baseline illustrates the inherent effectiveness of AAPL supervision.

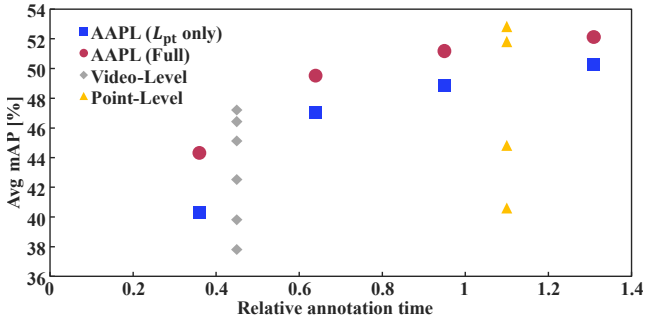

Figure 3: Trade-off between detection performance and annotation time. The blue squares represent AAPL-supervised training with L_{pt} only, the red circles represent that with our full objective, the gray diamonds represent video-level methods (Paul, Roy, and Roy-Chowdhury 2018; Min and Corso 2020; Qu et al. 2021; Huang, Wang, and Li 2022; Chen et al. 2022; Wang et al. 2023), and the yellow triangles represent point-level methods (Ma et al. 2020; Ju et al. 2021; Lee and Byun 2021; Li, Cao, and Ye 2023).

Supervision	Method	mAP@IoU [%] 0.50	0.75	0.95	Avg
Video	Li, He, and Xu (2022)	7.1	4.0	1.1	4.1
Point	Lee and Byun (2021)†	7.8	3.2	0.1	3.5
AAPL (30 sec.)	*Ours*	10.6	5.4	1.8	6.0

Table 4: Detection performance on FineAction. † indicates the results reproduced by us (Yoshida et al. 2024).

The results on FineAction are shown in Table 4. It shows that even our proposed method with the most sparse labels outperforms all the previous methods, demonstrating the strength of AAPL supervision with fine-grained and sparse actions. Interestingly, LACP (Lee and Byun 2021), the point-level method outperforming all the video-level methods on THUMOS '14, struggles with FineAction and falls short of the point-level approach, HAAN (Li, He, and Xu 2022). We conjecture this is because of the sparsity of action instances in FineAction videos. Very sparse point-level labels can enable the model to locate likely frames in actions but might not be informative enough to help a detection

Supervision	Method	mAP@IoU [%] 0.5	0.75	0.95	Avg
Point	Lee and Byun (2021)	40.4	24.6	5.7	25.1
AAPL (10 sec.)	*Ours*	41.3	25.4	5.8	25.7
AAPL (30 sec.)	*Ours*	39.6	24.3	5.6	24.7

Table 5: Detection performance on ActivityNet 1.3

	THUMOS '14				BEOID
	3 sec.	5 sec.	10 sec.	30 sec.	3 sec.
Random sampling	49.9	49.9	48.2	43.8	44.7
Regular intervals	52.1	51.2	49.5	44.3	**55.2**
Clustering	**52.9**	**51.6**	**50.2**	**45.7**	51.6

Table 6: Comparison of different methods of action-agnostic frame sampling in terms of Avg mAP [%]. The second row shows the average interval between AAPL labels.

model localize action boundaries. This hypothesis is consistent with the fact that LACP outperforms HAAN in terms of mAP@0.5, while lagging behind the latter at larger IoU thresholds; LACP successfully found the actions but failed to localize them.

Table 5 shows the results on ActivityNet 1.3. For all the intervals we experimented with, our method achieved the detection accuracy comparable or superior to Lee and Byun (2021), the state-of-the-art point-level method.

4.4 Analysis

In this section, we analyze and justify some of the design choices in our approach.

Action-Agnostic Frame Sampling. The design of action-agnostic frame sampling impacts the detection performance. To illustrate this, we conducted experiments with three different sampling schemes: random sampling, regular intervals, and clustering-based sampling. The clustering-based sampling first performs k-means clustering on snippet features extracted using a pre-trained model and then selects the frames closest to the cluster centers (see the extended version (Yoshida et al. 2024) for details). As shown in Tab. 6, the sampling at regular intervals and the clustering-based sampling consistently outperform random sampling.

This suggests that annotating diverse frames is crucial for achieving good detection performance. Indeed, both the regular-interval sampling and the clustering-based sampling tend to enhance the diversity of the annotated frames: the former does so by reducing the temporal correlation between the annotated frames, and the latter by selecting frames that are separated in the embedding space. Which of the two is better depends on the dataset, as shown in Tab. 6.

Effectiveness of Each Component. The proposed loss function consists of three components: L_{pt}, L_{vid}, and L_{pascl}. We also adopt the ground-truth anchored pseudo-labeling (PL) strategy. To evaluate the effectiveness of each component, we conducted the ablation study, as shown in Tab. 7.

L_{pt}	L_{vid}	L_{pascl}	PL	Avg mAP [%] THUMOS '14	BEOID
✓				40.3	32.0
✓	✓			42.8	46.2
✓	✓	✓		43.6	54.1
✓	✓	✓	✓	**44.3**	**55.2**

Table 7: Effectiveness of each component. AAPL labels for THUMOS '14 are sampled at intervals of 30 seconds, and those for BEOID are sampled at intervals of 3 seconds. "PL" stands for ground-truth anchored pseudo-labeling.

	Avg mAP [%] THUMOS '14	BEOID
L_{pt} only	40.3	32.0
L_{pt}+ BCE loss	40.8	42.0
$L_{\mathrm{pt}} + L_{\mathrm{vid}}$ (Ours)	**42.8**	**46.2**

Table 8: Comparison of video losses. AAPL labels for THUMOS '14 are sampled at intervals of 30 seconds, and those for BEOID are sampled at intervals of 3 seconds.

For both THUMOS '14 and BEOID, adding each component improves the detection accuracy, and the full objective achieves the best performance. The video loss makes particularly large contributions, showing that the self-training strategy based on top-/bottom-k pooling is effective with AAPL supervision, as with conventional weak supervision.

Form of the Video Loss. The proposed video loss is adapted specifically for AAPL supervision to handle the incompleteness of the video-level labels. To demonstrate the effectiveness of our design of the video loss, we compare our proposed video loss with the binary cross-entropy (BCE loss), which is the de facto standard in the field (Lee and Byun 2021; Li, Cao, and Ye 2023). As shown in Tab. 8, the forms of the video loss impact the detection performance. In particular, as shown in the extended version (Yoshida et al. 2024), mAP's at lower IoU thresholds are affected more than those at higher thresholds. This is reasonable because the video loss, as a pseudo-labeling strategy, does not concern the accurate localization of action instances but does help mine unlabeled instances.

5 Conclusion

We proposed action-agnostic point-level (AAPL) supervision for temporal action detection to achieve a better trade-off between action detection performance and annotation costs. We also proposed an action detection model and the training method to exploit AAPL-labeled data. Extensive empirical investigation suggested that AAPL supervision was competitive with or outperformed previous supervision schemes for a wide range of action detection benchmarks in terms of the cost-performance trade-off. Further analyses justified our design choices, such as frame sampling at regular intervals and the form of the video loss.

Acknowledgments

We thank Ryoma Ouchi for his efforts and commitment at the early stage of this project.

References

Baraka, A.; and Mohd Noor, M. H. 2022. Weakly-Supervised Temporal Action Localization: A Survey. *Neural Computing and Applications*, 34(11): 8479–8499.

Bodla, N.; Singh, B.; Chellappa, R.; and Davis, L. S. 2017. Soft-NMS — Improving Object Detection with One Line of Code. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 5562–5570.

Chen, M.; Gao, J.; Yang, S.; and Xu, C. 2022. Dual-Evidential Learning for Weakly-supervised Temporal Action Localization. In Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; and Hassner, T., eds., *Computer Vision – ECCV 2022*, Lecture Notes in Computer Science, 192–208. Cham: Springer Nature Switzerland. ISBN 978-3-031-19772-7.

Cioppa, A.; Deliege, A.; Giancola, S.; Ghanem, B.; Van Droogenbroeck, M.; Gade, R.; and Moeslund, T. B. 2020. A Context-Aware Loss Function for Action Spotting in Soccer Videos. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13123–13133. Seattle, WA, USA: IEEE. ISBN 978-1-72817-168-5.

Damen, D.; Leelasawassuk, T.; Haines, O.; Calway, A.; and Mayol-Cuevas, W. 2014. You-Do, I-Learn: Discovering Task Relevant Objects and Their Modes of Interaction from Multi-User Egocentric Video. In *British Machine Vision Conference (BMVC)*. Nottingham, UK.

Dutta, A.; Gupta, A.; and Zissermann, A. 2016. VGG Image Annotator (VIA).

Dutta, A.; and Zisserman, A. 2019. The VIA Annotation Software for Images, Audio and Video. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19. New York, NY, USA: ACM. ISBN 978-1-4503-6889-6.

Fathi, A.; Ren, X.; and Rehg, J. M. 2011. Learning to Recognize Objects in Egocentric Activities. In *CVPR 2011*, 3281–3288.

Giancola, S.; Amine, M.; Dghaily, T.; and Ghanem, B. 2018. SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1792–179210. Salt Lake City, UT: IEEE. ISBN 978-1-5386-6100-0.

Heilbron, F. C.; Escorcia, V.; Ghanem, B.; and Niebles, J. C. 2015. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 961–970.

Huang, L.; Huang, Y.; Ouyang, W.; and Wang, L. 2020. Relational Prototypical Network for Weakly Supervised Temporal Action Localization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07): 11053–11060.

Huang, L.; Wang, L.; and Li, H. 2022. Weakly Supervised Temporal Action Localization via Representative Snippet Knowledge Propagation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3262–3271.

Jiang, Y.-G.; Liu, J.; Roshan Zamir, A.; Toderici, G.; Laptev, I.; Shah, M.; and Sukthankar, R. 2014. THUMOS Challenge: Action Recognition with a Large Number of Classes.

Ju, C.; Zhao, P.; Chen, S.; Zhang, Y.; Wang, Y.; and Tian, Q. 2021. Divide and Conquer for Single-frame Temporal Action Localization. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 13435–13444.

Khosla, P.; Teterwak, P.; Wang, C.; Sarna, A.; Tian, Y.; Isola, P.; Maschinot, A.; Liu, C.; and Krishnan, D. 2020. Supervised Contrastive Learning. In *Advances in Neural Information Processing Systems*, volume 33, 18661–18673. Curran Associates, Inc.

Lee, P.; and Byun, H. 2021. Learning Action Completeness From Points for Weakly-Supervised Temporal Action Localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 13648–13657.

Lee, P.; Uh, Y.; and Byun, H. 2020. Background Suppression Network for Weakly-Supervised Temporal Action Localization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07): 11320–11327.

Li, P.; Cao, J.; and Ye, X. 2023. Prototype Contrastive Learning for Point-Supervised Temporal Action Detection. *Expert Systems with Applications*, 213: 118965.

Li, R.; Zhang, T.; and Zhang, R. 2024. Weakly Supervised Temporal Action Localization: A Survey. *Multimedia Tools and Applications*, 1–26.

Li, Z.; Abu Farha, Y.; and Gall, J. 2021. Temporal Action Segmentation From Timestamp Supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8365–8374.

Li, Z.; He, L.; and Xu, H. 2022. Weakly-Supervised Temporal Action Detection for Fine-Grained Videos with Hierarchical Atomic Actions. In Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; and Hassner, T., eds., *Computer Vision – ECCV 2022*, 567–584. Cham: Springer Nature Switzerland. ISBN 978-3-031-20080-9.

Lin, T.; Liu, X.; Li, X.; Ding, E.; and Wen, S. 2019. BMN: Boundary-Matching Network for Temporal Action Proposal Generation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 3888–3897. IEEE Computer Society. ISBN 978-1-72814-803-8.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2020. Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2): 318–327.

Liu, D.; Jiang, T.; and Wang, Y. 2019. Completeness Modeling and Context Separation for Weakly Supervised Temporal Action Localization. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1298–1307.

Liu, Q.; Wang, Z.; Rong, S.; Li, J.; and Zhang, Y. 2023. Revisiting Foreground and Background Separation in Weakly-supervised Temporal Action Localization: A Clustering-

based Approach. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10433–10443.

Liu, Y.; Wang, L.; Wang, Y.; Ma, X.; and Qiao, Y. 2022. FineAction: A Fine-Grained Video Dataset for Temporal Action Localization. *IEEE Transactions on Image Processing*, 31: 6937–6950.

Ma, F.; Zhu, L.; Yang, Y.; Zha, S.; Kundu, G.; Feiszli, M.; and Shou, Z. 2020. SF-Net: Single-Frame Supervision for Temporal Action Localization. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J.-M., eds., *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, 420–437. Cham: Springer International Publishing. ISBN 978-3-030-58548-8.

Min, K.; and Corso, J. J. 2020. Adversarial Background-Aware Loss for Weakly-Supervised Temporal Activity Localization. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J.-M., eds., *Computer Vision – ECCV 2020*, Lecture Notes in Computer Science, 283–299. Cham: Springer International Publishing. ISBN 978-3-030-58568-6.

Moltisanti, D.; Fidler, S.; and Damen, D. 2019. Action Recognition From Single Timestamp Supervision in Untrimmed Videos. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9907–9916. IEEE Computer Society. ISBN 978-1-72813-293-8.

Nguyen, P.; Han, B.; Liu, T.; and Prasad, G. 2018. Weakly Supervised Action Localization by Sparse Temporal Pooling Network. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6752–6761. Salt Lake City, UT, USA: IEEE. ISBN 978-1-5386-6420-9.

Paul, S.; Roy, S.; and Roy-Chowdhury, A. K. 2018. W-TALC: Weakly-Supervised Temporal Activity Localization and Classification. In Ferrari, V.; Hebert, M.; Sminchisescu, C.; and Weiss, Y., eds., *Computer Vision – ECCV 2018*, Lecture Notes in Computer Science, 588–607. Cham: Springer International Publishing. ISBN 978-3-030-01225-0.

Qu, S.; Chen, G.; Li, Z.; Zhang, L.; Lu, F.; and Knoll, A. 2021. ACM-Net: Action Context Modeling Network for Weakly-Supervised Temporal Action Localization. arXiv:2104.02967.

Shou, Z.; Gao, H.; Zhang, L.; Miyazawa, K.; and Chang, S.-F. 2018. AutoLoc: Weakly-Supervised Temporal Action Localization in Untrimmed Videos. In Ferrari, V.; Hebert, M.; Sminchisescu, C.; and Weiss, Y., eds., *Computer Vision – ECCV 2018*, Lecture Notes in Computer Science, 162–179. Cham: Springer International Publishing. ISBN 978-3-030-01270-0.

Singh, K. K.; and Lee, Y. J. 2017. Hide-and-Seek: Forcing a Network to Be Meticulous for Weakly-Supervised Object and Action Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, 3544–3553.

Sultani, W.; Chen, C.; and Shah, M. 2018. Real-World Anomaly Detection in Surveillance Videos. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6479–6488. Salt Lake City, UT: IEEE. ISBN 978-1-5386-6420-9.

Sun, C.; Shetty, S.; Sukthankar, R.; and Nevatia, R. 2015. Temporal Localization of Fine-Grained Actions in Videos by Domain Transfer from Web Images. In *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, 371–380. New York, NY, USA: Association for Computing Machinery. ISBN 978-1-4503-3459-4.

Vahdani, E.; and Tian, Y. 2022. Deep Learning-based Action Detection in Untrimmed Videos: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–20.

Vishwakarma, S.; and Agrawal, A. 2013. A Survey on Activity Recognition and Behavior Understanding in Video Surveillance. *The Visual Computer*, 29(10): 983–1009.

Wang, G.; Zhao, P.; Zhao, C.; Yang, S.; Cheng, J.; Leng, L.; Liao, J.; and Guo, Q. 2023. Weakly-Supervised Action Localization by Hierarchically-Structured Latent Attention Modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10203–10213.

Wang, L.; Xiong, Y.; Lin, D.; and Gool, L. V. 2017. UntrimmedNets for Weakly Supervised Action Recognition and Detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 6402–6411. IEEE Computer Society. ISBN 978-1-5386-0457-1.

Xia, H.; and Zhan, Y. 2020. A Survey on Temporal Action Localization. *IEEE Access*, 8: 70477–70487.

Xu, M.; Zhao, C.; Rojas, D. S.; Thabet, A.; and Ghanem, B. 2020. G-TAD: Sub-Graph Localization for Temporal Action Detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10153–10162. Seattle, WA, USA: IEEE. ISBN 978-1-72817-168-5.

Yoshida, S. M.; Shibata, T.; Terao, M.; Okatani, T.; and Sugiyama, M. 2024. Action-Agnostic Point-Level Supervision for Temporal Action Detection. arXiv:2412.21205.

Zhang, C.-L.; Wu, J.; and Li, Y. 2022. ActionFormer: Localizing Moments of Actions with Transformers. In Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; and Hassner, T., eds., *Computer Vision – ECCV 2022*, Lecture Notes in Computer Science, 492–510. Cham: Springer Nature Switzerland. ISBN 978-3-031-19772-7.