Improving Weakly Supervised Temporal Action Localization by Bridging Train-Test Gap in Pseudo Labels

Jingqiu Zhou [1] Linjiang Huang[1,2] * Liang Wang[4] Si Liu[5] Hongsheng Li[1,2,3]
[1]CUHK-SenseTime Joint Laboratory, The Chinese University of Hong Kong
[2]Centre for Perceptual and Interactive Intelligence, Hong Kong
[3]Xidian Uniersity
[4]Institute of Automation Chinese Academy of Science
[5]Beihang University
1155167063@link.cuhk.edu.hk, ljhuang524@gmail.com, hsli@ee.cuhk.edu.hk

Abstract

The task of weakly supervised temporal action localization targets at generating temporal boundaries for actions of interest, meanwhile the action category should also be classified. Pseudo-label-based methods, which serve as an effective solution, have been widely studied recently. However, existing methods generate pseudo labels during training and make predictions during testing under different pipelines or settings, resulting in a gap between training and testing. In this paper, we propose to generate high-quality pseudo labels from the predicted action boundaries. Nevertheless, we note that existing post-processing, like NMS, would lead to information loss, which is insufficient to generate high-quality action boundaries. More importantly, transforming action boundaries into pseudo labels is quite challenging, since the predicted action instances are generally overlapped and have different confidence scores. Besides, the generated pseudo-labels can be fluctuating and inaccurate at the early stage of training. It might repeatedly strengthen the false predictions if there is no mechanism to conduct self-correction. To tackle these issues, we come up with an effective pipeline for learning better pseudo labels. Firstly, we propose a Gaussian weighted fusion module to preserve information of action instances and obtain high-quality action boundaries. Second, we formulate the pseudo-label generation as an optimization problem under the constraints in terms of the confidence scores of action instances. Finally, we introduce the idea of Δ pseudo labels, which enables the model with the ability of self-correction. Our method achieves superior performance to existing methods on two benchmarks, THUMOS14 and ActivityNet1.3, achieving gains of 1.9% on THUMOS14 and 3.7% on ActivityNet1.3 in terms of average mAP. Our code is available at https://github.com/zhou745/GauFuse_WSTAL.git.

1. Introduction

The task of temporal action localization seeks to identify the action boundaries and to recognize action categories that are performed in the video. Action localization can contribute to video understanding, editing, etc. Previous works [3, 19, 20, 43, 51] mainly solved this task in the fully supervised setting, which requires both video-level labels and frame-wise annotations. However, frame-wisely annotating videos is labor-intensive and time-consuming. To reduce the annotation cost, researchers start to focus on the weakly supervised setting. Considering the rich video resources from various video websites and apps, weakly supervised setting would save tremendous annotation efforts.

Unlike its supervised counterpart, the weakly supervised temporal action localization task only requires video-level category labels. The existing works mainly follow the localization-by-classification pipeline [40, 50], which trains a video-level classifier with category labels [32], and applies the trained classifier to each video **snippet**[1]. However, due to the lack of fine-grained annotations, the model may assign high confidence to incorrect snippets such as the contextual background, which typically has a high correlation with the video-level labels, or only focus on the salient snippets, leading to incomplete localization results. There are many studies [21, 23, 25] that tried to address this discrepancy between classification and localization, and one of the promising solutions is to generate and utilize pseudo labels.

The advantage of using pseudo labels is that snippets are supervised with snippet-wise labels instead of video-level labels. Existing works [28, 36, 46, 47] achieve remarkable results by introducing pseudo labels into this prob-

*Corresponding author.

[1]We view snippets as the smallest granularity since the high-level features of consecutive frames vary smoothly over time [12, 42]. In our work, we treat every 16 frames as a snippet

lem. A commonly used strategy for generating pseudo labels is to directly utilize the temporal class activation map (TCAM) generated in previous training iterations. Nevertheless, we would like to argue that the TCAMs are not desirable pseudo labels. During testing, our goal is to obtain the action boundaries, employing the TCAMs as training targets arise the discrepancy between training and testing because they are quite different from the actual action boundaries. An intuitive way to address this issue is to leverage the predicted action boundaries as pseudo labels. However, it is non-trivial to achieve this goal. First, current post-processing schemes, such as NMS, would induce a large amount of information loss and are not sufficient to obtain high-quality action boundaries for generating effective pseudo labels. Second, the predicted action instances are usually overlapped with each other and have different confidence scores, it is hard to assign the action categories and confidence scores for each snippet.

To address the above issues, we propose the following two modules. First, we propose a **Gaussian Weighted Instance Fusion** module to preserve information on the boundary distributions and produce high-quality action boundaries. Specifically, this module weightedly fuses the information of overlapped action instances. Each candidate action instance is treated as an instance sampled from a Gaussian distribution. The confidence score of each action instance is viewed as its probability of being sampled. Accordingly, we can obtain the most possible action boundaries and their confidence scores by estimating the means of Gaussians from those candidate action instances. In this way, we can produce better action boundaries, which in return help to generate more reasonable pseudo labels.

After generating high-quality action boundaries, we need to convert them into snippet-wise pseudo labels. To handle the overlapped action instances and assign snippets with proper confidence scores, we propose a **LinPro Pseudo Label Generation** module to formulate the process of pseudo-label generation as a ℓ_1-minimization problem. First, we restrict that the average score of snippets within an action boundary should be equal to the confidence score of this action instance. This constraint guarantees that we can maintain the information of confidence scores in the generated pseudo labels. Second, snippets within an action instance might be equivalent in terms of their contribution to the confidence score. Thus we require snippet-wise scores within each action instance to be uniform. Based on the two constraints, we formulate the pseudo label generation as an optimization problem and solve it to obtain pseudo labels that are consistent with our predicted action boundaries.

Furthermore, there is still one problem regarding to the use of pseudo labels. Since the generated pseudo labels can be fluctuating and inaccurate at early stage of training, without a proper self-correction mechanism, the model would keep generating wrong pseudo labels of high confidences at later training stages. To address this issue, we propose to utilize the Δ **pseudo labels**, instead of the original pseudo labels, as our training targets. We calculate the difference between the pseudo labels of consecutive training epochs as the Δ pseudo labels. In general, the model would provide more accurate predictions along with the training. In this way, the model will update its predictions toward the class with the confidence increasing instead of the class with the largest pseudo label value, and thus empowers the model with the ability of self-correction.

The contribution of this paper is four-fold. (a) We propose a Gaussian Weighted Instance Fusion module, which can effectively generate high-quality action boundaries. (b) We propose a novel LinPro Pseudo Label Generation strategy by transforming the process of pseudo-label generation into a ℓ_1-minimization problem. (c) We propose to utilize Δ pseudo labels to enable model with self-correction ability for the generated pseudo labels. (d) Compared with state-of-the-art methods, the proposed framework yields significant improvements of **1.9%** and **3.7%** in terms of average mAP on THUMOS14 and ActivityNet1.3, respectively.

2. Related Work

For the task of weakly supervised temporal action localization, the discrepancy between classification and localization has been observed by many researchers [21, 31, 32]. To alleviate this problem, many efforts have been made. We divide these methods into the following four categories: metric learning-based methods, erasing-based methods, multi-branch methods, and pseudo-label-based methods.

For the metric learning-based methods, previous works like W-TALC [37], 3C-Net [33], RPN [7] and A2CL-PT [30] utilize the center loss [41], clustering loss [45] and triplet loss [38]. In general, these methods [7,37] tend to obtain video-level features from the most discriminative snippets features. Therefore, these methods failed to capture snippets of less-discriminative features. Although several methods like RSKP [10] have tried to address this issue by propagating the knowledge of representative snippets, this method is also a double-edge sward. The knowledge propagation is a bi-lateral process, while we are passing information from discriminative snippets to less-discriminative ones, the transverse also takes place, which may hinder the model from learning high confidence snippets.

Another category is the erasing-based methods. These methods [30, 52] embrace the idea of exploring less-discriminative features from erasing discriminative ones. Originated from the adversarial complementary learning [49], these methods repeatedly find the most discriminative snippets and erase them. However, it is difficult to set proper thresholds for different classes with different complexities.

The third category mainly adopts the multi-branch architecture. These methods [9, 11, 21, 23, 25] share the similar idea as the erasing-based methods, while they differ from those methods by parallel processing. Likewise, it shares the same problem with the erasing-based methods.

The last category is the pseudo-label-based methods. These methods originated from Refine-Loc [36], which generates snippet-level hard pseudo labels. Then vari-

ous works follow this idea and try to generate more accurate pseudo labels. For example, works like EM-MIL [28], RSKP [10] fit the pseudo-label generation into an expectation-maximization framework. UGCT [46] uses an uncertainty-guided collaborative training strategy. However, most of these methods utilize the TCAM or variants of TCAM as the pseudo label, leading to discrepancy between training and testing. Besides, these methods cannot address the problem that pseudo labels can be inaccurate at the early training stage. And the inaccurate pseudo labels will keep generating wrong pseudo labels without the self-correction mechanism. In contrast to existing pseudo-label-based methods, we propose to generate pseudo labels from action boundaries, alleviating the discrepancy between training and testing. Besides, the introduction of Δ pseudo labels enables the model with the ability of self-correction.

3. Method

In this section, we detail the proposed method. An overall illustration of the pipeline is demonstrated in Figure 1.

Problem definition. Given N training videos $\{V_i\}_{i=1}^{N}$, during training, we can only access their action categories $\{\boldsymbol{y}_i\}_{i=1}^{N}$, where $\boldsymbol{y}_i$ is a binary vector indicating the presence/absence of each of K action classes. During inference, for a video, our target is to predict a set of action instances $\{(c, q, s, e)\}$, where c denotes the predicted action class, q is the confidence score, s and e represent the start time and end time of the action instance.

Overview. Our target is to narrow down the gap between training and testing by generating pseudo labels from the predicted action boundaries. First, we propose a Gaussian Weighted Instance Fusion module to obtain high-quality action boundaries. This module serves as an effective alternative to the non-maximum suppression (NMS), which weightedly fuses action instances that are overlapped with each other, rather than filtering low-confident instances.

After generating the high-quality action instances, we need to transfer them into snippet-wise pseudo labels. The key idea is generating pseudo labels that can preserve the confidence score within each action instance and make the scores within an action instance as uniform as possible. Therefore, we formulate it as a linear programming problem with the above two constraints to obtain the optimal snippet-wise pseudo labels.

Lastly, we propose the idea of Δ pseudo labels for better utilization of the generated pseudo labels. Instead of naively using the generated pseudo labels as training target, we take the differences of pseudo labels between two consecutive epochs as the final pseudo labels. In this way, we can capture the confidence change of the model for the pseudo labels, which represents the pseudo labels' reliability, and thus enable the model with the ability of self-correction.

Feature extraction and network design. Following previous works [34, 37], for each video V_i, every 16 consecutive frames is processed into a snippet-level feature. In our case, the Inflated 3D (I3D) [2] pre-trained on the Kinetics-400 dataset [14] is used to encode the video snippets. The output snippet-level features are in $\mathbb{R}^{2048}$, thus we convert the video of l snippets into a feature matrix of $F \in \mathbb{R}^{l\times 2048}$. As a plug-in method, our method can be applied to most of the existing approaches. Here, we adopt RSKP [10] as the backbone to obtain the temporal class activation map.

In the sections below, we first describe how to fuse candidate action instances to obtain high-quality action boundaries, then we elaborate on the generation of pseudo labels, and finally, we introduce our Δ pseudo labels.

3.1. Gaussian Weighted Instance Fusion

The classification head predicts labels for each individual snippet, it results in temporal class activate map (TCAM) $L \in \mathbb{R}^{l\times K}$, where l is the length of a given video and K is the number of classes. There are multiple ways of transforming the TCAM into action instances of interest. A commonly used way is employing multiple thresholds [8, 9, 16] to obtain redundant action instances and then resort to the non-maximum suppression (NMS) for duplicating highly-overlapped instances.

Traditional NMS is designed to only pick out the predicted action instances with high confidence scores, while throwing away those low-confident action instances. However, due to the weakly supervised setting, the confidence score lacks explicit supervision by snippet-level annotations. It is unwise to just trust the predictions with the high confidence scores and suppress other predictions with low confidence scores. Moreover, those suppressed action instances indicates different level of confidence at different locations. Thus large amount of information is lost due to the NMS procedure. To address the above issues, we present a novel Gaussian weighted fusion module to aggregate all the action instances who were suppressed by NMS.

Suppose for a class c, we have M candidate action instances $\mathbf{A} = \{a_1, \cdots, a_M\}$. Each action instance is of the $a_i = (c_i, q_i, s_i, e_i)$ where c_i is the predicted class, q_i is the confidence score, and s_i, e_i are the boundaries. We collect all the prediction instances, whose IoUs are greater than a predefined threshold h_{fuse} to the most confident predicted instance a_* in set $\mathbf{A}$. The index set of these prediction segments are denoted as $\mathcal{I}_*$

$$\mathcal{I}_* = \{k|\text{IoU}(a_k, a_*) > h_{fuse}\}. \tag{1}$$

Assuming the confidence scores $\{q_i\}$, start points $\{s_i\}$ and end points $\{e_i\}$ of the collected instances in index set $\mathcal{I}_*$ satisfy independent Gaussian distribution, whose probability density function (PDF) can be formulated as:

$$N(\blacktriangle) = \frac{1}{\sigma_\blacktriangle\sqrt{2\pi}}\exp(-\frac{(\blacktriangle - \mu_\blacktriangle)^2}{2\sigma_\blacktriangle^2}), \tag{2}$$

where $\blacktriangle \in \{q, s, e\}$. Now we tend to match the template Gaussian distribution as close as to our sampled predictions. Note that each action instance has a confidence score, if we regard the confidence scores as the un-normalized logits of

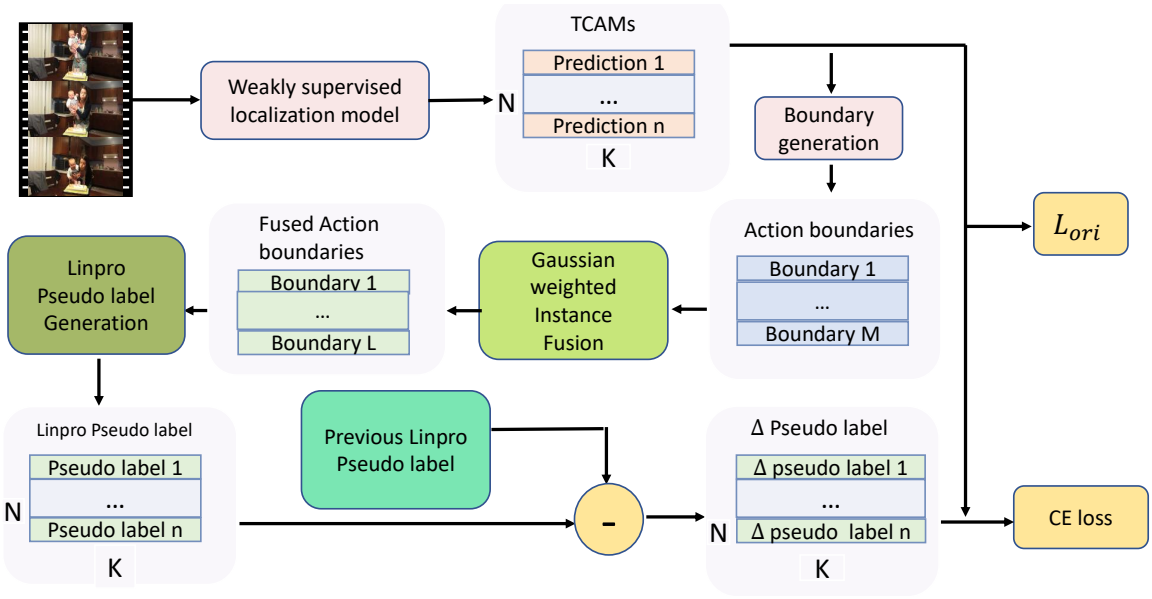

Figure 1. The overview of our method. We first feed the video into a weakly supervised action localization model to generate the temporal class activation map (TCAM). The obtained TCAM is transformed into action boundaries using multiple thresholds. We utilize the Gaussian Weighted Instance Fusion module to merge the candidate action instances into high-quality action boundaries. Then, we transform the action boundaries into snippet-wise pseudo labels by solving a linear programming problem. To enable the model with the ability of self-correction, we utilize the Δ pseudo labels, which are the differences between two pseudo labels generated at consecutive epochs, as the final pseudo labels. Note that, all the original losses of the weakly supervised localization model are preserved.

action instances, then we can compute the probability of sampling the k-th action instance as:

$$g_k = \frac{\exp\left(q_k/T\right)}{\sum_{i\in\mathcal{I}_*}\exp\left(q_i/T\right)}, \tag{3}$$

where T is a temperature hyper-parameter. By minimizing the cross-entropy between $N(\blacktriangle)$ and $\{g_k\}$ we have:

$$\mu_{\blacktriangle} = \sum_{i\in\mathcal{I}_*} \blacktriangle_i g_i, \tag{4}$$

where $\blacktriangle$ stands for $\{q, s, e\}$. Therefore, we can obtain the weightedly fused value μ_q, μ_s and μ_e, which are taken as the confidence score, start point and end point for the fused action instance. Then we delete all action instances of interest from $\mathcal{I}_*$ in $\mathbf{A}$, and repeat the above process until $\mathbf{A}$ is empty. Here we also tried some other distribution templates, such as exponential distribution, the Gaussian distribution achieves the best performance among them.

3.2. LinPro Pseudo Label Generation

After generating the high-quality action boundaries, we propose to transform them into snippet-wise pseudo labels. In this way, the video snippets are directly supervised by the final targets, *i.e.*, action boundaries, instead of the TCAM, and thus close up the gap between testing and training. A naive approach to achieve this is to directly assign a hard label to each snippet within the post-processed action instance. However, this naive approach suffers from several drawbacks. First, snippets from different action instances may have different confidence scores, assigning them with the same hard label brutally ignores this difference. Second, using hard labels actually overlooks the importance of the predicted confidence scores, which could represent the instances' quality measured by the model to some extent. Third, even after post-processing, some predicted instances are still overlapped, this naive approach will simply merge them together and result in false predictions.

To address the above issues, we propose an optimization-based method to generate desirable pseudo-labels. Since the confidence score are important information action instances carry on, we intend to find the pseudo label which can characterize the property of the confidence score of each action instance. To this end, we formulate this optimization problem based on two constraints: (1) the generated pseudo label should preserve the confidence score of each action instance. (2) The scores of the generated pseudo labels should be uniform. The first constraint is easy to comprehend because we intend to maintain the confidence score of each action instance. As for the second constraint, it follows from the fact that most of the snippets within an action instance should be equivalent for classification and localization.

To address the first constraint, we convert each action instance into a linear constraint. Specifically, given the n action instances of the class c, we first construct n weighting vectors $\{W_j \in \mathbb{R}^l | j = 1 \dots n\}$, where l is the length of the video, each of which serves as the constraint for the corresponding action instance. Here, we follow Inner-Outer

Contrast score [16, 39] to construct the weighting vectors, by dividing the whole video into the inner areas, outer areas, and background. In specific, for W_j, we set its i-th element $W_{j,i}$ as 1 if the i-th snippet is in the inner area of action instance j, *i.e.*, $s_j \leq i \leq e_j$. Likewise, we set it as -1 if it lies in the outer area, *i.e.*, $s_j - \alpha(e_j - s_j) \leq i < s_j$ or $e_j \leq i < e_j + \alpha(e_j - s_j)$), otherwise 0 for background. This process is detailedly shown in Figure 2. From such a construction, we have the following linear equations for the n action instances:

$$\begin{aligned} q_{c,1} &= W_1^T g_c, \\ &\vdots \\ q_{c,n} &= W_n^T g_c, \end{aligned} \tag{5}$$

where $g_c \in \mathbb{R}^l$ is the pseudo label to be generated for the class c, and $q_{c,i}$ is the confidence score of class c for the i-th action instance. To this end, we constrain the generated pseudo label to maintain the Inner-Outer Contrast scores of action instances. Besides, the larger the value of $\sum_{j=1}^{n} W_{j,i}$, the more important the i-th snippet is. It represents the i-th snippet lies in more inner areas and fewer outer areas. Intuitively, we want to generate the pseudo label, which has a larger value at more important snippets. To achieve this, we minimize the ℓ_1-norm of the pseudo-label g_c. The optimization problem can be finally formulated as:

$$\begin{aligned} \hat{g}_c &= \textbf{argmin} \|g_c\|_1 \\ \textbf{s.t. } q_c &= W^T g_c \\ g_c &\geq 0, \end{aligned} \tag{6}$$

where $q_c = (q_{c,1}, \ldots, q_{c,n})$ and $W = [W_1; \ldots; W_n] \in \mathbb{R}^{l \times n}$. Even though, the above optimization target can achieve minimizing ℓ_1-norm of g_c, it does not guarantee a uniform-shaped pseudo label. To satisfy the second constraint, we first collect the snippets, which are equivalent for the optimization problem. Here, we define the two snippets i, j as equivalent, if the i-th and the j-th rows of W are the same, which means they fall on the same inner area/outer area/background area, or the same overlapped area. For those equivalent snippets, we average their scores in the solved pseudo label $\hat{g}_c$ of Eq. (6), and re-assign them with the average value. In this way, we transform the pseudo label $\hat{g}_c$ into a more uniform form while satisfying the optimization target of Eq. (6).

For each of the K classes, we conduct the above procedure to obtain its pseudo label. After that, we concatenate all the pseudo labels of K classes together into the final pseudo label $G \in \mathbb{R}^{l \times K}$ for the input video.

3.3. Δ Pseudo Label

Although the generated pseudo labels can capture the boundaries of predicted action instances and preserve the information of confidence scores, they still have certain drawbacks when used directly. If we apply our pseudo label G directly to the TCAM L with cross-entropy (CE) loss,

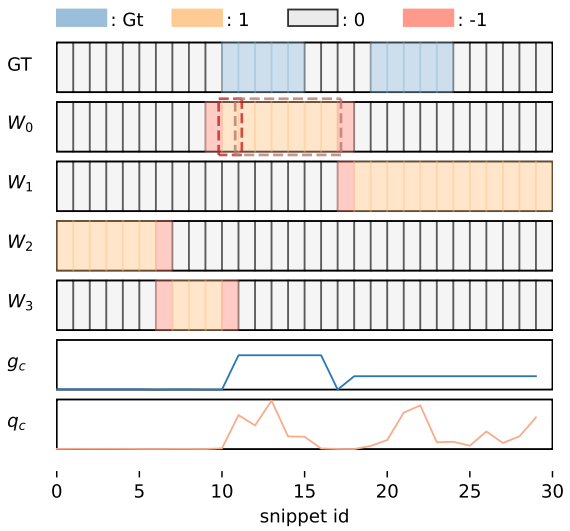

Figure 2. Illustration of the process of pseudo label generation. GT is the ground truth snippet-wise label, where the blue snippet denotes the action locations. W_0, W_1, W_2, W_3 are four constraints generated from four action instances. The yellow, white, and red snippets denote the constraints weights of $1, 0, -1$ respectively. q_c is the predicted snippet-wise score, and g_c is the generated pseudo label. The snippets in each dashed rectangle are equivalent.

i.e., $\mathcal{L}_{ce} = \sum \text{CE}(\text{softmax}(L_j), G_j)$, where j denotes the j-th snippet, the negative gradient of the CE loss with respect to the snippet j and the class c is:

$$-\frac{\partial \mathcal{L}_{ce}}{\partial P_{j,c}} = G_{j,c}(1 - P_{j,c}), \tag{7}$$

where $P_j = \text{softmax}(L_j)$. Since $(1 - P_{j,c})$ is always positive, this negative gradient indicates that we are pushing the TCAM towards the direction of $G_{j,c}$, which is also always positive. What's worse, at the early stage of training, the generated pseudo labels are usually fluctuating and inaccurate, it would force the model to keep generating wrong pseudo labels of high confidence at later training stages.

To alleviate this issue, we propose to utilize the Δ pseudo labels. In specific, we take the difference of two pseudo labels, which are collected at two consecutive epochs of pseudo label generation, as the Δ pseudo label, which can be obtained as:

$$\Delta G^t = G^t - G^{t-1}, \tag{8}$$

where upper-script t stands for the pseudo label generated at the t-th time. Now, applying the cross entropy loss, the negative gradient turns into:

$$-\frac{\partial \mathcal{L}_{ce}}{\partial P_{j,c}} = \Delta G^t_{j,c}(1 - P_{j,c}). \tag{9}$$

In this way, the training loss is driving the TCAM to the direction of $\Delta G^t_{j,c}$, which could be either positive or negative. Since the model in general tends to provide more accurate

predictions when the training goes on, it can be expected that when the snippet j is a part of the action category i, its $\Delta G^t_{j,c}$ could be positive. In contrast, if the snippet j does not belong to the class i, its $\Delta G^t_{j,c}$ could be negative since the model is prone to lower its score. Therefore, this mechanism mitigates the issue of generating inaccurate pseudo labels at the early stage of training, by considering the relative changes of pseudo labels rather than the absolute values, and thus empowers the model with the ability of self-correction, so as to obtain better performance.

4. Experiments

4.1. Datasets

THUMOS14 [13]. THUMOS14 is a dataset with 20 classes of actions. In our experiments, we consider of subset of THUMOS14 which has frame-wise annotations for all 20 classes of actions. We train our model on the 200 validation video and evaluate it on the 212 testing video. Note that we do not use the frame-wise label at training stage.

ActivityNet1.3 [1]. ActivityNet1.3 is a dataset that contains 200 daily activities. This dataset provides 10,024 videos for training, 4,926 validation videos and 5,044 testing ones. Our model is trained on the training set and tested on the validation set.

4.2. Implementation Details

Model details. We pre-process each video into snippets, which are extracted into 2048-d features by the I3D model pre-trained on Kinetics-400 [2]. Then we strictly follow the network design of [10] for model designing.

Training details. Our method is trained with a mini-batch size of 10 and 128 with Adam [15] optimizer for THUMOS14 and ActivityNet1.3, respectively. The hyper-parameter of temperature T for Gaussian weighted fusion is set as 0.1. At the early training stage, the model is insufficient to generate high quality pseudo labels, thus we start to generate pseudo labels from epoch 200, and renew the pseudo label at epochs 215, 230, 245, 270 and 290. The training procedure stops at 350 epochs with the learning rate 5×10^{-5}. Unless state otherwise, this is our default training setting in the following experiments.

Testing details. The whole sequence of a video is used as testing input. Our model produces snippet-level predictions, and we simply up-sample the predictions to match the original frame rate. Following [16], we use a set of thresholds to obtain the predicted action instances. After that, we apply our Gaussian weighted fusion strategy instead of non-maximum suppression to generate the more accurate action instances. During testing, the hyper-parameter of temperature T for Gaussian weighted fusion is set as 0.03.

4.3. Comparison with State-of-the-art Methods

In this section, we compare our method with state-of-the-art weakly supervised methods. Meanwhile, we also compare it several fully supervised methods. The results are shown in Table 1 and Table 2. On the THUMOS14 [13] benchmark, our method largely outperforms the previous weak-supervised approaches in almost every metric. Besides, on the important criterions: average mAP (0.1:0.5), average mAP (0.3:0.7), average mAP (0.1:0.7), we surpass the state-of-the-art method, DELU [4], by 3.2%, 1.5% and 1.9%, respectively. Most excitingly, our method even outperforms some recent fully-supervised methods on the mAP (0.1:0.5) and average mAP (0.3:0.7) metrics. Some other weakly supervised methods (*i.e.*, Weak † in Table 2) utilize additional weak supervisions, such as action frequency, our method still outperforms these methods.

Table 1. Results on ActivityNet1.3 validation set. AVG indicates the average mAP at IoU thresholds 0.5:0.05:0.95.

Method	mAP @ IoU			
	0.5	0.75	0.95	AVG
TAL-Net [3]	38.2	18.3	1.3	20.2
BSN [20]	46.5	30.0	8.0	30.0
GTAN [26]	52.6	34.1	8.9	34.3
BaS-Net (I3D) [16]	34.5	22.5	4.9	22.2
A2CL-PT (I3D) [30]	36.8	22.0	5.2	22.5
ACM-BANet (I3D) [31]	37.6	24.7	6.5	24.4
TSCN (I3D) [47]	35.3	21.4	5.3	21.7
WUM (I3D) [17]	37.0	23.9	5.7	23.7
TS-PCA (I3D) [22]	37.4	23.5	5.9	23.7
UGCT (I3D) [46]	39.1	22.4	5.8	23.8
AUMN (I3D) [27]	38.3	23.5	5.2	23.5
FAC-Net (I3D) [9]	37.6	24.2	6.0	24.0
RSKP (I3D) [10]	40.6	24.6	5.9	25.0
ASM-Loc (I3D) [5]	41.0	24.9	6.2	25.1
Ours	**43.4**	**28.8**	**9.9**	**28.8**

On the larger ActivityNet1.3 [1] dataset, our method outperforms all existing weakly supervised methods by a significant margin. Compared with previous methods, we consistently achieve a gain about 3.0% on every metric. In terms of the average mAP, our method obtains a 3.7% gain.

4.4. Ablation Study

Our ablation studies are conducted on the THUMOS14 benchmark. Unless explicitly stated, we follow our default setting in Sec. 4.2.

Gaussian Weighted Instance Fusion As the initiative procedure in our pipeline, the Gaussian Weighted Instance Fusion module is of great importance. As pointed out previously, besides the Gaussian distribution, other distributions are also suitable for our fusion strategy.

In Table 3, we conduct several experiments to verify the effectiveness of our fusion strategy during the training stage. There are four different distributions: Gaussian distribution, Uniform distribution, Exponential distribution, and T-distribution to be considered. All these experiments are conducted with the same set of candidate action instances. Meanwhile, the LinPro Pseudo Label Generation module and Δ pseudo-label is turned off. We use the non-maximum suppression (NMS) approach as our baseline

Table 2. Comparisons of detection performance on THUMOS14. UNT and I3D represent UntrimmedNet features and I3D features, respectively. † means that the method utilizes additional weak supervisions, *e.g.*, action frequency.

Supervision	Method	Feature	mAP @ IoU (%)							AVG (0.1:0.5)	AVG (0.3:0.7)	AVG (0.1:0.7)
			0.1	0.2	0.3	0.4	0.5	0.6	0.7			
Full	SSN [51] (ICCV'17)	-	60.3	56.2	50.6	40.8	29.1	-	-	49.6	-	-
	BSN [20] (ECCV'18)	-	-	-	53.5	45.0	36.9	28.4	20.0	-	36.8	-
	GTAN [26] (CVPR'19)	-	69.1	63.7	57.8	47.2	38.8	-	-	55.3	-	-
Weak †	STAR [44] (AAAI'19)	I3D	68.8	60.0	48.7	34.7	23.0	-	-	47.0	-	-
	3C-Net [33] (ICCV'19)	I3D	59.1	53.5	44.2	34.1	26.6	-	8.1	43.5	-	-
Weak	CleanNet [24] (ICCV'19)	UNT	-	-	37.0	30.9	23.9	13.9	7.1	-	22.6	-
	TSCN [47] (ECCV'20)	I3D	63.4	57.6	47.8	37.7	28.7	19.4	10.2	47.0	28.8	37.8
	EM-MIL [28] (ECCV'20)	I3D	59.1	52.7	45.5	36.8	30.5	22.7	**16.4**	45.0	30.4	37.7
	A2CL-PT [30] (ECCV'20)	I3D	61.2	56.1	48.1	39.0	30.1	19.2	10.6	46.9	29.4	37.8
	HAM-Net [11] (AAAI'21)	I3D	65.4	59.0	50.3	41.1	31.0	20.7	11.1	49.4	30.8	39.8
	WUM [17] (AAAI'21)	I3D	67.5	61.2	52.3	43.4	33.7	22.9	12.1	51.6	32.9	41.9
	AUMN [27] (CVPR'21)	I3D	66.2	61.9	54.9	44.4	33.3	20.5	9.0	52.1	32.4	41.5
	CoLA [48] (CVPR'21)	I3D	66.2	59.5	51.5	41.9	32.2	22.0	13.1	50.3	32.1	40.9
	TS-PCA [22] (CVPR'21)	I3D	67.6	61.1	53.4	43.4	34.3	24.7	13.7	52.0	33.9	42.6
	UGCT [46] (CVPR'21)	I3D	69.2	62.9	55.5	46.5	35.9	23.8	11.4	54.0	34.6	43.6
	ASL [29] (CVPR'21)	I3D	67.0	-	51.8	-	31.1	-	11.4	-	-	-
	CO_2-Net [6] (MM'21)	I3D	70.1	63.6	54.5	45.7	38.3	26.4	13.4	54.4	35.6	44.6
	D2-Net [32] (ICCV'21)	I3D	65.7	60.2	52.3	43.4	36.0	-	-	51.5	-	-
	FAC-Net [9] (ICCV'21)	I3D	67.6	62.1	52.6	44.3	33.4	22.5	12.7	52.0	33.1	42.2
	RSKP [10] (CVPR'22)	I3D	71.3	65.3	55.8	47.5	38.2	25.4	12.5	55.6	35.9	45.1
	ASM-Loc [5] (CVPR'22)	I3D	71.2	65.5	57.1	46.8	36.6	25.2	13.4	55.4	35.8	45.1
	Li et al. [18] (MM'22)	I3D	69.7	64.5	58.1	49.9	39.6	27.3	14.2	56.3	37.8	46.1
	DELU [4] (ECCV'22)	I3D	71.5	66.2	56.5	47.7	40.5	**27.4**	15.3	56.5	37.4	46.4
	Ours	I3D	**74.0**	**69.4**	**60.7**	**51.8**	**42.7**	26.2	13.1	**59.7**	**38.9**	**48.3**

Table 3. Localization results of using different distributions for action instance fusion during training and testing. The non-maximum suppression (NMS) is used as *baseline* method.

Distribution	mAP @ IoU					
	Training			Testing		
	0.5	0.7	AVG	0.5	0.7	AVG
Baseline	36.2	12.2	44.9	36.6	11.1	44.9
Uniform	24.2	8.1	31.2	33.8	9.7	41.5
Exponential	36.2	12.3	45.3	41.1	12.4	47.1
t-distribution	36.7	12.7	45.6	41.2	11.5	47.1
Gaussian	36.6	12.7	45.6	41.2	11.6	47.1

method. As we can see, our method achieves the best performance using the Gaussian distribution, the gain is about 0.7% compared with the baseline method. This is because Gaussian distribution generally exists in a stochastic process. We also note that the uniform distribution deteriorates the performance because it significantly deviates from the ground truth distribution. Although the t-distribution exhibits a similar performance with Gaussian, we do not encourage using it. Since using t-distribution has no analytic form of fusing formulas, we have to solve a transcendental equation through Newton iteration.

Similarly, we conduct experiments in Table 3 for evaluating the effectiveness of our fusion strategy during testing. With exactly the same distributions in the previous experiments, we choose NMS as our baseline. Specially, the baseline model is trained without the LinPro Pseudo Label Generation module and the pseudo labels for a fair comparison. From Table 3, we can see that the Gaussian distribution still achieves the best performance and acquire a gain as high as 2.3% at testing time. Intriguingly, we notice that the exponential distribution and t-distribution achieve the same performance. We argue that this is because those three distributions are very similar when the hyper-parameter of temperature is low during testing.

Although our method is almost hyper-parameter-free, we still suffer from the impact of the template distribution parameter temperature T. For this reason, we study the temperature's impact on our method. We conduct the following experiments for two aspects. (1) We study the temperature's impact during the training stage, these experiments are carried out with the default setting. To ensure fair comparison, we do not use the fusion strategy during testing. (2) We study how the temperature influences the post-processing during testing. Also, we use the default settings with the training temperature set as 0.1.

From the results in Figure 3, we claim that our method is not sensitive to the hyper-parameter of temperature in a large range (from 0.05 to 0.2) during the training stage. Besides, our method suffers some performance loss when the temperature is too low. Under this circumstance, the candidate action instances of high sampling probability would be overwhelming during the fusion process, we cannot collect information from regions where the probability is low.

Similarly, during testing, the temperature impact on our method is also limited. There is a large region where our method doesn't change much as the temperature varies. One may notice that our method achieved 48.5% in terms of mAP as temperature 0.01, however, we do not report it as our main result because we believe this temperature is too

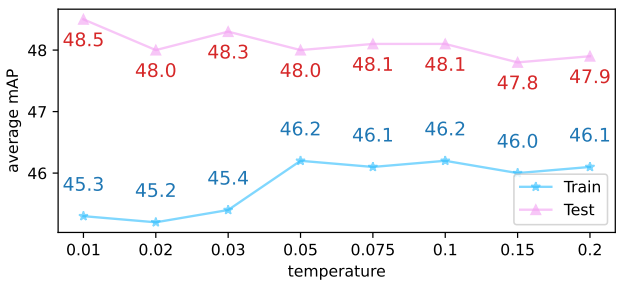

Figure 3. Impact of temperature on Gaussian weighted Instance Fusion module during training and testing.

Table 4. Evaluation of each component of our method.

componet	mAP @ IoU			
	0.3	0.5	0.7	AVG
baseline	55.8	38.2	12.5	45.1
+ LinPro Pseudo Label Generation	57.4	36.2	12.2	44.9
+ Gaussian weighted fusion	58.2	36.6	12.7	45.6
+ Δ pseudo label	58.4	37.6	12.2	46.2
+ Gaussian weighted fusion (testing)	60.7	42.7	13.1	48.3

Table 5. The detection results of applying our method to existing methods. *Embedding* means we add a learnable network after the backbone network to learn the video features. The results of the original methods are reproduced.

Method	mAP @ IoU			
	0.3	0.5	0.7	AVG
STPN [34] + embedding	38.4	19.1	4.7	28.4
STPN + embedding + Ours	41.6	22.1	6.8	31.6 $_{\uparrow 2.2}$
BM [35]	45.2	26.2	8.7	35.3
BM + Ours	47.4	28.5	10.3	37.3 $_{\uparrow 2.0}$
WUM [17]	51.0	32.8	10.9	40.4
WUM + Ours	53.2	34.0	11.4	42.3 $_{\uparrow 1.9}$
FAC-Net [9]	53.2	34.4	13.7	42.9
FAC-Net + Ours	56.4	37.2	13.0.	44.8 $_{\uparrow 1.9}$

radical. Instead, we choose temperature 0.03 as our default setting since it lies in the center of our temperature interval.

Performance gain of each component. Our method consists of three components, *i.e.*, Gaussian weighted instance fusion, LinPro pseudo label generation, and Δ pseudo label. Here, we study the performance gain of each component and their combinations, we take the RSKP [10] as our baseline method, and gradually add the three components. As we can see in Table 4, when only adopting the LinPro pseudo label generation to the NMS-generated action boundaries, the performance drops, indicating the NMS cannot generate high-quality action boundaries. After adding the Gaussian weighted fusion module, the performance increases by 1.5% over the baseline. Further benefitted from the introduction of Δ pseudo labels, our method can reach the average mAP of 46.2%. Finally, an evident gain of 2.1% can be obtained when the Gaussian weighted fusion strategy is adopted as post-processing during testing.

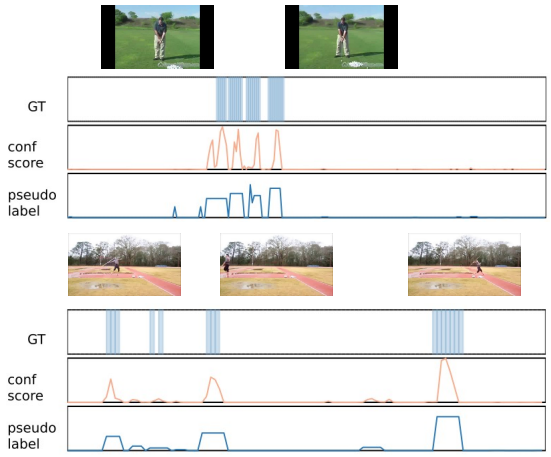

Figure 4. Illustration of the detection results of *GolfSwing* and *JavelinThrow*. GT stands for the ground truth, conf score is the predicted confidence score of the given action. The pseudo labels are generated by our Linpro pseudo label generation module.

Integrating our modules to existing methods. In Table 5, we plug our proposed pipeline into some existing methods. The default settings of these methods are used for fair comparisons. As we can see, our method can consistently improve the performances of the previous methods.

Qualitative results. We visualize some examples of the confidence scores and Linpro pseudo labels generated by our method. As we can see from Figure 4, the generated pseudo labels preserve the confidence score within each action instances, meanwhile they are as uniform as possible.

5. Conclusion

In this paper, we propose a novel framework to generate better pseudo labels from action boundaries. We first propose a Gaussian weighted instance fusion module to obtain high-quality action boundaries. After that, we generate the pseudo labels from action boundaries by solving the the optimization problem under the constraints in terms of the confidence scores of action instances. Finally, we propose to utilize the Δ pseudo-label for introducing a self-correction mechanism into the model. Our method achieves state-of-the-art performance on THUMOS14 and ActivityNet1.3 and can consistently improve the performance of existing methods.

6. Acknowledgments

This project is funded in part by National Key RD Program of China Project 2022ZD0161100, by the Centre for Perceptual and InteractiveIntelligence (CPII) Ltd under the Innovation and TechnologyCommission (ITC)'s InnoHK, by General Research Fund Project 14204021 and Reseasrch Impact Fund Project R5001-18 of Hong Kong RGC.

References

[1] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, pages 961–970. IEEE, 2015. 6

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 6299–6308. IEEE, 2017. 3, 6

[3] Yu-Wei Chao, Sudheendra Vijayanarasimhan, Bryan Seybold, David A Ross, Jia Deng, and Rahul Sukthankar. Rethinking the faster r-cnn architecture for temporal action localization. In *CVPR*, pages 1130–1139. IEEE, 2018. 1, 6

[4] Mengyuan Chen, Junyu Gao, Shicai Yang, and Changsheng Xu. Dual-evidential learning for weakly-supervised temporal action localization. In *European Conference on Computer Vision*, pages 192–208. Springer, 2022. 6, 7

[5] Bo He, Xitong Yang, Le Kang, Zhiyu Cheng, Xin Zhou, and Abhinav Shrivastava. Asm-loc: Action-aware segment modeling for weakly-supervised temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13925–13935, 2022. 6, 7

[6] Fa-Ting Hong, Jia-Chang Feng, Dan Xu, Ying Shan, and Wei-Shi Zheng. Cross-modal consensus network for weakly supervised temporal action localization. In *ACM MM*. ACM, 2021. 7

[7] Linjiang Huang, Yan Huang, Wanli Ouyang, and Liang Wang. Relational prototypical network for weakly supervised temporal action localization. In *AAAI*, volume 34, pages 11053–11060, 2020. 2

[8] Linjiang Huang, Yan Huang, Wanli Ouyang, and Liang Wang. Modeling sub-actions for weakly supervised temporal action localization. *IEEE TIP*, 30:5154–5167, 2021. 3

[9] Linjiang Huang, Liang Wang, and Hongsheng Li. Foreground-action consistency network for weakly supervised temporal action localization. In *ICCV*, pages 8002–8011. IEEE, 2021. 2, 3, 6, 7, 8

[10] Linjiang Huang, Liang Wang, and Hongsheng Li. Weakly supervised temporal action localization via representative snippet knowledge propagation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3, 6, 7, 8

[11] Ashraful Islam, Chengjiang Long, and Richard Radke. A hybrid attention mechanism for weakly-supervised temporal action localization. In *AAAI*, volume 35, pages 1637–1645, 2021. 2, 7

[12] Dinesh Jayaraman and Kristen Grauman. Slow and steady feature analysis: higher order temporal coherence in video. In *CVPR*, pages 3852–3861. IEEE, 2016. 1

[13] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. http://crcv.ucf.edu/THUMOS14/, 2014. 6

[14] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 3

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Pilhyeon Lee, Youngjung Uh, and Hyeran Byun. Background suppression network for weakly-supervised temporal action localization. In *AAAI*, volume 34, pages 11320–11327, 2020. 3, 5, 6

[17] Pilhyeon Lee, Jinglu Wang, Yan Lu, and Hyeran Byun. Weakly-supervised temporal action localization by uncertainty modeling. In *AAAI*, volume 35, pages 1854–1862, 2021. 6, 7, 8

[18] Ziqiang Li, Yongxin Ge, Jiaruo Yu, and Zhongming Chen. Forcing the whole video as background: An adversarial learning strategy for weakly temporal action localization. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 5371–5379, 2022. 7

[19] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *ICCV*, pages 3889–3898, 2019. 1

[20] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*, pages 3–19. Springer, 2018. 1, 6, 7

[21] Daochang Liu, Tingting Jiang, and Yizhou Wang. Completeness modeling and context separation for weakly supervised temporal action localization. In *CVPR*, pages 1298–1307. IEEE, 2019. 1, 2

[22] Yuan Liu, Jingyuan Chen, Zhenfang Chen, Bing Deng, Jianqiang Huang, and Hanwang Zhang. The blessings of unlabeled background in untrimmed videos. In *CVPR*, pages 6176–6185. IEEE, 2021. 6, 7

[23] Ziyi Liu, Le Wang, Wei Tang, Junsong Yuan, Nanning Zheng, and Gang Hua. Weakly supervised temporal action localization through learning explicit subspaces for action and context. In *AAAI*, volume 35, pages 2242–2250, 2021. 1, 2

[24] Ziyi Liu, Le Wang, Qilin Zhang, Zhanning Gao, Zhenxing Niu, Nanning Zheng, and Gang Hua. Weakly supervised temporal action localization through contrast based evaluation networks. In *ICCV*, pages 3899–3908. IEEE, 2019. 7

[25] Ziyi Liu, Le Wang, Qilin Zhang, Wei Tang, Junsong Yuan, Nanning Zheng, and Gang Hua. Acsnet: Action-context separation network for weakly supervised temporal action localization. In *AAAI*, volume 35, pages 2233–2241, 2021. 1, 2

[26] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *CVPR*, pages 344–353. IEEE, 2019. 6, 7

[27] Wang Luo, Tianzhu Zhang, Wenfei Yang, Jingen Liu, Tao Mei, Feng Wu, and Yongdong Zhang. Action unit memory network for weakly supervised temporal action localization. In *CVPR*, pages 9969–9979. IEEE, 2021. 6, 7

[28] Zhekun Luo, Devin Guillory, Baifeng Shi, Wei Ke, Fang Wan, Trevor Darrell, and Huijuan Xu. Weakly-supervised action localization with expectation-maximization multi-instance learning. In *ECCV*, pages 729–745. Springer, 2020. 1, 3, 7

[29] Junwei Ma, Satya Krishna Gorti, Maksims Volkovs, and Guangwei Yu. Weakly supervised action selection learning in video. In *CVPR*, pages 7587–7596. IEEE, 2021. 7

[30] Kyle Min and Jason J Corso. Adversarial background-aware loss for weakly-supervised temporal activity localization. In *ECCV*, pages 283–299. Springer, 2020. 2, 6, 7

[31] Md Moniruzzaman, Zhaozheng Yin, Zhihai He, Ruwen Qin, and Ming C Leu. Action completeness modeling with background aware networks for weakly-supervised temporal action localization. In *ACM MM*, pages 2166–2174. ACM, 2020. 2, 6

[32] Sanath Narayan, Hisham Cholakkal, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. D2-net: Weakly-supervised action localization via discriminative embeddings and denoised activations. In *ICCV*, pages 13608–13617. IEEE, 2021. 1, 2, 7

[33] Sanath Narayan, Hisham Cholakkal, Fahad Shahbaz Khan, and Ling Shao. 3c-net: Category count and center loss for weakly-supervised action localization. In *ICCV*, pages 8679–8687. IEEE, 2019. 2, 7

[34] Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*, pages 6752–6761. IEEE, 2018. 3, 8

[35] Phuc Xuan Nguyen, Deva Ramanan, and Charless C Fowlkes. Weakly-supervised action localization with background modeling. In *ICCV*, pages 5502–5511. IEEE, 2019. 8

[36] Alejandro Pardo, Humam Alwassel, Fabian Caba, Ali Thabet, and Bernard Ghanem. Refineloc: Iterative refinement for weakly-supervised action localization. In *Winter Conference on Applications of Computer Vision*, pages 3319–3328. IEEE, 2021. 1, 2

[37] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification. In *ECCV*, pages 563–579. Springer, 2018. 2, 3

[38] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823. IEEE, 2015. 2

[39] Zheng Shou, Hang Gao, Lei Zhang, Kazuyuki Miyazawa, and Shih-Fu Chang. Autoloc: Weakly-supervised temporal action localization in untrimmed videos. In *ECCV*, pages 154–171. Springer, 2018. 5

[40] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *CVPR*, pages 4325–4334. IEEE, 2017. 1

[41] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, pages 499–515. Springer, 2016. 2

[42] Laurenz Wiskott and Terrence J Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 14(4):715–770, 2002. 1

[43] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *ICCV*, pages 5783–5792. IEEE, 2017. 1

[44] Yunlu Xu, Chengwei Zhang, Zhanzhan Cheng, Jianwen Xie, Yi Niu, Shiliang Pu, and Fei Wu. Segregated temporal assembly recurrent networks for weakly supervised multiple action detection. In *AAAI*, volume 33, pages 9070–9078, 2019. 7

[45] Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Robust classification with convolutional prototype learning. In *CVPR*, pages 3474–3482. IEEE, 2018. 2

[46] Wenfei Yang, Tianzhu Zhang, Xiaoyuan Yu, Tian Qi, Yongdong Zhang, and Feng Wu. Uncertainty guided collaborative training for weakly supervised temporal action detection. In *CVPR*, pages 53–63. IEEE, 2021. 1, 3, 6, 7

[47] Yuanhao Zhai, Le Wang, Wei Tang, Qilin Zhang, Junsong Yuan, and Gang Hua. Two-stream consensus network for weakly-supervised temporal action localization. In *ECCV*, pages 37–54. Springer, 2020. 1, 6, 7

[48] Can Zhang, Meng Cao, Dongming Yang, Jie Chen, and Yuexian Zou. Cola: Weakly-supervised temporal action localization with snippet contrastive learning. In *CVPR*, pages 16010–16019. IEEE, 2021. 7

[49] Xiaolin Zhang, Yunchao Wei, Jiashi Feng, Yi Yang, and Thomas S Huang. Adversarial complementary learning for weakly supervised object localization. In *CVPR*, pages 1325–1334. IEEE, 2018. 2

[50] Tao Zhao, Junwei Han, Le Yang, and Dingwen Zhang. Equivalent classification mapping for weakly supervised temporal action localization. *arXiv preprint arXiv:2008.07728*, 2020. 1

[51] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *ICCV*, pages 2914–2923. IEEE, 2017. 1, 7

[52] Jia-Xing Zhong, Nannan Li, Weijie Kong, Tao Zhang, Thomas H Li, and Ge Li. Step-by-step erasion, one-by-one collection: A weakly supervised temporal action detector. In *ACM MM*, pages 35–44. ACM, 2018. 2