Learning Action Completeness from Points for Weakly-supervised Temporal Action Localization – Supplementary Material –

Pilhyeon Lee[1] Hyeran Byun[1,2*]

[1]Department of Computer Science, Yonsei University
[2]Graduate school of AI, Yonsei University
{lph1114, hrbyun}@yonsei.ac.kr

A. Regarding Point-level Supervision

In this paper, we tackle temporal action localization under point-level supervision. Here, timestamp are denoted by "points" in the temporal axis, whereas "points" have also been widely used to represent spatial pixels in the literature. Bearman *et al*. [1] introduce the first weakly-supervised semantic segmentation framework that takes as supervision a single annotated pixel for each object. Since that work, a great amount of efforts [4, 5, 6, 14, 15] have been endeavored to utilize point-level supervision to solve various segmentation tasks in images or videos, thanks to its affordable annotation cost. Meanwhile, there are also attempts to employ point-level supervision to train object detectors [9, 12, 13]. On the other hand, spatial points have also been explored to provide supervision for the weakly-supervised spatio-temporal action localization task [10, 11].

We remark that the definition of "point" in our problem setting is based on the temporal dimension, differing from that of the work above.

B. Greedy Optimal Sequence Search

As discussed in the main paper, the search space of optimal sequence selection would grow exponentially as the length of the input video increases, which makes the optimal sequence search intractable. To bypass the cost issue, we design a greedy algorithm that makes locally optimal choices at each step under a fixed budget. Specifically, we process an input video in a sequential way, taking one segment at a timestep. At each timestep t, we consider all possible t-length candidate sequences consistent with point labels, and compute their completeness scores by averaging contrast scores of the action and background instances constituting the sequences (Eq. (6) of the main paper). In this calculation, we do not include the ongoing (*i.e*., not terminated) instance, as it is infeasible to derive its contrast score without looking ahead to the future. Afterwards, we keep only the top α (budget size) candidates regarding the completeness scores. When the step t reaches the end of the video, we terminate the algorithm and select the optimal sequence with the highest score. In this way, we can save a large amount of the computational cost, thereby making the search process tractable. The pseudo-code of our algorithm for class c is described in Algorithm 1.

$\alpha \longrightarrow$	1	5	10	25	50	100
mAP@AVG (%)	51.3	52.5	52.6	52.8	52.7	52.7
Execution time (sec)	0.683	1.343	2.151	4.398	8.512	16.769

Table 1: Analysis on the budget size α on THUMOS'14. We provide the execution times as well as the average mAPs under IoU thresholds 0.1:0.1:0.7 with varying α from 1 to 100. The average execution time for optimal sequence selection per epoch is reported in seconds.

Since the budget α affects the computational cost as well as the performance, we investigate several different budget sizes on THUMOS'14. For the computational cost, we train the model for 100 epochs and report the average execution time of optimal sequence selection for an epoch (*i.e*., 200 training videos). The selection is implemented in multiprocessing with 16 worker processes and performed on a single AMD-3960X Threadripper CPU. Table 1 shows the average mAPs (%) and the execution times (sec) with varying α. As can be expected, when the budget increases, the computational cost grows in a nearly linear way. Besides, when α is set to a too-small value (*e.g*., 1), the selected optimal sequence is likely to be a local optimum, leading to a significant performance drop. On the other hand, the performance differences are insignificant when α is larger than 5. This indicates that the model is fairly robust against the budget size and a not-too-small α is sufficient to find the sequences that can provide helpful completeness guidance to the model. In practice, we set α to 25, as it achieves the best performance at an affordable cost of fewer than 5 seconds for processing the whole training videos.

*Corresponding author

Algorithm 1 Greedy Optimal Sequence Search

Input: class-specific action points (ascending) $\mathcal{B}_c^{\text{act}} = \{t_i^{\text{act}}\}_{i=1}^{M_c^{\text{act}}}$, pseudo background points (ascending) $\mathcal{B}^{\text{bkg}} = \{t_j^{\text{bkg}}\}_{j=1}^{M^{\text{bkg}}}$, the number of class-specific action points M_c^{act}, the number of pseudo background points M^{bkg}, fixed budget size α

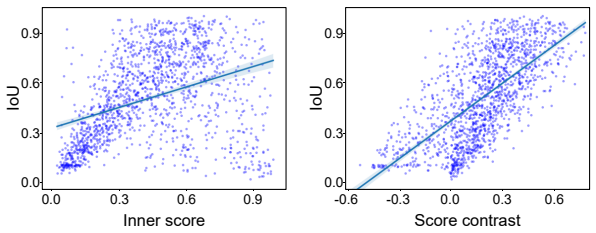
Output: optimal sequence π_c^*

// Definition: $\pi_c = \{(s_n, e_n, z_n)\}_{n=1}^{N}, \mathcal{S}_c = \{(\pi_c, \mathcal{R}(\pi_c))\}$ (refer to Sec. 3.2 of the main paper for the definition of π_c and $\mathcal{R}(\pi_c)$)
// Initialize the first instance ($s_1 = e_1 = 1$) with the same category as that of the first point label
if $t_1^{\text{act}} > t_1^{\text{bkg}}$, **then** $\pi_c^0 \leftarrow \{(1, 1, 0)\}$ **else** $\pi_c^0 \leftarrow \{(1, 1, 1)\}$
$\mathcal{S}_c \leftarrow \{(\pi_c^0, \infty)\}$
$i \leftarrow 1$; $j \leftarrow 1$

// For each step t, find the top α sequences which span from the first segment to the t-th segment while agreeing with point labels.
for $t = 2$ **to** T **do**
  // Find the upcoming points for action and background, respectively.
  if $t > t_i^{\text{act}}$, **then** $i \leftarrow \min(i + 1, M_c^{\text{act}})$; **if** $t > t_j^{\text{bkg}}$, **then** $j \leftarrow \min(j + 1, M^{\text{bkg}})$
  // Remember the category of the closest upcoming point, as it will determine the possible cases (to continue or to be terminated)
  if $t_i^{\text{act}} > t_j^{\text{bkg}}$, **then** $z^{\text{upcoming}} \leftarrow 0$ **else** $z^{\text{upcoming}} \leftarrow 1$
  // If t surpasses either of the last points for action and background, reverse the upcoming category
  if $t > \min(t_i^{\text{act}}, t_j^{\text{bkg}})$, **then** $z^{\text{upcoming}} \leftarrow 1 - z^{\text{upcoming}}$

  // Update the candidate sequence set for the timestep t
  $\mathcal{S}_c^{\text{next}} \leftarrow \varnothing$
  while $\mathcal{S}_c \neq \varnothing$ **do**
    pop $\left(\pi_c = \{(s_n, e_n, z_n)\}_{n=1}^{N}, \mathcal{R}^{\text{current}}\right)$ from $\mathcal{S}_c$
    pop the last instance (s_N, e_N, z_N) from π_c *// e_N should be equal to $t - 1$*
    // The case where the last instance continues at timestep t
    if $z_N = z^{\text{upcoming}}$ **or** $t \notin \left(\mathcal{B}_c^{\text{act}} \cup \mathcal{B}^{\text{bkg}}\right)$ **then**
      $\pi_c^{\text{new}} \leftarrow \pi_c \cup \{(s_N, e_N + 1, z_N)\}$
      $\mathcal{S}_c^{\text{next}} \leftarrow \mathcal{S}_c^{\text{next}} \cup \left\{\left(\pi_c^{\text{new}}, \mathcal{R}^{\text{current}}\right)\right\}$
    end if
    // The case where the last instance is terminated at timestep $t - 1$ and a new instance starts at timestep t
    if $z_N \neq z^{\text{upcoming}}$ **then**
      $\pi_c^{\text{last}} \leftarrow \{(s_N, e_N, z_N)\}$
      // Update the score of the candidate sequence by averaging the contrast scores again
      if $N = 1$, **then** $\mathcal{R}^{\text{new}} \leftarrow \mathcal{R}(\pi_c^{\text{last}})$ **else** $\mathcal{R}^{\text{new}} \leftarrow \left(\mathcal{R}(\pi_c^{\text{last}}) + (N - 1)\mathcal{R}^{\text{current}}\right)/N$
      // Create a new instance that starts right after the last instance, with the category of $z^{upcoming}$
      $\pi_c^{\text{new}} \leftarrow \pi_c \cup \pi_c^{\text{last}} \cup \{(e_N + 1, e_N + 1, z^{\text{upcoming}})\}$
      $\mathcal{S}_c^{\text{next}} \leftarrow \mathcal{S}_c^{\text{next}} \cup \left\{\left(\pi_c^{\text{new}}, \mathcal{R}^{\text{new}}\right)\right\}$
    end if
  end while
  $\mathcal{S}_c \leftarrow \mathcal{S}_c^{\text{next}}$
  // Pruning with the budget size α
  while $|\mathcal{S}_c| > \alpha$ **do**
    $\pi_c^{\text{min}} \leftarrow \arg\min_{\pi_c} \mathcal{R}^{\text{current}}$ for $\left(\pi_c, \mathcal{R}^{\text{current}}\right) \in \mathcal{S}_c$
    pop $\left(\pi_c^{\text{min}}, \mathcal{R}^{\text{current}}\right)$ from $\mathcal{S}_c$
  end while
end for

// Return the optimal sequence
$\pi_c^* \leftarrow \arg\max_{\pi_c} \mathcal{R}(\pi_c)$ for $\pi_c \in \mathcal{S}_c$
return π_c^*



(a) (b)

Figure 1: Correlation between scores and IoUs with ground-truths. (a) The inner score shows moderate correlation (Pearson's r = 0.38), whereas (b) the score contrast displays much stronger correlation (Pearson's r = 0.68).

Mining approach	mAP@IoU (%)							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	AVG
Global mining [8]	67.4	61.1	54.9	46.3	36.4	25.7	13.4	43.6
Ours w/o filling	70.1	64.4	57.6	49.5	39.4	29.5	15.5	46.6
Ours	70.7	65.2	58.1	49.8	40.7	30.2	16.1	47.3

Table 2: Comparison of different pseudo background mining approaches on THUMOS'14. AVG represents the average mAP at the IoU thresholds 0.1:0.1:0.7.

C. Additional Experiments

C.1. Score contrast *vs.* completeness

To analyze the correlation between score contrast and action completeness, we draw the scatter plot of score contrast *vs.* IoUs with ground-truth action instances, using the randomly sampled 2,000 temporal intervals in the THUMOS'14 training videos. For reference, we also present the scatter plot of inner action scores *vs.* IoUs with the same intervals. In the experiments, we use the *baseline* model for fair comparison. Fig. 1a demonstrates that there is a moderate correlation between inner action scores and IoUs, but there are many cases with large inner scores but low IoUs (see bottom right). On the contrary, as shown in Fig. 1b, score contrast correlates much stronger with IoUs, demonstrating its efficacy as a proxy for measuring the action completeness without any supervision.

C.2. Analysis on Pseudo Background Mining

We compare different variants of pseudo background mining on THUMOS'14. Specifically, we consider three variants: (1) "Global mining" selects the top ηM^{act} points throughout the whole video without considering their locations as in SF-Net [8], where M^{act} is the number of action instances and η is set to 5, (2) "Ours w/o filling" follows the principle described in Sec. 3.1 except the filling stage, *i.e.*, we select at least one background point for each section between two action points, and (3) "Ours" mines all points between the background points for each section if multiple points are found in the second variant. Note that we use the *baseline* model without completeness learning for clear comparison.

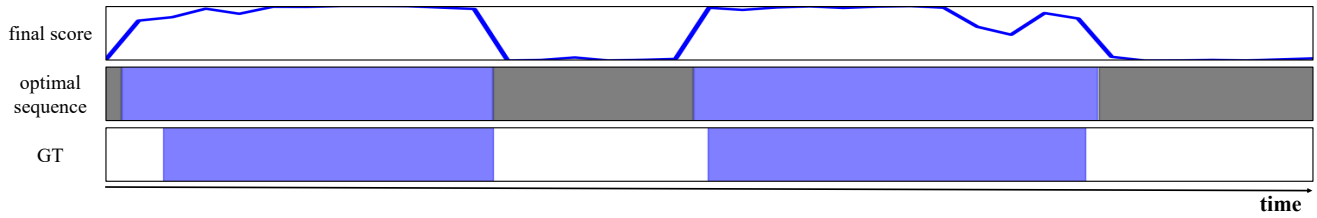
The results are demonstrated in Table 2. It can be observed that both of our methods significantly outperform the "Global mining" approach, which verifies the effectiveness of our selection principle that at least one background point should be placed for each section. Moreover, by ensuring at least one background point for each section, the search space of optimal sequence selection can be significantly reduced, although we do not include the cost analysis for this experiment. Meanwhile, we notice that filling between two background points slightly boosts the localization performance. This is presumably because hard background points with low scores can be collected in the filling step.

C.3. Optimal Sequence Visualization

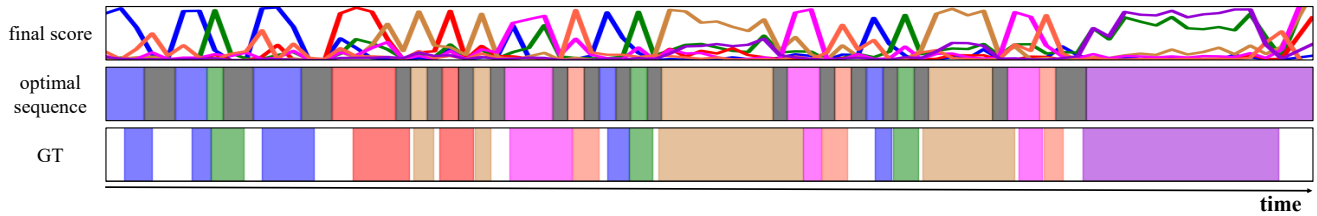
In Fig. 2, we visualize the obtained optimal sequences for the examples from the three benchmarks. In the first example from THUMOS'14 (a), the optimal sequence covers the ground-truth action instances well so that the model could learn action completeness from it. Moreover, although the examples from GTEA (b) and BEOID (c) contain a variety of action classes in a single video, our method successfully finds the optimal sequence that shows large overlaps with the ground-truth ones. Overall, it is shown from all the examples that the optimal sequences are quite accurate even though they are selected based on point-level labels without full supervision. They in turn provide completeness guidance to our model, which proves to improve localization performances at high IoU thresholds in Sec. 4.3 of the main paper.

C.4. More Qualitative Comparison

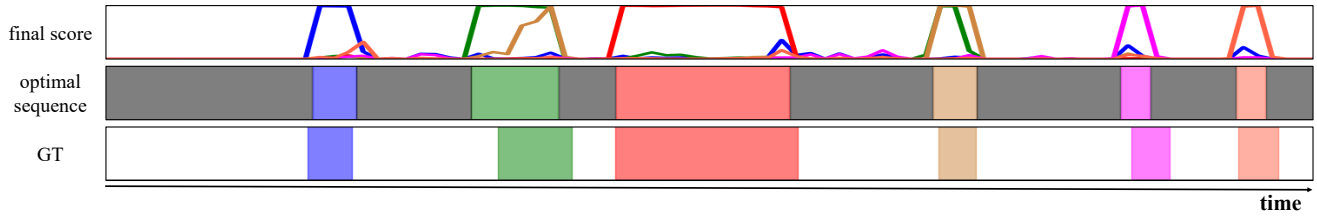
We qualitatively compare our method with SF-Net [8] on the three benchmarks. The comparison on THUMOS'14 [3] is demonstrated in Fig. 3. As shown, SF-Net produces fragmentary predictions by splitting action instances, whereas our method outputs complete ones with high IoUs even for the extremely long action instance (b). The comparison result on GTEA [7] is presented in Fig. 4. It would be noted that action localization on GTEA is challenging as the frames with different action categories are visually similar, leading to false positives. We see that SF-Net has difficulty in distinguishing action instances from background ones, resulting in inaccurate localization. On the other hand, our method successfully finds the action instances by learning completeness, showing fewer false positives. Lastly, the comparison on BEOID [2] is shown in Fig. 5. It can be clearly noticed that SF-Net fails to predict the ending times of action instances, leading to the overestimation problem. On the contrary, with the help of the completeness guidance, our method better separates actions from their surroundings and locates the action instances more precisely.



(a) An example from THUMOS'14 (video_validation_0000261)

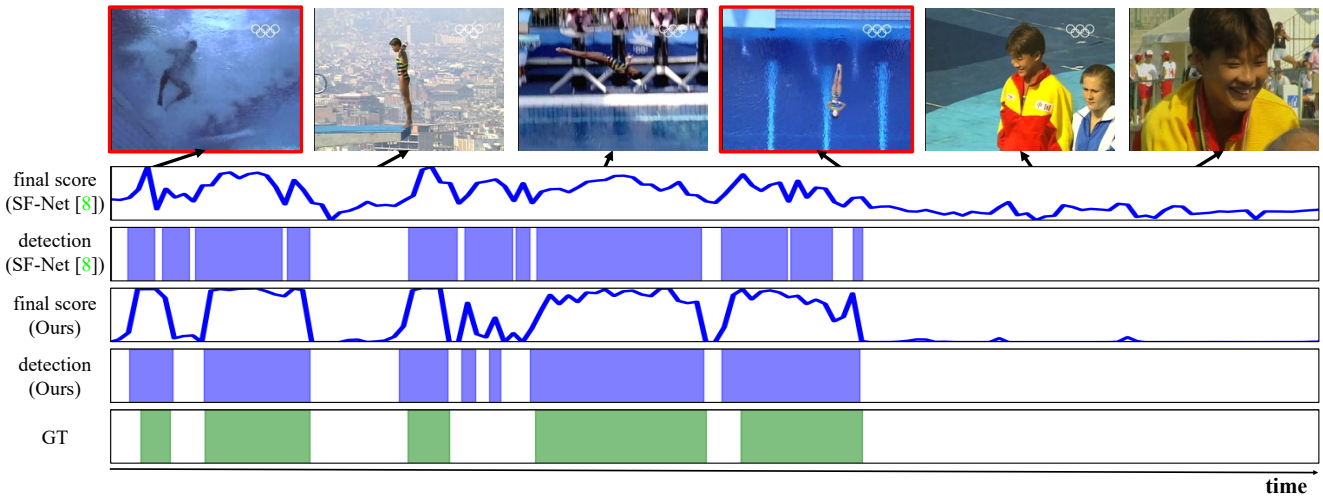


(b) An example from GTEA (S1_CofHoney_C1)

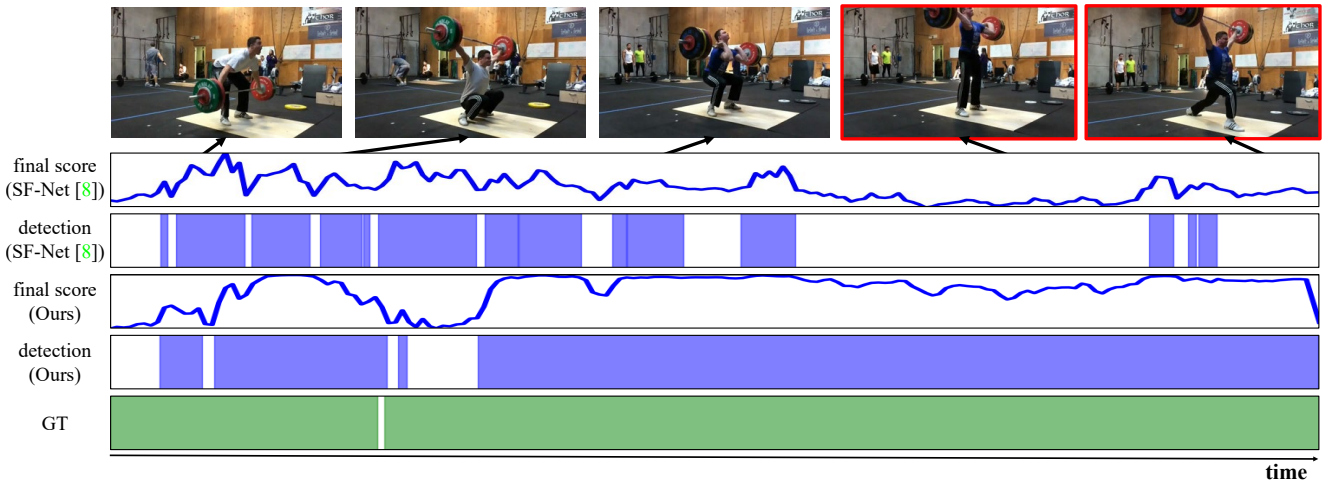


(c) An example from BEOID (02_Desk1)

Figure 2: Optimal sequence visualization on the three benchmarks. The examples are taken from (a) THUMOS'14, (b) GTEA, and (c) BEOID, respectively. Note that all of the examples belong to the training set of the corresponding benchmarks. For each video, we present the final scores and the obtained optimal sequences as well as ground-truth action intervals. The horizontal axis in each plot denotes the timesteps of the video, while the vertical axis in the first plot indicates the score values ranging from 0 to 1. For each example, different colors correspond to different action categories, while the gray color indicates the background class.

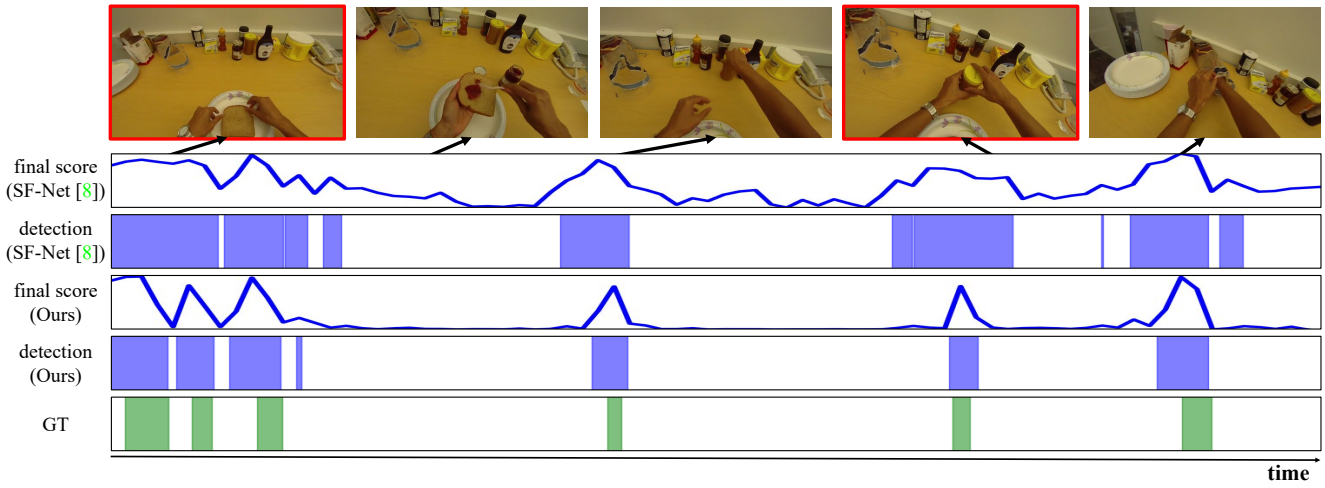


(a) An example of *Diving* action (video_test_0001309)

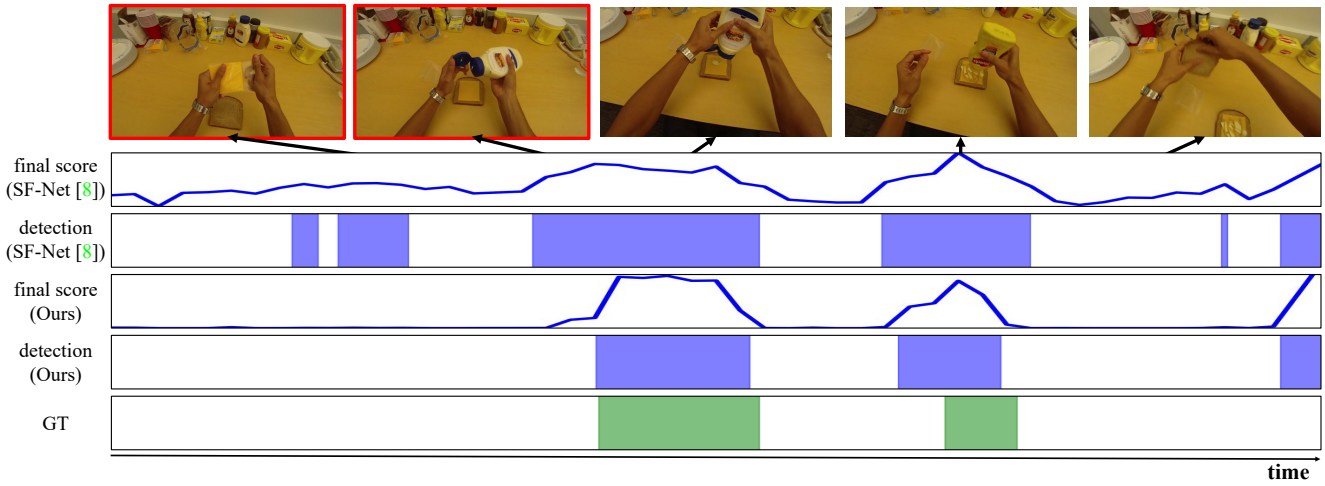


(b) An example of *CleanAndJerk* action (video_test_000058)

Figure 3: Qualitative comparison with SF-Net [8] on THUMOS'14. We provide two examples with different action classes: (a) *Diving* and (b) *CleanAndJerk*. For each video, we present the final scores and detection results from SF-Net and our model as well as ground-truth action intervals. The horizontal axes denote the timesteps of the video, while the vertical axes are the score values ranging from 0 to 1. The detection threshold is set to 0.2 for our method and set to the mean score for SF-Net following the original paper. The red boxes indicate the frames that are misclassified by SF-Net but detected by our method. All of our detection results show high IoUs (> 0.5) with the corresponding ground-truths regardless of their lengths.

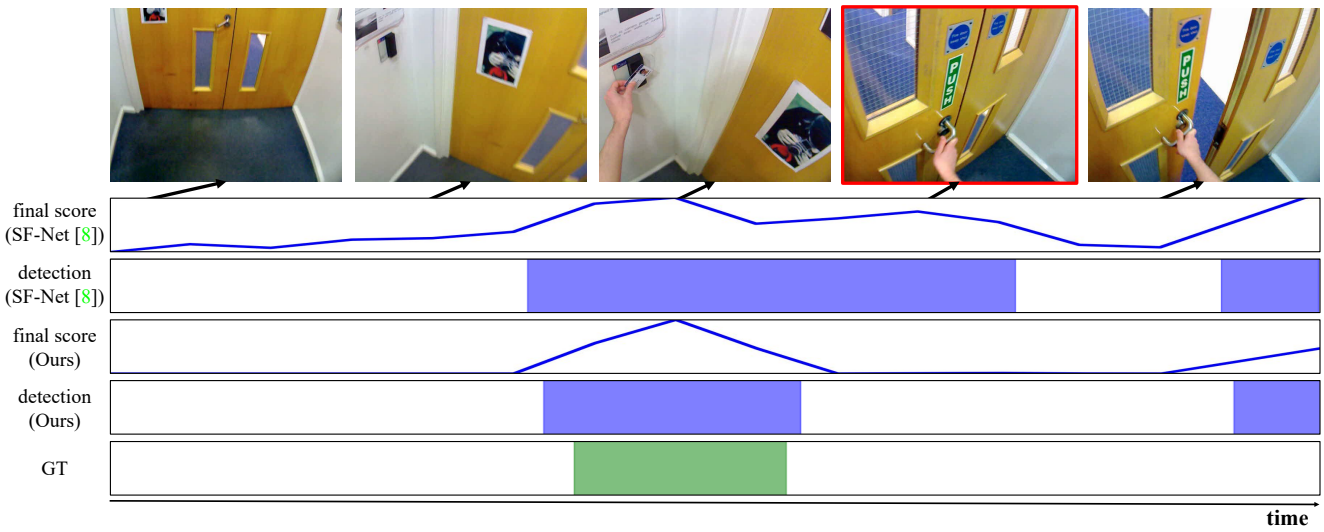


(a) An example of *Take* action (S4_Pealate_C1)

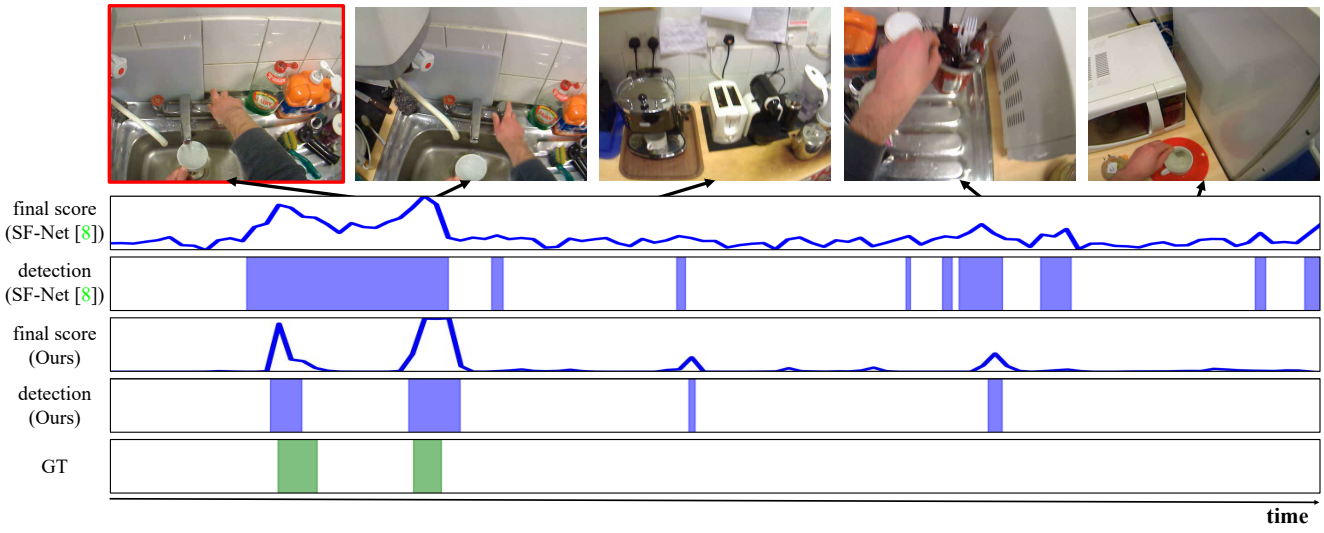


(b) An example of *Pour* action (S4_Cheese_C1)

Figure 4: Qualitative comparison with SF-Net [8] on GTEA. We provide two examples with different action classes: (a) *Take* and (b) *Pour*. For each video, we present the final scores and detection results from SF-Net and our model as well as ground-truth action intervals. The horizontal axis in each plot denotes the timesteps of the video, while the vertical axes are the score values ranging from 0 to 1. The detection threshold is set to 0.2 for our method and set to the mean score for SF-Net following the original paper. The red boxes indicate false alarms of SF-Net, but they, however, are rejected by our method. Compared to SF-Net, our method localizes action instances more precisely with fewer false positives.



(a) An example of *Scan_Card-reader* action (01_Door1)



(b) An example of *Turn_Tap* action (02_Sink2)

Figure 5: Qualitative comparison with SF-Net [8] on BEOID. We provide two examples with different action classes: (a) *Scan_Card-reader* and (b) *Turn_Tap*. For each video, we present the final scores and detection results from SF-Net and our model as well as ground-truth action intervals. The horizontal axis in each plot denotes the timesteps of the video, while the vertical axes are the score values ranging from 0 to 1. The detection threshold is set to 0.2 for our method and set to the mean score for SF-Net following the original paper. The red boxes indicate false alarms of SF-Net deteriorating the performances at high IoU thresholds. While SF-Net overestimates the action instances, our method detects the complete action instances by discriminating action instances from background ones well.

References

[1] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *ECCV*, pages 549–565, 2016. 1

[2] Dima Damen, Teesid Leelasawassuk, Osian Haines, Andrew Calway, and Walterio W Mayol-Cuevas. You-do, i-learn: Discovering task relevant objects and their modes of interaction from multi-user egocentric video. In *BMVC*, volume 2, page 3, 2014. 3

[3] 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. 3

[4] Tsung-Wei Ke, Jyh-Jing Hwang, and Stella X Yu. Universal weakly supervised segmentation by pixel-to-segment contrastive learning. In *ICLR*, 2021. 1

[5] Issam Laradji, Pau Rodriguez, Oscar Manas, Keegan Lensink, Marco Law, Lironne Kurzman, William Parker, David Vazquez, and Derek Nowrouzezahrai. A weakly supervised consistency-based learning method for covid-19 segmentation in ct images. In *WACV*, pages 2453–2462, 2021. 1

[6] Issam H Laradji, Negar Rostamzadeh, Pedro O Pinheiro, David Vazquez, and Mark Schmidt. Proposal-based instance segmentation with point supervision. In *ICIP*, pages 2126–2130, 2020. 1

[7] Peng Lei and Sinisa Todorovic. Temporal deformable residual networks for action segmentation in videos. In *CVPR*, pages 6742–6751, 2018. 3

[8] Fan Ma, Linchao Zhu, Yi Yang, Shengxin Zha, Gourab Kundu, Matt Feiszli, and Zheng Shou. Sf-net: Single-frame supervision for temporal action localization. In *ECCV*, 2020. 3, 5, 6, 7

[9] R Austin McEver and BS Manjunath. Pcams: Weakly supervised semantic segmentation using point supervision. *arXiv preprint arXiv:2007.05615*, 2020. 1

[10] Pascal Mettes and Cees GM Snoek. Pointly-supervised action localization. *International Journal of Computer Vision*, 127(3):263–281, 2019. 1

[11] Pascal Mettes, Jan C Van Gemert, and Cees GM Snoek. Spot on: Action localization from pointly-supervised proposals. In *ECCV*, pages 437–453, 2016. 1

[12] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. Extreme clicking for efficient object annotation. In *ICCV*, pages 4930–4939, 2017. 1

[13] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. Training object class detectors with click supervision. In *CVPR*, pages 6374–6383, 2017. 1

[14] Zhongzheng Ren, Zhiding Yu, Xiaodong Yang, Ming-Yu Liu, Alexander G Schwing, and Jan Kautz. Ufo^2: A unified framework towards omni-supervised object detection. In *ECCV*, pages 288–313, 2020. 1

[15] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *CVPR*, pages 850–859, 2019. 1