Efficient and Effective Weakly-Supervised Action Segmentation via Action-Transition-Aware Boundary Alignment

Supplementary Material

A1. Details of Temporal Network

As the recent works [3, 12] have pointed out, the vanilla self-attention mechanism is not suitable for action segmentation task, since it is hard to be learned to focus on meaningful temporal positions over a very long video. Hence, we replace the vanilla self-attention with a pyramid hierarchical local attention as in [12] to achieve a local-to-global learning pattern which is similar to CNNs. Specifically, each frame only performs self-attention with the frames in a local window centered at itself, and the window size increases in the deeper layers. The radius of the window is set to 2^{l-1} in the l-th (beginning from 1) encoder layer.

A2. Construction of Pairwise Similarity Matrix

In the class-agnostic boundary scoring step of our Action-Transition-Aware Boundary Alignment (ATBA), a pairwise similarity matrix $\mathbf{\Gamma}^{(t)} \in \mathbb{R}^{w^{\mathrm{b}} \times w^{\mathrm{b}}}$ is calculated within the local window with size w^{b} centered at t, from the model output $\boldsymbol{P}$:

$$\begin{aligned} \mathbf{\Gamma}^{(t)}_{i,j} &= 1 - 2\,\mathrm{JS}(\boldsymbol{p}_{\mathrm{ind}^{\mathrm{b}}(t,i)}, \boldsymbol{p}_{\mathrm{ind}^{\mathrm{b}}(t,j)}),\ 1 \le i, j \le w^{\mathrm{b}}, \\ \mathrm{ind}^{\mathrm{b}}(t,i) &= t - \lfloor \frac{w^{\mathrm{b}}}{2} \rfloor + i - 1, \end{aligned} \tag{1}$$

where $\mathrm{ind}^{\mathrm{b}}(t,i)$ is the index transform from the index i of the local window centered at t to the global timestamp index, $\mathrm{JS}(\cdot,\cdot)$ is the Jensen–Shannon divergence and $\boldsymbol{p}_{\mathrm{ind}^{\mathrm{b}}(t,i)}$ is the class probability distribution of the frame at timestamp $\mathrm{ind}^{\mathrm{b}}(t,i)$. $\mathbf{\Gamma}^{(t)}_{i,j}$ represents the output similarity between $\mathrm{ind}^{\mathrm{b}}(t,i)$-th and $\mathrm{ind}^{\mathrm{b}}(t,j)$-th frames, of which the range is $[-1,1]$.

A3. Details of Action Transition Alignment

To help better understand the action transition alignment algorithm in our ATBA, we provide the pseudo code in Alg. A1. The algorithm consists of three stages, *i.e.*, initialization (Line 1-20), calculation by dynamic programming (Line 21-30), and backtracking (Line 31-45). The middle stage is stated in the main paper.

- Initialization. The first row and the first two columns of the cumulative cost matrix $\boldsymbol{D}$ can not be calculated via the recursive equation, and need to be directly initialized before the computation. The rules for the initialization are following:

- For the first column, the path through $(k,1)$ means the k-th candidate is matched with the *first* empty symbol, *i.e.*, the first k candidates are *all* dropped. However, there are only $K-M+1$ candidates can be dropped, so a valid path cannot pass through the last $M-1$ positions of the first column, so their values are set to ∞ (Line 2-9).
- The situation in the second column is similar to the first column, where a path through $(k,2)$ means that the k-th candidate is matched with the first transition. To ensure that the remaining $M-2$ transitions can be matched, at least the last $M-2$ candidates cannot be matched with the first transition. Hence the values of the last $M-2$ entries of the second column are set to ∞ (Line 10-17).
- For the first row, as mentioned in the main paper, only $(1,1)$ and $(1,2)$ are valid, so the values of other entries in the first row are set to ∞ (Line 18-20).
- The remaining valid positions can be initialized with the values of corresponding entries in $\boldsymbol{\Delta}$, as these positions are all in the first two columns, and so relevant to at most one transition matching without accumulating multiple costs.

- Backtracking. After filling the matrix $\boldsymbol{D}$, we find out the optimal boundary set $\mathcal{B}$ (*i.e.*, an alignment path) from it using backtracking. Clearly, any valid path has exactly one point in each row, meaning that each candidate is matched with one symbol (transition or ϕ). As mentioned in the main paper, the end position of the optimal path is one of $(K, 2(M-1))$ and $(K, 2(M-1)+1)$ depending on whose $\boldsymbol{D}$ value is minimal (Line 35 in the first loop, *i.e.*, $i = K$). The backtracking starts from this end position, and runs from bottom to top until the first row. The Line 34-39 calculate the column position in current row i based on the determined position in the next row (*i.e.*, next point in the path). Similar to the forward process, if the column position j of the next row $i+1$ is odd, *i.e.*, the candidate b_{i+1} is dropped, then in current row i, candidate b_i can be either dropped ($new\_j = j$) or matched with the previous transition ($new\_j = j-1$) depending on the cumulative cost (Line 34-36). The meaning of Line 37-39 (j is even) is similar. If the point of current row is matched with a transition, we add it into $\mathcal{B}$ (Line 40-42).

A4. Additional Training Details

During training, the batch size is 32 and the AdamW [7] optimizer is adopted. We train the model for 400/300/300 epochs for Breakfast [4], Hollywood [1] and CrossTask [13], respectively, of which the first 40 epochs

Algorithm A1: Action Transition Alignment

Input: Candidate boundary set $\widetilde{\mathcal{B}} = \{b_k\}_{k=1}^{K}$; Cost matrix $\mathbf{\Delta} \in \mathbb{R}^{K\times(2(M-1)+1)}$

/* Initialize the cumulative cost matrix $\boldsymbol{D}$ */

$\boldsymbol{D} \leftarrow$ RandomMatrix $\in \mathbb{R}^{K\times(2(M-1)+1)}$;

/* Initialize the 1st column */

for $i \leftarrow 1$ **to** K **do**
  if $i \leq K-(M-1)$ **then**
    $\boldsymbol{D}_{i,1} \leftarrow \mathbf{\Delta}_{i,1}$;
  end
  else
    $\boldsymbol{D}_{i,1} \leftarrow \infty$;
  end
end

/* Initialize the 2nd column */

for $i \leftarrow 1$ **to** K **do**
  if $i \leq K-(M-1)+1$ **then**
    $\boldsymbol{D}_{i,2} \leftarrow \mathbf{\Delta}_{i,2}$;
  end
  else
    $\boldsymbol{D}_{i,2} \leftarrow \infty$;
  end
end

/* Initialize the 1st row */

for $j \leftarrow 3$ **to** $2(M-1)+1$ **do**
  $\boldsymbol{D}_{1,j} \leftarrow \infty$;
end

/* Dynamic programming */

for $i \leftarrow 2$ **to** K **do**
  for $j \leftarrow 3$ **to** $2(M-1)+1$ **do**
    if j *is odd* **then**
      $\boldsymbol{D}_{i,j} \leftarrow \mathbf{\Delta}_{i,j} + \min(\boldsymbol{D}_{i-1,j}, \boldsymbol{D}_{i-1,j-1})$;
    end
    else
      $\boldsymbol{D}_{i,j} \leftarrow \mathbf{\Delta}_{i,j} + \min(\boldsymbol{D}_{i-1,j-1}, \boldsymbol{D}_{i-1,j-2})$;
    end
  end
end

/* Backtracking */

Initialize the optimal boundary set $\mathcal{B} = \phi$;

$j \leftarrow 2(M-1)+1$;

for $i \leftarrow K$ **to** 1 **do**
  if j *is odd* **then**
    $new\_j \leftarrow \arg\min_{\{j,j-1\}}(\boldsymbol{D}_{i,j}, \boldsymbol{D}_{i,j-1})$;
  end
  else
    $new\_j \leftarrow \arg\min_{\{j-1,j-2\}}(\boldsymbol{D}_{i,j-1}, \boldsymbol{D}_{i,j-2})$;
  end
  if new_j *is even* **then**
    Add b_i into $\mathcal{B}$;
  end
  $j \leftarrow new\_j$;
end

Reverse $\mathcal{B}$;

Output: Optimal boundary set $\mathcal{B}$

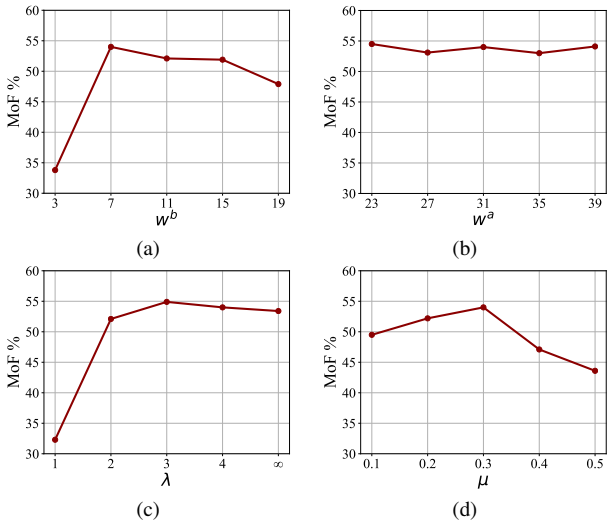


Figure A1. The Effect of (a) the size of class-agnostic boundary pattern template w^{b}, (b) the size of action-transition pattern template w^{a}, (c) λ, which controls the upper bound of the number of candidate boundaries, and (d) μ, which controls the size of non maximum suppression (NMS) area. The case of $\lambda = \infty$ means that the candidate selection process terminates only when all remaining timestamps are invalid. Experiments are all conducted on the Breakfast.

are the first stage. The initial learning rate is set to 5e-4. The cosine annealing strategy [6] is used only for the second stage to lower the learning rate to 1/100 of the initial value finally, while the warmup strategy is used for the first 10 epochs of both two stages, beginning from 1/100 and 1/10 of the initial learning rate, respectively.

A5. Detailed Sources of Results

In Table 1 of the main paper, some results are not reported by the original paper, and the detailed sources are as follows:

- Breakfast. The MoF results with standard deviation of ISBA [2], NN-Viterbi [9] and CDFL [5] are from [10]. The MoF-Bg, IoU and IoD results of MuCon [10] are from [11].

- Holloywood Extended. The IoU result of MuCon [10] are from [11].

- CrossTask. All the results of NN-Viterbi [9] and CDFL [5] are from [8].

A6. Analysis of Hyper-parameters

- Effect of w^{b}. Fig. A1(a) shows the effect of the size of class-agnostic boundary pattern template w^{b}. The model performs bad with too small w^{b}, possibly because it is more susceptible to noise interference. On the other hand, the large w^{b} can also lead to performance decrease due to the poor ability of capturing local changes.

- Effect of w^{a}. The effect of the size of action transition pattern template w^{a} is also shown in Fig. A1(b). Our method is insensitive to it over a wide range (at least 23-29). Note that these feasible values are much higher than that of w^{b}, since the action transition scoring aims to capture two adjacent segments which both lasts for a period of time.

- Effect of λ. We investigate the effect of λ in Fig. A1(c), which controls the upper bound of the number of candidate boundaries. Note that $\lambda = 1$ is equivalent to not applying action transition alignment (*i.e.*, Exp.1 of the ablation study on ATBA in the main paper), so the performance is poor. When $\lambda > 1$, the performance can be maintained at a high level and keep stable as the number of candidates increases, since the additional candidates may be *unambiguous* non-boundary points and have little effect.

- Effect of μ. Fig. A1(d) shows the effect of μ, which controls the size of non maximum suppression (NMS) area. Our ATBA prefers relatively small NMS area, since the large NMS area will lead to missing the transitions involving short segments.

A7. More Qualitative Results

To help more intuitively understand the advantage of our method, we provide more qualitative results on three datasets: Breakfast [4], Hollywood [1] and CrossTask [13] in Fig. A2. Our method is significantly more accurate in locating actions than MuCon [10] and TASL [8]. Note that in Fig. A2(e), there is indeed a shot of *espresso* in the video (the 2nd picture, but without a *pouring* action) after action "*Pour Milk*" (the 1st picture), so the activation on action "*Pour Espresso*" in our result is not exactly a hallucination compared to the result of TASL [8].

References

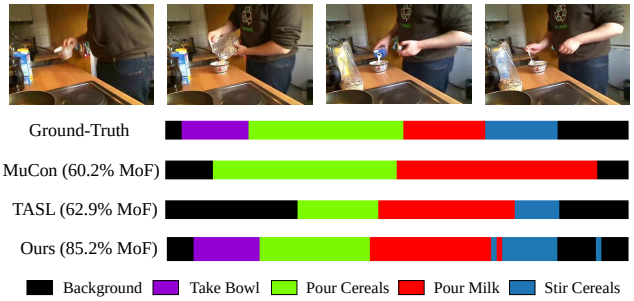
[1] Piotr Bojanowski, Rémi Lajugie, Francis Bach, Ivan Laptev, Jean Ponce, Cordelia Schmid, and Josef Sivic. Weakly supervised action labeling in videos under ordering constraints. In *Proceedings of the European Conference on Computer Vision*, pages 628–643. Springer, 2014. 1, 3

[2] Li Ding and Chenliang Xu. Weakly-supervised action segmentation with iterative soft boundary assignment. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6508–6516, 2018. 2

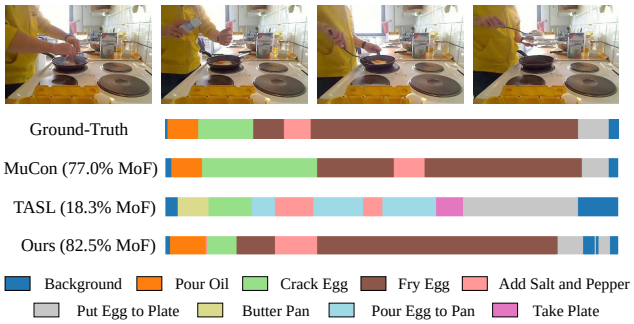
[3] Dazhao Du, Bing Su, Yu Li, Zhongang Qi, Lingyu Si, and Ying Shan. Efficient u-transformer with boundary-aware loss for action segmentation. *arXiv preprint arXiv:2205.13425*, 2022. 1

[4] Hilde Kuehne, Ali Arslan, and Thomas Serre. The language of actions: Recovering the syntax and semantics of goal-directed human activities. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 780–787, 2014. 1, 3

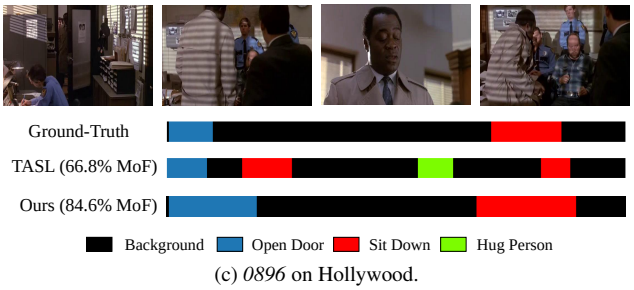
[5] Jun Li, Peng Lei, and Sinisa Todorovic. Weakly supervised energy-based learning for action segmentation. In *Proceed-*



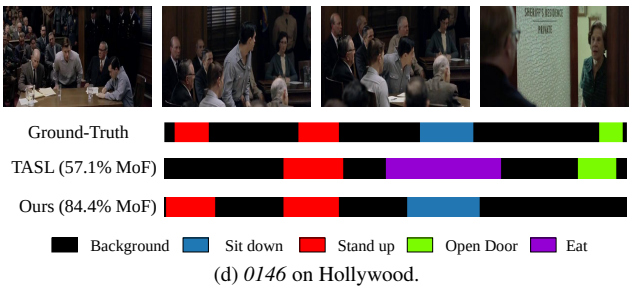
(a) *P03-webcam02-P03-cereals* on Breakfast.



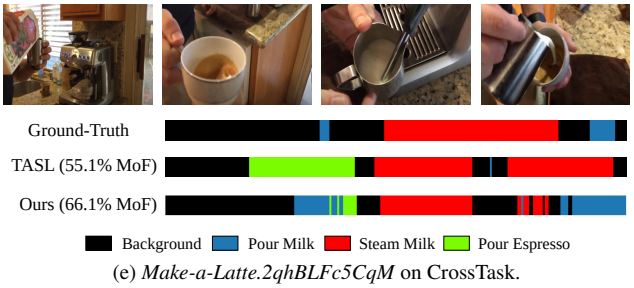
(b) *P10-webcam01-P10-friedegg* on Breakfast.



(c) *0896* on Hollywood.



(d) *0146* on Hollywood.



(e) *Make-a-Latte.2qhBLFc5CqM* on CrossTask.

Figure A2. More qualitative results. The names of example test videos are shown below each visualization. Best viewed in color.

ings of the IEEE/CVF International Conference on Computer Vision, pages 6243–6251, 2019. 2

[6] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 2

[7] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 1

[8] Zijia Lu and Ehsan Elhamifar. Weakly-supervised action segmentation and alignment via transcript-aware union-of-subspaces learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8085–8095, 2021. 2, 3

[9] Alexander Richard, Hilde Kuehne, Ahsan Iqbal, and Juergen Gall. Neuralnetwork-viterbi: A framework for weakly supervised video learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7386–7395, 2018. 2

[10] Yaser Souri, Mohsen Fayyaz, Luca Minciullo, Gianpiero Francesca, and Juergen Gall. Fast weakly supervised action segmentation using mutual consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):6196–6208, 2021. 2, 3

[11] Yaser Souri, Yazan Abu Farha, Fabien Despinoy, Gianpiero Francesca, and Juergen Gall. Fifa: Fast inference approximation for action segmentation. In *Proceedings of the DAGM German Conference on Pattern Recognition*, pages 282–296. Springer, 2022. 2

[12] Fangqiu Yi, Hongyu Wen, and Tingting Jiang. Asformer: Transformer for action segmentation. *arXiv preprint arXiv:2110.08568*, 2021. 1

[13] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3537–3545, 2019. 1, 3