Supplementary Materials: Two-Stage Active Learning for Efficient Temporal Action Segmentation

Yuhao Su and Ehsan Elhamifar

Khoury College of Computer Sciences, Northeastern University, Boston, USA
{su.yuh,e.elhamifar}@northeastern.edu

1 Additional Experiments

1.1 Active Learning Hyperparameters

We analyze the performance as we change the active learning hyperparameters while keeping the total annotation budget at 0.16% frames. More specifically, we change the total active cycles c, percentage of selected videos and clips in each selected video m and ρ so that $(c \times m \times \rho)/32 = 0.16\%$ frames. Tab. 1 shows the results on 50Salads. Comparing the first and the last two rows of the table suggests that selecting more clips in selected videos (while reducing either the number of active learning cycles or the number of selected videos) does not necessarily contribute to performance improvement. This is because more active learning cycles allow us to improve the model and subsequently better select videos and clips for annotation and using more videos allows seeing more diverse action sequences for better training. We conclude that under the same budget, labeling a small percentage of videos and clips along with relatively more active cycles often yields the best performance.

1.2 Comparison with Timestamp Supervision Methods

We compare our method against D-TSTAS (timestamp supervision method) and EM-gen (SkipTag supervision method) on 50Salads and GTEA in Tab. 2. We achieve comparable results to EM-gen using less budget. When comparing with D-TSTAS, we do not expect our method to consistently outperform it, given their much stronger supervision.

1.3 Annotation Method Effectiveness

We compare different annotation methods and show results in Tab. 3. Our standard annotation method involves labeling the middle frame of each selected clip and using it as clip label. Majority voting, as an upper bound, requires labeling every frame in each selected clip and takes the majority as clip label. We follow our standard AL setting ($\rho = 25\%$, $m = 5\%$) and run our method for 4 AL iterations, resulting in 0.16% and 5% frames labeled for middle frame annotation

$c \times m \times \rho$	acc	edit	f1@{10,25,50}		
$4 \times 5\% \times 25\%$	**57.8**	45.0	**55.1**	**49.1**	**32.9**
$2 \times 5\% \times 50\%$	40.0	32.2	35.0	29.5	15.5
$2 \times 10\% \times 25\%$	53.7	**45.1**	54.1	46.7	30.3

Table 1: Effect of AL hyperparameters on 50Salads, all cases use our AL method.

dataset	method	budget	acc	edit	f1@{25,50}	
GTEA	*D-TSTAS [4]*	*2.90%*	*75.7*	*88.5*	*90.1*	*76.2*
GTEA	EM-gen [5]	2.90%	**69.8**	73.5	76.7	57.9
GTEA	Ours	**2.60%**	67.7	**75.6**	**77.3**	**58.7**
50Salads	*D-TSTAS [4]*	*0.18%*	*80*	*77.6*	*82.1*	*71.5*
50Salads	EM-gen [5]	0.18%	74.4	64.3	68.1	**54.9**
50Salads	Ours	**0.15%**	**74.6**	**68.8**	**73.7**	54.0

Table 2: Comparison with timestamp and SkipTag supervision methods on 50Salads and GTEA.

and majority voting, respectively. Tab. 3 shows similar performance are obtained from both annotation methods, suggesting annotating the middle frame is as effective as majority voting, while significantly reducing the annotation budget.

2 Complexity

We analyze the complexity of our propose Video-Aligned summarization (VAS) method. VAS aims to find a summary $\boldsymbol{S} = [\boldsymbol{s}_1, \ldots, \boldsymbol{s}_K]$ from video sequence $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n]$. Finding such summary is challenging due to the combinatorial search over all possible 2^n subsets. By leveraging greedy algorithm, VAS reduces the complexity from 2^n to $\mathcal{O}(n^2K^2)$.

3 Implementation Details

For model architecture, we use one encoder and three decoders, with each encoder/decoder block having a relative small number of layers: 3, 4, 6, 6 for GTEA [2], 50Salads [6], Breakfast [3] and CrossTask [8], respectively. Prototypes are initialized in the embedding space, Following [7], their dimensions are 64. All experiments are performed using PyTorch on one Nvidia RTX 6000 GPU.

4 Clip-Based Learning and Efficiency Analysis

Our method utilizes clip features and is trained with a lighter model. Clip features are used for the following reasons. First, single frame features are insufficient for representing entire actions for selection, as actions typically span at

50Salads					
annotation method	acc	edit	f1@{10,25,50}		
Majority Voting	57.1	**48.2**	**56.3**	**50.0**	**34.2**
Middle Frame	**57.8**	45.0	55.1	49.1	32.9
Breakfast					
annotation method	acc	edit	f1@{10,25,50}		
Majority Voting	**64.1**	58.4	62.8	57.6	42.3
Middle Frame	63.5	**58.6**	**62.8**	**58.1**	**43.5**

Table 3: Performance comparison with different annotation method on 50Salads and Breakfast. All cases use our propose active learning method and run 4 AL iterations.

model	#params(M)	FLOPs(g)	G-Mem.(g)	Infer Speed(ms)
MS-TCN [7]	0.799	4.8	∼1.7G	N/A
ASFormer [7]	1.134	6.8	∼3.5G	N/A
Ours	**0.538**	**0.13**	**∼1.2G**	**37**

Table 4: Efficiency comparison with the original ASFormer and MS-TCN on 50Salads.

least a few seconds. Second, employing clip features enables the use of a lighter model, significantly reducing training time and improving efficiency.

We compare the efficiency of our model with the original ASFormer [7] and TCN-based model MS-TCN [1] on 50Salads in Tab. 4. Compared with the original ASFormer, our model decreases the number of parameters and GPU-memory by 0.569 M and 2.3G, respectively.

We train our model using frame features under full supervision on 50Salads and GTEA. Tab. 5 shows comparison with the original ASFormer [7]. Results demonstrate that our method performs better than the original ASFormer, likely due to our proposed action prototypes and regularized contrastive loss.

dataset	method	acc	edit	f1@{10,25,50}		
50Salads	ASFormer [7]	85.6	79.6	85.1	83.4	76.0
50Salads	ours	**87.8**	**80.9**	**88.9**	**88.3**	**82.4**
GTEA	ASFormer [7]	79.7	84.6	90.1	88.8	79.2
GTEA	ours	**82.2**	**88.8**	**91.4**	**89.2**	**82.7**

Table 5: Frame feature training comparison with the original ASFormer

References

1. Farha, Y.A., Gall, J.: Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3575–3584 (2019)
2. Fathi, A., Ren, X., Rehg, J.M.: Learning to recognize objects in egocentric activities. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)
3. Kuehne, H., Arslan, A., Serre, T.: The language of actions: Recovering the syntax and semantics of goal-directed human. IEEE Conference on Computer Vision and Pattern Recognition (2014)
4. Liu, K., Li, Y., Liu, S., Tan, C., Shao, Z.: Reducing the label bias for timestamp supervised temporal action segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6503–6513 (June 2023)
5. Rahaman, R., Singhania, D., Thiery, A., Yao, A.: A generalized and robust framework for timestamp supervision in temporal action segmentation. In: Computer Vision–ECCV 2022: 17th European Conference (2022)
6. Stein, S., McKenna, S.J.: Combining embedded accelerometers with computer vision for recognizing food preparation activities. In: Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (2013)
7. Yi, F., Wen, H., Jiang, T.: Asformer: Transformer for action segmentation. In: The British Machine Vision Conference (BMVC) (2021)
8. Zhukov, D., Alayrac, J.B., Cinbis, R.G., Fouhey, D., Laptev, I., Sivic, J.: Cross-task weakly supervised learning from instructional videos. IEEE Conference on Computer Vision and Pattern Recognition (2019)