Supplementary: A Generalized & Robust Framework For TimeStamp Supervision in Temporal Action Segmentation

Rahul Rahaman[1], Dipika Singhania[2], Alexandre Thiery[3], and Angela Yao[4]

rahul.rahaman@u.nus.edu, dipika16@comp.nus.edu.sg, a.h.thiery@nus.edu.sg, ayao@comp.nus.edu.sg
National University of Singapore

In Sec. S1, we show the ablation results including the impact of prior, impact of auxiliary losses, impact of timestamp annotation position, impact of segmentation architecture, and number of epochs required for each maximization step. In Sec. S2, we visualize the performance of our EM-TSS. Sec. S3 details the posterior and prior for EM-Gen. In Sec. S4, we provide some additional details of our user study.

S1 Ablation Studies

S1.1 Impact of Prior

In Sec. 4.4 of the main text, we described the use of a length prior for Timestamp supervision. We applied a Poisson prior distribution to the action segment length with parameter μ_c, which is the average action length for action c. We compared **(1)** a non-informative prior, i.e., equal μ_c for all actions c, and with **(2)** μ_c roughly estimated from a randomly sampled video. As shown in Table T1, the performance of EM-TSS is fairly robust and varies marginally for the two different priors. This is unsurprising as the likelihood contributes significantly more to the posterior than the prior, due to the large number of frames involved in the likelihood.

Table T1: **Effect of length prior on EM-TSS**. Results are averaged over three sets of sampled timestamps. The change in performance due to different priors is marginal.

Prior	50Salads					Breakfast				
	$F1@\{10, 25, 50\}$			Edit	MoF	$F1@\{10, 25, 50\}$			Edit	MoF
non-informative	77.9	75.0	63.2	70.5	77.0	66.7	62.9	50.1	66.9	67.1
from one video	78.4	75.4	63.7	70.9	77.3	66.8	63.0	49.9	67.0	67.1

S1.2 Impact of Auxiliary Losses

Table T2 shows the effect of the additional loss terms. In the first row, we show the performance of the base loss '$\mathcal{L}_{\text{CE}}$' from the TSS work [3] on both the 50Salads and Breakfast datasets. This is a cross-entropy loss derived from hard assignments of the segment boundary to the location that minimizes their handcrafted energy function. The second row shows our base E-M loss. It is evident that our base loss yields significantly better results. Additionally, we can see that adding the auxiliary losses of [3] provides some benefit for the segment metrics of Edit Distance and F1-score, but has little impact on MoF. This is unlike [3], where the auxiliary losses also improved MoF. We believe that our frame-wise weights $\omega_{j,c}^{(m)}$ naturally embed some components of transition and confidence, so these auxiliary losses are less important for improving MoF.

Table T2: **Effect of losses on EM-TSS (same set of timestamp as [3])**. Our base E-M loss based on the negative $\mathcal{Q}$-function significantly outperforms the base cross-entropy loss of TSS [3], which uses hard boundaries from minimizing their proposed energy function. Gradually adding the different auxiliary losses to our base loss improves Edit and F1 scores, while MoF remains largely unchanged.

Losses	50Salads					Breakfast				
	$F1@\{10,25,50\}$			Edit	MoF	$F1@\{10,25,50\}$			Edit	MoF
$\mathcal{L}_{\text{CE}}$ (Base loss of [3])	65.7	62.6	50.7	57.7	72.8	60.3	52.8	36.7	64.2	60.2
$\mathcal{L}_{\text{EM}}$ (*Our* base loss)	71.8	69.0	58.7	62.8	77.4	64.9	61.2	48.1	64.9	67.8
$\mathcal{L}_{\text{EM}}+\mathcal{L}_{\text{TR}}$	75.1	71.7	60.1	66.2	76.9	65.4	61.6	48.2	65.3	67.0
$\mathcal{L}_{\text{EM}}+\mathcal{L}_{\text{TR}}+\mathcal{L}_{\text{Conf}}$	79.9	75.9	64.7	71.6	77.7	67.5	63.7	49.8	67.2	67.0

Table T3: **Effect of timestamp locations on EM-TSS.** As expected, the start frame has the worst performance because of poor initialization. The middle and random frame have similar performance.

Location of timestamp	**Method**	50Salads					Breakfast				
		$F1\{10,25,50\}$			Edit	MoF	$F1\{10,25,50\}$			Edit	MoF
Start	Naive	21.0	12.5	3.2	17.5	31.9	17.1	10.4	3.1	21.3	30.6
	EM-TSS	62.9	50.5	25.0	63.9	52.0	57.3	46.9	25.0	61.7	48.5
Centre	Naive	51.0	47.7	35.0	40.9	69.7	36.8	32.3	21.9	37.6	53.4
	EM-TSS	78.4	76.0	63.5	71.1	77.1	57.3	46.9	25.0	61.7	48.5
Random	Naive	44.7	39.4	29.3	34.2	69.9	31.9	27.6	19.5	35.4	58.0
	EM-TSS	78.4	75.4	63.7	70.9	77.3	66.8	63.0	49.9	67.0	67.1

S1.3 Impact of Timestamp Positions

In line with the previous work [3], our EM-TSS results in the main text were obtained from random timestamp annotations within the action segments. We refer to these randomly positioned timestamp annotations as '*Random*'. Alternatively, we can choose the timestamps to be some specific frame of the action segments, e.g., start frame ('*Start*') and centre frame ('*Centre*').

In Table T3, we show the variation in the results with respect to the position of the annotated timestamps. As discussed in Section 5.4 and Table 8 of the main text, the boundary annotations are highly ambiguous. Hence, the '*Naive*' performance of the start frame is extremely low. When we applied our EM-TSS, we observed a significant performance increase over the baseline. However, due to the poor performance of the initialization (i.e., naive baseline), the final performance of *'Start'* using EM-TSS is relatively low compared to other frame locations. Choosing the centre frame yields a similar result to the randomly chosen timestamp. This is in line with our expectation as the middle frame is well within the action segment, hence free from ambiguity.

Table T4: **Effect of segmentation architectures on EM-TSS.** We show that our algorithm performs equally well with different kinds of architectures. '*Backbone from [3]*' refers to the modified MSTCN model used as default backbone in the TSS work [3] as well as our work.

Backbone	**Method**	50Salads				Breakfast				GTEA			
		$F1\{25,50\}$		Edit	MoF	$F1\{25,50\}$		Edit	MoF	$F1\{25,50\}$		Edit	MoF
MSTCN [1]	Naive	38.9	28.2	34.4	67.3	39.5	30.3	39.3	55.4	54.5	38.0	49.9	55.4
	EM-TSS	68.9	57.0	64.4	75.0	61.8	49.1	64.8	65.0	80.7	65.0	80.3	68.3
MSTCN++	Naive	36.8	25.5	32.2	64.2	31.0	21.9	37.4	58.2	56.9	38.8	53.3	55.8
from [2]	*EM-TSS*	71.4	58.8	67.5	74.3	63.4	49.9	64.1	66.7	82.1	65.7	79.2	70.3
Backbone	Naive	43.3	34.0	37.2	69.6	29.1	20.1	37.4	56.8	55.3	39.6	51.1	56.5
from [3]	*EM-TSS*	75.9	64.7	71.6	77.9	63.7	49.8	67.2	67.0	82.7	66.5	82.3	70.5

S1.4 Impact of Architectures

We adopted the modified MSTCN architecture used by [3] in our work. However, results in Table T4 show that our EM-TSS works equally well with other architectures such as MSTCN [1] and MSTCN++ [2].

S1.5 Impact of N^{max}

We have three hyper-parameters regarding epochs, which are N^{init} (initialization epochs), N^{max} (epochs per maximization step) and M (number of E-M iterations). Among these, N^{init} and M can be set based on convergence. In Table T5, we show an ablation of the more non-intuitive hyper-parameter N^{max}

on 50Salads dataset (for one set of timestamp annotations as in [3]) to study its impact on the performance of EM-TSS. The table indicates that the performance is fairly stable for different values of N^{max}. We chose $N^{max} = 5$ for all datasets to balance runtime efficiency and stable performance.

Table T5: **Effect of epoch-per-maximization step N^{max}.** The performance of EM-TSS on 50Salads dataset is fairly stable with respect to different values of N^{max}, for the specific set of timestamp as chosen by [3].

	50Salads				
N^{max}	$F1@\{10, 25, 50\}$			Edit	MoF
3	77.6	73.5	60.2	69.8	74.9
5	79.9	75.9	64.7	71.6	77.9
7	77.4	74.8	64.4	70.0	77.1

S2 Qualitative Analysis

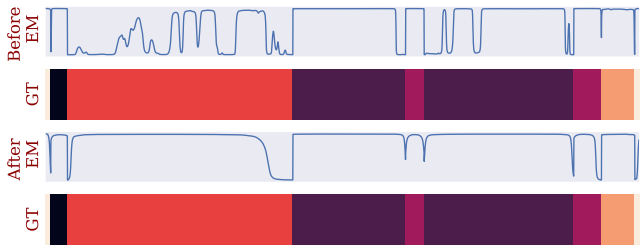

Fig. F1: **Effect of EM-TSS on ground truth probability in a training video.** In the line charts, we plot the probability assigned to the correct classes. '*Before EM*' denotes the probability obtained after 'Naive' initialization. '*After EM*' denotes the probability after convergence of EM algorithm. It is visibly clear that the probabilities are much smoother after applying E-M.

In Fig. F1, we focus on a specific training video and plot the class probabilities assigned to the correct classes. For example, for the entire red segment, we plot the probability assigned to that red action class in the line chart. The *'GT'* plots the ground truth labels. The other two line plots show the probabilities before (obtained from '*Naive*' initialization) and after the E-M algorithm (at the end of EM-TSS). It is visibly clear that the assigned probabilities improve significantly after the E-M algorithm. The naive initialization gives a highly irregular probability to the correct class. After applying E-M, the assigned probabilities are

much smoother and only become uncertain near the boundaries. Fig. F2 shows the inference result on a sample test video. We show ground truth ('*GT*'), predictions from the '*Naive*' baseline ('*Before EM*'), and predictions obtained after training with the E-M algorithm ('*After EM*') in the figure. The final model prediction is considerably close to the ground truth compared to the baseline. The noticeable changes in the prediction are that there are no fragmented action segments in the prediction, and the action segments are better aligned with the ground truth segments.

Fig. F2: **Inference performance on test video.** '*GT*' denotes the ground truth. '*Before EM*' shows the prediction after the initialization. '*After EM*' shows the prediction with the final model obtained after the convergence of the EM algorithm.

S3 EM-Gen Results and Generalization to SkipTag

S3.1 Performance of EM-Gen Under Missing Segment

In Fig. 4 of the main text, we provided a comparison between EM-Gen and the TSS methods (EM-TSS and [3]). Here in Table T6, we provide the detailed numerical results of EM-Gen under a varying degree of missing segments. Our EM-Gen outperforms the TSS work [3] when there are subtle violations in annotation constraints and remains robust even with 20% segments missing.

S3.2 Generalization of EM-Gen to SkipTag

As briefly discussed in Sec. 4.6 of the main text, we generalized our EM-Gen method to a weaker form of annotation under SkipTag supervision. This supervision allows annotators to freely annotate a set of random timestamps spread somewhat evenly throughout the video, removing the constraint 'one timestamp per action segment' of timestamp supervision.

We have shown that our EM-Gen is robust to one missing action segment between consecutive annotated timestamps. However, there are two other possible scenarios that EM-Gen must be generalized to. Firstly, there can be a case where two consecutive timestamps are from the same action segment, i.e., '*no action boundary*' between the timestamps (depicted in Fig. F3). Secondly, there can be

Table T6: **Performance under (% of) missed action segments.** Our EM-TSS performs significantly better than [3] under missing action segments. Our EM-Gen is robust to missing segments. With 20% missing segments, EM-Gen significantly outperforms the TSS methods.

		50Salads				Breakfast				GTEA			
Err%	**Method**	$F1\{25,50\}$		Edit	MoF	$F1\{25,50\}$		Edit	MoF	$F1\{25,50\}$		Edit	MoF
5%	[3]	58.5	44.8	56.1	69.0	57.3	42.0	60.9	62.4	69.7	53.1	71.2	64.1
	EM-TSS	72.0	59.0	67.6	73.6	60.5	46.2	63.6	64.5	80.5	62.1	79.5	65.3
	EM-Gen	72.0	61.7	68.9	75.8	61.5	49.4	65.7	67.1	80.8	64.8	79.7	67.9
10%	[3]	58.1	43.9	55.2	69.7	53.0	38.2	58.0	59.5	66.3	51.1	66.5	61.6
	EM-TSS	69.2	55.9	66.0	72.1	55.8	40.8	59.6	62.5	77.0	57.3	76.4	63.0
	EM-Gen	70.9	58.3	67.4	75.0	60.9	48.4	63.7	66.4	78.6	61.7	77.3	66.5
20%	[3]	52.2	36.8	50.9	63.0	48.5	32.7	55.3	55.1	58.4	41.1	60.6	53.6
	EM-TSS	63.2	47.4	58.6	65.6	50.6	36.6	57.3	60.5	73.2	52.9	71.2	56.4
	EM-Gen	68.5	55.7	66.6	71.9	57.9	45.2	60.1	65.3	75.0	59.7	74.0	65.0

'*more than one missing segment*' between two timestamps. Here in this section, we show how we reformulate our $\mathcal{Q}$-function to take the case of '*no boundary*' into account and left the case of '*more than one missing segment*' out of the scope of our current work. However, the formulation can be generalized easily to handle more than two missing segments.

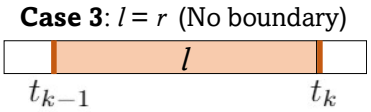

Fig. F3: A typical timestamp segment with two timestamps t_{k-1} and t_k belonging to the same action segment.

Case 3 (C_3): *No action boundary* between t_{k-1} and t_k, as timestamp segment $\mathcal{S}$ belongs to one continuous action. This case is possible only when $y[t_{k-1}] = y[t_k] = l$, i.e., the two action labels match, reducing the segment likelihood to

$$\mathcal{P}(\mathcal{S}|C_3, \boldsymbol{\Theta}) = \prod_{i=t_{k-1}}^{t_k-1} p_{i,l}^{\boldsymbol{\Theta}}.$$

When we take this new case into consideration along with our previously discussed cases (case 1, case 2 from Sec.4.3 of the main text), the new $\mathcal{Q}$-function

becomes

$$\begin{aligned}\mathcal{Q}(\boldsymbol{\Theta}, \boldsymbol{\Theta}^{(m)}) = &\sum\nolimits_s \mathbb{P}^{(m)}[C_1, s|\mathcal{D}] \cdot \log \mathcal{P}(\mathcal{S}|C_1, s, \boldsymbol{\Theta}) \\ &+ \sum\nolimits_\tau \mathbb{P}^{(m)}[C_2, \tau|\mathcal{D}] \cdot \log \mathcal{P}(\mathcal{S}|C_2, \tau, \boldsymbol{\Theta}) \\ &+ \mathbb{P}^{(m)}[C_3|\mathcal{D}] \cdot \log \mathcal{P}(\mathcal{S}|C_3, \boldsymbol{\Theta}),\end{aligned} \tag{1}$$

where $\mathbb{P}^{(m)}[C_3|\mathcal{D}]$ is the posterior probability that the timestamp segment $\mathcal{S}$ belongs to Case 3. For SkipTag supervision, when the left and right action are not the same, i.e., $l \neq r$, the posterior weights become

$$\begin{aligned}\omega_{j,l}^{(m)} &= \sum_{s>j} \mathbb{P}^{(m)}[C_1, s|\mathcal{D}] + \sum_{\tau:s_1>j} \mathbb{P}^{(m)}[C_2, \tau|\mathcal{D}] \\ \omega_{j,c}^{(m)} &= \sum_{s_1 \leq j} \sum_{s_2>j} \mathbb{P}^{(m)}[C_2, s_1, s_2, c|\mathcal{D}] \qquad \text{if } c \neq l, c \neq r,\end{aligned} \tag{2}$$

and $\omega_{j,r}^{(m)} = 1 - \omega_{j,l}^{(m)} - \sum_c \omega_{j,c}^{(m)}$. When the actions are the same, $l = r$, the first term in $\omega_{j,l}^{(m)}$ is replaced by $\mathbb{P}^{(m)}[C_3|\mathcal{D}]$.

S3.3 Posterior Probabilities of EM-Gen

In equation 1, we used the posterior probabilities to form the $\mathcal{Q}$-function. In addition, we use these posterior probabilities later in equation 2 to compute the per-frame posterior weights $\omega_{j,c}^{(m)}$. The form of the posterior probabilities is as follows:

$$\begin{aligned}\mathbb{P}^{(m)}[C_1, s\,|\mathcal{D}] &= \frac{1}{Z} \cdot \mathcal{P}(\mathcal{S}|C_1, s, \boldsymbol{\Theta}) \cdot \pi(C_1, s) \\ \mathbb{P}^{(m)}[C_2, \tau\,|\mathcal{D}] &= \frac{1}{Z} \cdot \mathcal{P}(\mathcal{S}|C_2, \tau, \boldsymbol{\Theta}) \cdot \pi(C_2, \tau) \\ \mathbb{P}^{(m)}[C_3\,|\mathcal{D}] &= \frac{1}{Z} \cdot \mathcal{P}(\mathcal{S}|C_3, \boldsymbol{\Theta}) \cdot \pi(C_3) \\ \text{where, } \; Z = &\sum_s \mathcal{P}(\mathcal{S}|C_1, s, \boldsymbol{\Theta}) \cdot \pi(C_1, s) + \\ &\sum_\tau \mathcal{P}(\mathcal{S}|C_2, \tau, \boldsymbol{\Theta}) \cdot \pi(C_2, \tau) \; + \mathcal{P}(\mathcal{S}|C_3, \boldsymbol{\Theta}) \cdot \pi(C_3).\end{aligned} \tag{3}$$

Similar to previous notations, s denotes valid boundary locations of Case 1, and τ iterates over all possible triplets (s_1, s_2, c) of Case 2, which are the start frame of the middle segment, start frame of the right segment and action class of the middle segment, respectively.

S3.4 Prior for EM-Gen

Recall that for Timestamp supervision, we could directly use the Binomial prior on the location of the boundary (start) frames derived from the Poisson prior

on the lengths of the action segments. This is because in TSS, we have the exact information on the (ordered) sequence of actions in the video implicitly from the timestamp annotations. For TSS with missed segments as well as SkipTag, however, we do not have this information as it is no longer guaranteed that all the action segments are annotated exactly once.

To calculate the prior probabilities for the current timestamp segment $\mathcal{S}_k$, we utilize the expected location of the *last* boundary, denoted as β_{k-1}, of previous timestamp segment $\mathcal{S}_{k-1}$. We naturally define $\beta_0 \equiv 0$. First, we discuss how we use β_{k-1} to calculate priors for $\mathcal{S}_k$, then we will discuss how to calculate β_k for the segment $\mathcal{S}_k$. Using β_{k-1}, we estimate the prior $\pi(C_1, s)$ for the timestamp segment $\mathcal{S}_k$ as

$$\begin{aligned}\pi(C_1, s) &= \pi(s \,|C_1, \beta_{k-1}) \cdot \pi(C_1) \\ &= \frac{e^{-\mu_l} {\mu_l}^{(s-\beta_{k-1})}}{(s-\beta_{k-1})!} \cdot \pi(C_1). \end{aligned} \tag{4}$$

This is because given the last expected boundary β_{k-1} and Case 1, the quantity $(s - \beta_{k-1})$ becomes the length of the action segment of the '*left*' action class l (refer to Figure 3 of main text). Hence, it follows a Poisson distribution Pois(μ_l), with μ_l being the mean length of the action class l. Next for Case 2, we calculate $\pi(C_2, s_1, s_2, c)$ for a triplet (s_1, s_2, c), as

$$\begin{aligned}\pi(C_2, s_1, s_2, c) &= \pi(s_1, s_2, c \,|C_2, \beta_{k-1}) \cdot \pi(C_2) \\ &= \pi(s_1 \,|C_2, \beta_{k-1}) \cdot \pi(s_2 \,|C_2, s_1, c) \cdot \pi(C_2, c) \\ &= \frac{e^{-\mu_l} {\mu_l}^{(s_1-\beta_{k-1})}}{(s_1-\beta_{k-1})!} \cdot \frac{e^{-\mu_c} {\mu_c}^{(s_2-s_1)}}{(s_2-s_1)!} \cdot \pi(C_2, c). \end{aligned} \tag{5}$$

Similar to Case 1, the quantity $(s_1 - \beta_{k-1})$ given Case 2, follows Pois(μ_l). Whereas $(s_2 - s_1) \sim$ Pois(μ_c) when it is conditioned on Case 2 and the fact that the middle segment has action class c. We set the priors $\pi(C_1) = \pi(C_3) = \frac{1}{3}$. For any $c \neq l, c \neq r$, we set $\pi(C_2, c) = \frac{1}{3(C-2)}$ if $l \neq r$, and $\pi(C_2, c) = \frac{1}{3(C-1)}$ if $l = r$, where C is the total number of classes. That is, we naively set all the case priors to be equally likely.

Finally, we update the last boundary as β_k for the current timestamp segment $\mathcal{S}_k$. We estimate it as

$$\begin{aligned}\beta_k = \beta_{k-1} \cdot \mathbb{P}^{(m)}[C_3 \,|\mathcal{D}] \; + \sum_s s \cdot \mathbb{P}^{(m)}[C_1, s \,|\mathcal{D}] \\ + \sum_{s_2} s_2 \cdot \sum_{s_1, c} \mathbb{P}^{(m)}[C_2, s_1, s_2, c \,|\mathcal{D}]. \end{aligned} \tag{6}$$

The last boundary β_k for timestamp segment $\mathcal{S}_k$, (1) stays the same as β_{k-1} for Case 3 with probability $\mathbb{P}^{(m)}[C_3 \,|\mathcal{D}]$, (2) for Case 1 is equal to s with probability $\mathbb{P}^{(m)}[C_1, s \,|\mathcal{D}]$, (3) and finally for Case 2 is equal to s_2 with probability $\sum_{s_1,c} \mathbb{P}^{(m)}[C_2, s_1, s_2, c \,|\mathcal{D}]$. Hence, the expected value combines all of these possible cases.

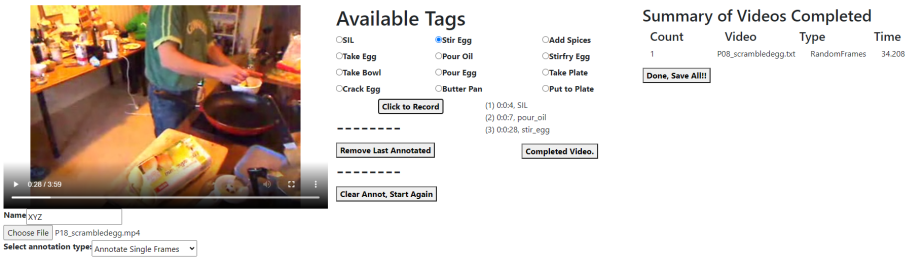

Fig. F4: **User Interface for Temporal Segmentation Annotation.** Left Panel: annotators can enter id, choose video, the type of annotation and watch the video. Middle Panel: annotators can pause and select the action-tag and record it. This panel shows a list of action tags marked thus far. Right Panel: displays the tagging time summary for completed videos.

S4 Details of user study

As mentioned in Section 5.4 of the main text, our user study revealed that full-annotation (i.e., labelling the start of every action segment in the video) takes about 90% of the video duration. Timestamp (TSS) annotation (i.e., labelling a random frame from every action segment in the video) takes about 70% of the video duration, while SkipTag annotation (i.e., labelling 7-8 random frames in a video from Breakfast) only takes about 40% of the video's duration. This means that for a 1-minute video, full-annotation takes about 54s, TSS annotation takes about 40s and SkipTag annotation takes about 21s on average.

A similar user study on TSS and full-annotation was conducted by [4]. They reported that for a 1-minute video, annotation took an average of 300s for full-annotation and 50s for TSS. Though our method requires similar amount of time for TSS annotation, it reduces the full-annotation time by 5 times compared to [4]. We speculate that the difference is likely due to the use of different annotation interfaces.

For full-supervision annotations, we found that the start times of the same action segments, marked by different annotators, had a standard deviation of ≈ 1.5 seconds or 23 frames (breakfast containing frames at 15fps). This further gives us an indication of the ambiguity in annotating the boundary frames.

Annotation Interface: We developed a simple and user-friendly custom annotation interface to perform the user study for all three annotations, which is shown in Fig. F4. We will make the annotation tool available along with our code repository.

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. Li, S.J., AbuFarha, Y., Liu, Y., Cheng, M.M., Gall, J.: Ms-tcn++: Multi-stage temporal convolutional network for action segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
3. Li, Z., Abu Farha, Y., Gall, J.: Temporal action segmentation from timestamp supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8365–8374 (2021)
4. Ma, F., Zhu, L., Yang, Y., Zha, S., Kundu, G., Feiszli, M., Shou, Z.: Sf-net: Single-frame supervision for temporal action localization. In: European conference on computer vision. pp. 420–437. Springer (2020)