A Appendix

A.1 Loss functions of SC

For completeness, we present in detail how we adjust the loss functions of SC from baseline's.

MS-TCN's Loss function. In vanilla MS-TCN [3], the author adopted a combination of classification loss and a smoothing loss:

(1) A cross entropy loss as classification loss

$$\mathcal{L}_{cls} = -\frac{1}{T}\sum_{t} \log\left(y_{t,c}\right), \tag{5}$$

where $y_{t,c}$ is the the predicted probability for the ground truth label at time t.

(2) A truncated mean squared error loss over the frame-wise log-probabilities as smoothing loss

$$\mathcal{L}_{T-MSE} = \frac{1}{TC}\sum_{t,c} \tilde{\Delta}^2_{t,c}, \tag{6}$$

$$\tilde{\Delta}_{t,c} = \begin{cases} \Delta_{t,c} : \Delta_{t,c} \leq \tau \\ \tau \quad : \text{otherwise} \end{cases}, \tag{7}$$

$$\Delta_{t,c} = \left|\log y_{t,c} - \log y_{t-1,c}\right|. \tag{8}$$

where T is the video length, C is the number of classes, and $y_{t,c}$ is the probability of class c at time t. Then the final loss function of MS-TCN is

$$\mathcal{L}_s = \mathcal{L}_{cls} + \lambda \mathcal{L}_{T-MSE} \tag{9}$$

SC's Loss function. In our stage cascade, we use the original form of loss function for fusion stage. For cascade stage i, we keep the smoothing loss $\mathcal{L}_{T-MSE}$ the same and change the distribution of cross entropy loss as classification loss as described in main paper

$$\mathcal{L}^{sc}_i = -\frac{\sum_t w^i_t \cdot \log\left(y_{t,c}\right)}{\sum_t w^i_t} \tag{10}$$

where w^i_t is the weight for cascade stage i and frame t, and it follows our cascade strategy (shown in equation (1)).

A.2 Detailed architecture of BGM

The detailed description of BGM will be given here for better reproducibility. A temporal convolutional layer can be simply denoted as $Conv(c_f, c_k, c_d, Act)$, where c_f, c_k, c_d and Act are filter numbers, kernel size, dilation factor and activation function of temporal convolutional layer. To operate on full temporal resolution, we use two version of barrier generation module here: (1) 'resized' version: resized temporal scale and $conv(512, 3, 1, ReLU) \rightarrow conv(512, 3, 1, ReLU) \rightarrow conv(1, 1, 0, Sigmoid)$; (2) 'full-length' version: full temporal scale and $conv(256, 3, 2^l, ReLU)$ with $l \in \{2, 3, 4\} \rightarrow conv(1, 1, 0, Sigmoid)$ following the setting of the baseline [3] to exponentially increase receptive field. The last layer with sigmoid activation will generate boundary probability.

BGM's Loss function. As mentioned in main paper, we construct ground-truth of BGM from segmentation ground-truth following BSN [19]: We denote Φ_ω and Ψ_ω as feature sequence and annotations within the window respectively. For ground truth action instance $\varphi_g = (t_s, t_e)$ in Ψ_ω, we denote both starting region $r_g^s = [t_s - d_g/10, t_s + d_g/10]$ and ending region $r_g^e = [t_e - d_g/10, t_e + d_g/10]$ as boundary regions r_g, where $d_g = t_e - t_s$. Taking Φ_ω as input, BGM generates probabilities sequence P_ω with length l_w, just to be consistent to [19] here). For each temporal location t_n, we denote its region as $r_{t_n} = [t_n - d_s/2, t_n + d_s/2]$ and get the probability scores p_{t_n} from P_ω, where $d_s = t_n - t_{n-1}$ is temporal interval between two snippets. Then for each r_{t_n}, we calculate its IoP ratio with all boundary region r_g, where IoP is defined as the overlap ratio with ground-truth (boundary region here) proportional to the duration (assigned temporal interval of location t_n here). Thus we can represent information of t_n as $\phi_n = (p_{t_n}, g_{t_n})$, where g_{t_n} is maximum matching overlap IoP of all boundary regions.

We use the same binary logistic regression loss as BSN [19] for classify the action boundary, denoted as:

$$L_{bd} = \frac{1}{l_w} \sum_{t_n=1}^{l_w} \left(\alpha^+ \cdot b_{t_n} \cdot \log(p_{t_n}) + \alpha^- \cdot (1 - b_{t_n}) \cdot \log(1 - p_{t_n}) \right) \tag{11}$$

where l_w is the length of feature sequence, g_{t_n} is temporal intersection over union (tIoU) of location t_n and boundary region, and $b_{t_n} = \mathbb{I}(g_{t_n} > 0.5)$ is a indicator function to select location t_n where $g_{t_n} > 0.5$ as positive sample (in practice, local maximal strategy is also used to remove repeated b_{t_n}). Let $l^+ = \sum g_t$ and $l^- = l_w - l^+$, then we set $\alpha^+ = \frac{l_w}{l^+}$ and $\alpha^- = \frac{l_w}{l^-}$.

A.3 Additional ablation studies of LBP

By more ablation study, we show that our LBP is insensitive to most of hyper-parameters except for pooling window size illustrated in our main paper, thus we believe that LBP is robust for temporal action segmentation task.

Note that we use the same hyper-parameters as BSN [19] to construct our BGM, except the architecture of 'full-length' BGM and threshold 0.5 for selecting barriers (0.9 in [19]) mentioned in main paper. Although we may get better performance by carefully adjusting the hyper-parameters because of the data distribution gap between datasets of temporal action segmentation task (ours) and temporal action detection task ([19]'s), our LBP's superior performance over baseline's smoothing loss and two heuristic smoothing operator also shows the robustness of our novel LBP.

In our Local Barrier Pooling, we use the following weighted sum to smooth the prediction:

$$y'_{t,c} = \frac{y_{t,c} + \sum\limits_{s\in\{-1,+1\}} \sum\limits_{\beta=1}^{L} y_{t+s\cdot\beta,c} \exp(-\alpha \sum\limits_{j=1}^{\beta} b_{t+s\cdot j})}{1 + \sum\limits_{s\in\{-1,+1\}} \sum\limits_{\beta=1}^{L} \exp(-\alpha \sum\limits_{j=1}^{\beta} b_{t+s\cdot j})} \tag{12}$$

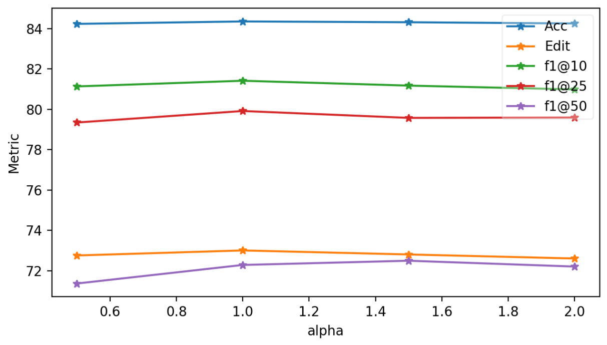

Fig. 7: Metrics as functions of α in 'resized' LBP on 50Salads (mid).

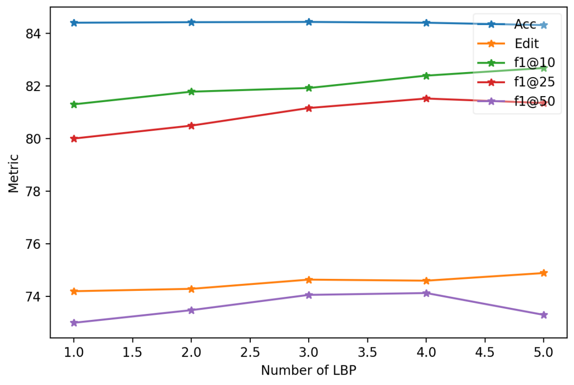

Fig. 8: Metrics as functions of number of LBP on 50Salads (mid).

where $y_{t,c}$ is segmentation confidence score in location t and class c predicted by frame-wise classification network (i.e., Stage Cascade), b_t is barrier strength, α controls the decay rate of weights when passing through a barrier and the pooling window size is $2L + 1$.

Study on LBP's α. In LBP, α controls the decay rate of weights when passing through a barrier. Fig. 7 illustrates that the change of α will only cause very little influence on the performance. In our experiments we set $\alpha = 1$ for 'resized' LBP and $\alpha = 0.2$ for 'full-length' LBP as default.

Study on the Number of LBP. We can use multiple LBPs on the output of Stage Cascade. Fig. 8 illustrates that the performance tends to increase a little before 4 LBPs are performed and starts to decrease a little when the number of LBP is beyond 5. Our proposed LBP in ideal condition should have precise boundaries and infinity barrier strength, where multiple LBPs will generate the

Table 6: Study on LBP's location on 50Salads (mid) dataset.

50 Salads (mid)	F1@{10,25,50}			Edit	Acc
BCN w/o inner LBP	**82.3**	81.1	73.7	**74.3**	**84.4**
BCN w/ inner LBP	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**

Table 7: Comparison of LBP and λ in [3] on 50Salads (mid) dataset. (* reported in [3])

50 Salads (mid)	F1@{10,25,50}			Edit	Acc
MS-TCN ($\lambda = 0.05$)*	74.1	71.7	62.4	66.6	80.0
MS-TCN ($\lambda = 0.15$)*	76.3	74.0	64.5	67.9	80.7
MS-TCN ($\lambda = 0.25$)*	74.7	72.4	63.7	**68.1**	78.9
MS-TCN w/ LBP	**78.3**	**75.9**	**66.1**	**68.1**	**81.5**

same result as single LBP. However, although LBP in practice is indeed sensitive to action boundaries, the strength of barriers can not be unlimited high. As a result, too many times of LBP may cause over-smooth and affects the performance a little, but we want to emphasize that all of results in this experiment outperforms the performance without LBP by large margin. We believe that number of LBP does not affect our performance a lot and we use 4 LBPs in other experiments.

Study on the Location of LBP. Except for placing LBP after the output of fusion stage in Stage Cascade, we can also put LBP in other locations if the semantic information is continuous among neighborhood. We use a 'full-length' LBP of $\alpha = 1$ and pooling window size as 7 on the collection of all cascade stages' output (i.e., input of fusion stage) as comparison, denoted as inner LBP. Table 6 shows additional LBP will only cause very little increase of F1 score while other metrics only vary less than 0.1. We conclude that the location of LBP does not affect our BCN's performance a lot either. We also find that inner LBP helps to reduce the fluctuation of performance when the model converges to final result in training process. In our other experiments, we use an inner LBP as default.

Comparison between LBP and weight of smoothing loss in [3]. In the main paper, we have compared LBP and two heuristic smoothing operator: gaussian smoothing and average pooling. To further justify our LBP's performance, we compare LBP and the weight of smoothing loss in [3] based on the vanilla MS-TCN architecture, shown in Table.7. By giving more weight λ for smoothing loss (show in equation (2)), we can enforce the output of network being more temporally consistent by penalizing the difference of confidence score in adjacent locations. MS-TCN achieves best performance in $\lambda = 0.15$ and can not get further improvement when $\lambda = 0.25$ except the edit score. In contrast, LBP consistently improves all the metrics. Note that our LBP will have more improvement on the results when we introduce our SC into action segmentation task for their complementary effects on improving segmentation performance based on the analysis in main paper.

Based on the above analysis, we see that our proposed novel LBP is insensitive to most of hyper-parameters, thus robust for temporal action segmentation task. LBP significantly improves the smoothness of predictions, which is reflected in F1 score and edit score.

A.4 Qualitative result of GTEA dataset

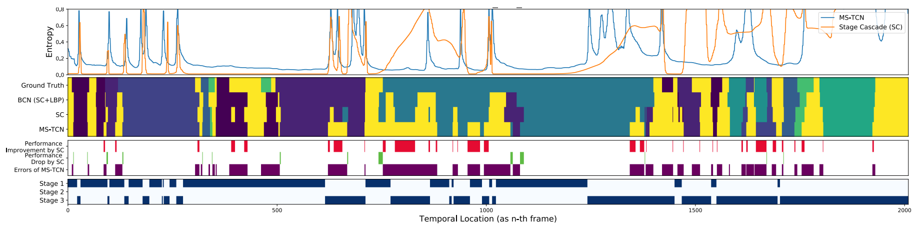

Fig. 9: Qualitative result of S1_Tea_C1 from GTEA dataset. There are several rows: (1) entropy of SC vs. baseline; (2) action segmentation results of groundtruth, BCN, SC and baseline [3]; (3) SC's improvement (red) and decline (green) over baseline, and baseline's error (purple); (4) The cascade stage among stage 1,2 or 3 which dominates the weight is in blue.