Boundary-Aware Cascade Networks for Temporal Action Segmentation

Zhenzhi Wang[1], Ziteng Gao[1], Limin Wang[1][0000−0002−3674−7718], Zhifeng Li[2], and Gangshan Wu[1]

[1] State Key Laboratory for Novel Software Technology, Nanjing University, China
[2] Tencent AI Lab, Shenzhen, China
{zhenzhiwang,gzt}@outlook.com, {lmwang, gswu}@nju.edu.cn, michaelzfli@tencent.com

Abstract. Identifying human action segments in an untrimmed video is still challenging due to boundary ambiguity and over-segmentation issues. To address these problems, we present a new boundary-aware cascade network by introducing two novel components. First, we devise a new cascading paradigm, called Stage Cascade, to enable our model to have adaptive receptive fields and more confident predictions for ambiguous frames. Second, we design a general and principled smoothing operation, termed as local barrier pooling, to aggregate local predictions by leveraging semantic boundary information. Moreover, these two components can be jointly fine-tuned in an end-to-end manner. We perform experiments on three challenging datasets: 50Salads, GTEA and Breakfast dataset, demonstrating that our framework significantly outperforms the current state-of-the-art methods. The code is available at https://github.com/MCG-NJU/BCN.

Keywords: Temporal action segmentation · Cascade strategy · Smoothing operator · Untrimmed video

1 Introduction

Understanding human actions in videos is of great importance for many real-life applications such as surveillance and interactive robotics. Recognizing actions from short trimmed videos has achieved great performance [28, 24, 31, 1, 29, 30]. However, densely labeling all the frames in a long untrimmed video is still challenging compared with action recognition. Recent works on temporal action segmentation mainly focus on capturing complex temporal structure with enriched temporal modeling. Typical methods include bi-directional LSTM networks [9, 25] and temporal convolution with encoder-decoder structure [16, 13] or dilated convolution [13]. The recent state-of-the-art MS-TCN [3] stacks multiple dilated convolutions to enlarge temporal modeling capacity and achieve extremely large receptive field in order to operate on the full temporal resolution.

Although modeling complex temporal structures is vital for segmenting hard-to-recoginze frames, simply increasing modeling capacity incurs overfitting problems for simple frames aside from more computation cost. Moreover, some frames

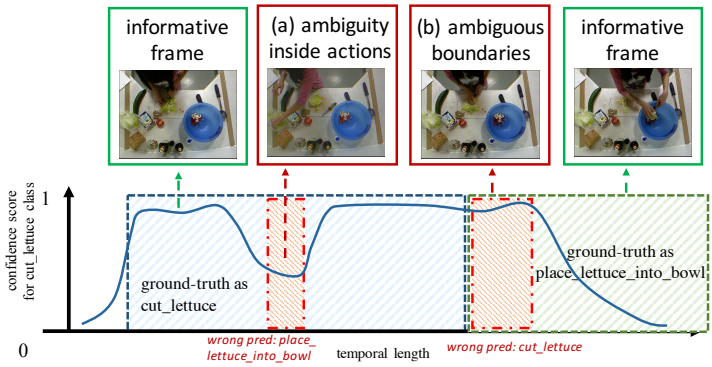


Fig. 1: Illustration of challenges: (a) the woman taking new lettuce in a long 'cut lettuce' action leads to a short misclassified 'place lettuce into bowl' result; (b) visual similarity between cutting lettuce and holding lettuce with a knife and a hand makes action boundary ambiguous. Our motivation is to tackle (b) by our stage cascade (SC) which greatly improves accuracy near boundaries, while (a) may exacerbate over-segmentation errors for the enhanced discrimination ability of SC, which will be alleviated by our proposed boundary-aware temporal regularizer: local barrier pooling.

are ambiguous other than difficult such as action boundaries (e.g., Fig. 1 (b)) due to sudden label changes but gradual transitions of visual features. Training a single model on these inconsistent samples tend to output low confident or even wrong predictions for ambiguous frames. Given the concurrent existence of informative and ambiguous frames in a video, a more dynamic temporal modeling method is desired to cope with these problems.

To tackle the challenge above, we design a new temporal action segmentation method, termed as *Stage Cascade* (SC), which leverages cascade strategy on stage level and enable networks to predict frames through different and adaptive stages based on their complexities. Different from previous works on the enlarged modeling capacity but fixed receptive field [13, 16, 3], SC provides a new perspective for modeling adaptive temporal structure. Stages are allocated for each frame on the criterion of accuracy to modulate the stage-wise effective receptive fields. The unique design of SC enables to dynamically model sudden label changes and progressively produce better and more confident predictions in a simple-to-hard manner, where early stages focus on recognizing simple frames and late stages pay more attention to ambiguous and difficult frames.

Another common challenge still exists in this task despite the dynamic temporal modeling of SC: the over-segmentation errors (e.g., Fig. 1 (a)). This problem is probably even worse in SC because of noise sensitivity and fluctuating stage assignment during inference. Thus, it is expected to devise a more adaptive and effective temporal smoothing operation to suppress the exacerbated over-segmentation issues. Previous methods commonly relieve this problem by prior knowledge such as additional temporal smoothing loss function [3]. Instead, we argue that action boundaries serve as powerful signals for temporally regularizing smooth predictions of action instances, as they naturally indicate intervals

for semantic consistency inside and border-crossing discrepancy. Accordingly, we present an adaptive temporal regularizer which leverages the boundary information to ensure temporal consistency of instances, termed as *Local Barrier Pooling* (LBP). LBP predicts action boundaries from a binary classification network, whose supervision signals are derived from the segmentation ground truth. Then, LBP performs local aggregation of frame-level action predictions, where the aggregating weights are video-specific and boundary-aware. In this sense, LBP is able to greatly reduce unexpected over-segmentations by smoothing noisy predictions with confident ones.

We evaluate our framework on three challenging datasets for action segmentation: 50Salads [26], GTEA [5] and Breakfast dataset [11]. Experimental results demonstrate that the combination of our SC and LBP yields a notable performance gain against with the strong baseline [3], which is the current state-of-the-art method. In particular, our method achieves about 4% gain in frame-wise accuracy for all datasets and a consistent improvement about 4% for GTEA, 6% for 50salads and 10% for Breakfast in F1 score as the scale of datasets increases. In summary, our paper makes two main contributions:

- Our cascade design is the first attempt in the task of temporal action segmentation by enabling temporal models to have dynamic temporal modeling and achieve more confident results. This new cascade design provides a general solution to improve frame-level recognition accuracy over the existing multi-stage action segmentation methods.
- We explicitly improve the smoothness of frame-wise predictions by cooperating action boundary information with them for the first time in action segmentation task. To achieve this, we propose a novel temporal regularizer *Local Barrier Pooling* (LBP) which alleviates over-segmentation problem and meanwhile avoids reducing segmentation accuracy. Our LBP is differentiable which enables end-to-end training of our framework.

2 Related Work

Temporal Action Segmentation. Segmentation methods typically use temporal models for frame labeling upon extracted frame-level features. For example, Fathi *et al.* [4] modeled actions by the change of objects' states. Lea *et al.* [13] presented a temporal convolutional network for action segmentation and detection using an encoder-decoder architecture to capture long-range dependencies. Lei *et al.* [16] introduced deformable convolutions into [13] and added a residual stream with high temporal resolution. Farha *et al.* [3] extended dilated temporal convolution in speech synthesis [20] to action segmentation for capturing long-term dependencies and operated it on the full temporal resolution. Gammulle *et al.* [6] proposed a conditional GAN model to utilize multiple modalities for better extraction of salient details from environmental context. These works mainly focused on improving receptive field for modeling long-term dependency with encoder-decoder structure [16, 13], dilated convolution [3] or deformable convolution [16]. Different from these methods, our framework tackles the problems

of inaccurate boundaries and misclassified short actions caused by existing long-term temporal modeling methods via adaptive receptive field, and provides a general improvement on existing multi-stage temporal model. Our method also provides a general smoothing operator to solve over-segmentation problem inside long actions.

Temporal Action Detection. Many of action detection methods share similar temporal modeling with our task. Singh *et al.* [25] used a multi-stream bi-directional recurrent neural network for fine-grained action detection. Yeung *et al.* [32] used reinforcement learning and RNNs to predict temporal intervals of actions based on glimpses of a small portion of the video. Zhao *et al.* [33] modeled temporal structure of actions by structured temporal pyramid. Gao *et al.* [7] extended Faster R-CNN [21] to action detection by iteratively regressing boundaries. Lin *et al.* [19] employed binary classification to predict boundaries of action instances in a sequence of temporal locations and evaluate proposals combined by these classified boundaries. Although two tasks are similar, their methods can not be directly applied to the other task for both different goal of output and different metrics.

Deep Learning Cascade. Cascade networks have been studied in detection [17], pose estimation [27] and semantic segmentation [18]. Li *et al.* [17] adopted CNN cascade for face detection, which quickly rejects false detections in early stages and refines detections in later stages. DeepPose [27] employed a divide-and-conquer strategy and designed a deep regression cascade framework for pose estimation. Li *et al.* [18] reduced computational cost and improved accuracy of semantic segmentation by distinguishing easy pixels from the hard ones and only propagate pixels with low confidence to subsequent networks. Our cascade framework differs from previous ones in adjusting weights of loss functions and combining stages' predictions for parts of a sample (i.e., frames of a video) similar to attention mechanism, but not accept or reject the whole sample [17] or a part of the sample [18] in each step.

3 Boundary-Aware Cascade Networks

As analyzed in Sec.1, we observe that temporal action segmentation is challenging mainly in two situations: (1) ambiguous frames near action boundary or sudden actions; (2) ambiguous frames inside a long action. To address these issues, as shown in Fig. 2, we present a unified framework by designing a new stage cascade architecture and a novel local barrier pooling. The stage cascade architecture focuses on learning a progressively weak-to-strong frame-level classifier, where early stage recognize informative frames with weak capacity and later stage pay more attention to ambiguous frames with stronger capacity. The local barrier pooling presents a new smoothing technique by leveraging explicit action instance boundary with a attentive aggregation operation. These two new modules are unified in our *Boundary-aware Cascade Network* (BCN) framework with a two-branch architecture and the whole pipeline can be easily optimized in an end-to-end manner.

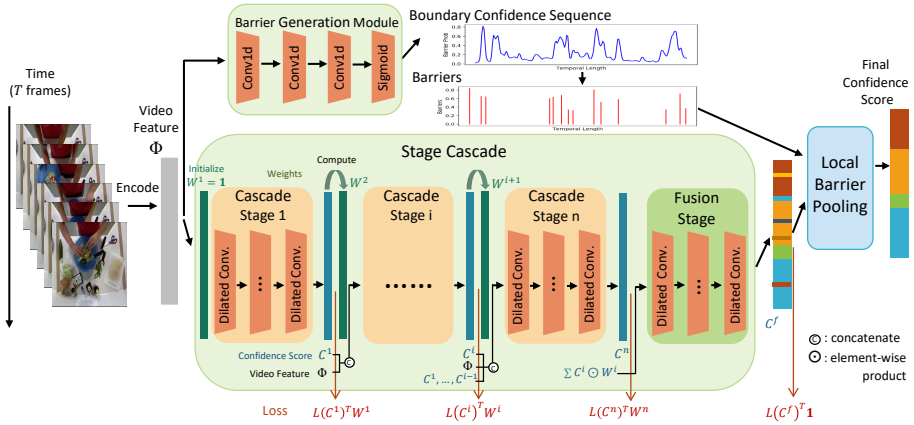


Fig. 2: Overview of our framework. Given an untrimmed video, we first encode it into a feature sequence Φ. *Stage Cascade* handles the video feature in cascade manner: cascade stages receive Φ and all previous stages' outputs, then predict frame-wise confidence scores, which will determine the weights of loss functions over frames and aggregation of cascade stages' outputs for fusion stage; *Barrier Generation Module* evaluates the boundary probabilities of each temporal location and selects barriers for our novel temporal regularizer: local barrier pooling.

Our BCN provides a general and adaptive framework to boost temporal action segmentation performance in videos for any multi-stage models. To demonstrate the effectiveness of BCN, we use the state-of-the-art MS-TCN [3] as the backbone, and aim to improve action segmentation performance over a very strong baseline. Sec. 3.2 illustrates how to adapt the existing action segmentation approach into our *stage cascade*. Sec. 3.3 shows how our proposed novel *Local Barrier Pooling* is applied to action segmentation task using boundary information. Sec. 3.4 introduces the training procedure of our framework.

3.1 Video Encoding

To save the memory consumption, we first extract visual features with an off-the-shelf pre-trained video network. Given an untrimmed video with T frames $X_{1:T} = \{x_t\}_{t=1}^{T}$ as the input, our goal is to predict the class label for frames $C_{1:T} = \{c_t\}_{t=1}^{T}$. Video encoding aims to obtain a condensed video representation capturing appearance and motion patterns of video clips. In particular, following the baseline [3] for fair comparison, we use I3D [1] without fine-tune as our video encoder ϕ to generate a sequence of feature vectors $\Phi = \{\phi(x_1), \phi(x_2), ..., \phi(x_T)\} \in \mathbb{R}^{T \times D}$ where $D = 2048$ is feature dimension. Then extracted video features are fed into BCN for temporal action segmentation.

3.2 Stage Cascade

The goal of stage cascade is to generally boost the performance of frame-level classification network by treating video frames with modules of different com-

plexities. Specifically, for informative frames in long action segments, we can use a weak model of low capacity yet still able to capture long-term temporal dependency in order to prevent over-fitting; and for ambiguous frames near action boundaries or in sudden actions, we should devise a stronger model of high capacity with adaptive receptive field and focus on ambiguous frames for a more precise prediction. Our stage cascade adaptively process different frames with different stages and obtain the frame-level prediction that mainly rely on the corresponding stage. In practice, cascade strategy will automatically assign a weight distribution to all the cascade stages for each frame. We initialize the weights of first stage w_t^1 with 1 for all frames and update the i^{th} $(i \geq 1)$ cascade stage's weight w_t^i for t^{th} frame as follows:

$$w_t^i = \begin{cases} e^{-c_t^{i-1}} w_t^{i-1} & \text{if } \ c_t^{i-1} \geq \rho, \\ e^{c_t^{i-1}} w_t^{i-1} & \text{if } \ \forall j \leq i-1, c_t^j < \rho \\ w_t^{i-1} & \text{if } \ c_t^{i-1} < \rho \ \text{ and } \ \exists j < i-1, c_t^j \geq \rho \end{cases} \tag{1}$$

where ρ is a parameter and c_t^i is confidence score of t^{th} frame for i^{th} cascade stage. In equation (1), we mainly adjust the weights for the next stage by the factor $e^{c_t^{i-1}}$: increasing weights by $e^{c_t^{i-1}}$ for less confident frames, and decreasing weights by $e^{-c_t^{i-1}}$ for very confident frames. In addition, we enforce that exactly one stage should dominate the weight among n stages for each frame, so we will stop increasing weights again once any earlier stage shows enough confidence for it. The prediction of all cascade stages will be aggregated as the input of *fusion stage* according to the weight matrix $\mathbf{w}_{i,t}$ as follows:

$$c_t^f = \frac{\sum_i w_t^i c_t^i}{\sum_i w_t^i}, \tag{2}$$

The fused classification score c_t^f combine the outputs of different stages adaptively for each frame and it will be passed to the fusion stage to yield final prediction of stage cascade. The fusion stage aims to smooth frame-wise classification results and generate more reasonable temporal segmentation result. Following the common practice in action segmentation [3], we add loss functions for all cascade stages and fusion stage to make training converge stably, which are composed of a classification loss and a smoothing loss in [3]. For smoothing loss, we use the same form as baseline for all cascade stages and the fusion stage. For classification loss, we keep fusion stage's loss the same with baseline (equation (3), left) and adjust the distribution of loss for each cascade stage over frames (equation (3), right) according to weight matrix $\mathbf{w}_{i,t}$ get in equation (1). Note that the weight matrix is aggregated in the direction of i for fusing classification score and t for adjusting distribution of loss, respectively.

$$\mathcal{L}_{baseline} = \frac{1}{T} \sum_t -\log\left(y_{t,c}\right), \quad \mathcal{L}_i^{SC} = -\frac{\sum_t w_t^i \cdot \log\left(y_{t,c}\right)}{\sum_t w_t^i} \tag{3}$$

Based on the analysis above, over-segmentations are more likely to arise for SC being more flexible than baseline and there are many stage switches, which leads to the input of fusion stage being less smooth. Moreover, when encountering the situation analyzed above (Fig. 1 (a)), all methods, even with

additional smoothing loss function in [3], can lead to over-segmentation errors, which is a common challenge to this task. To generally improve smoothness of action segmentation methods adaptively, we propose the following local barrier pooling.

Implementation details. Our stage cascade is composed of several cascade stages and a fusion stage, where each stage is a SS-TCN in [3]. In MS-TCN [3], each stage only receives confidence score of the last stage as input except that the first stage receives video feature. Different from MS-TCN, our cascade stages take concatenation of video feature and all previous stages' outputs (similar to the structure of DenseNet [10]) as input and evaluates current frame-wise classification score. For the weight adjustment strategy, we use confidence score of ground truth class to update weights in training and the maximum confidence score among classes in testing. The confidence-score-selection gap between training and testing may also lead to more over-segmentation problems.

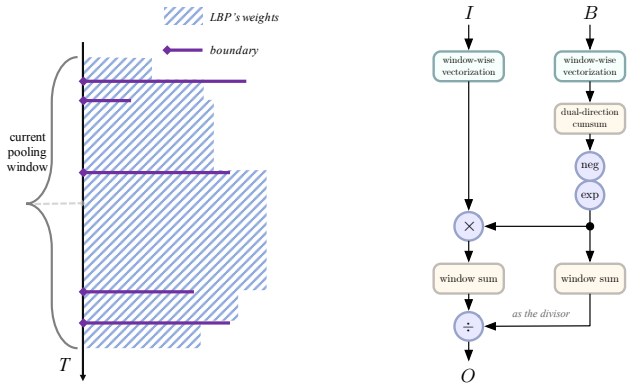
3.3 Local Barrier Pooling

To avoid over-segmentation risk with more powerful discriminative ability of the proposed Stage Cascade, we need to ensure the temporal consistency of predictions inside an action. Previous work has shown that auxiliary losses of temporal consistency [3] might be beneficial for good segmentation result. However, we find the help of auxiliary losses is limited and implicit, and temporal models including recent state-of-the-art model [3] still tend to over-segment actions.

We resort to a more explicit approach, i.e., the smoothing operator inside networks for consistent frame-wise predictions within the same action instance. Heuristic smoothing operators like gaussian smoothing and average pooling with fixed window sizes may be effective only when the action duration is much longer than window sizes or there happens to be only one action instance in a video, which is a tight restriction. Instead, we expect the smoothing operator to be separative between different action instances and consistent inside one action instance, meanwhile have adaptive smoothing window size with regard to different action duration.

The action boundary gives us a good point to achieve both separative and adaptive properties for expected smoothing procedure since a action boundary naturally indicates a start or end of an action. Motivated by adaptive pooling operator aware of spatial importance [8], we design the local barrier pooling (LBP), a smoothing operator aware of action boundaries to ensure the consistency of predictions within action instances. At a macro level, LBP regards action boundaries as barriers in the diffusion of each class's actionness. LBP can be simply decomposed into two steps after the prediction of existing models: *First*, train a classification network to predict boundaries and then select temporal locations with high confidence to be boundaries. *Second*, compute the weighted sum of predictions in a local pooling window where weights are aware of and adaptive on barriers across from the pooling center.

Local Barrier Pooling. LBP generates smoother predictions from the output of temporal model by averaging predictions among neighborhood with adaptive



(a) (b)

Fig. 3: (a) Visualization of our proposed Local Barrier Pooling in "hard" version. Given barriers (denoted as purple bar), for frame located in dash line, the unnormalized weights of LBP in a local window is shown as shadow region. (b) Architecture of LBP, in which I is frame-wise confidence scores, B is barriers and O is output of LBP.

weights aware of action boundaries. For each frame, LBP utilizes a local pooling window centering in current frame and from center calculates weights in two directions. As shown in Fig. 3, LBP decreases the weight by an adaptive ratio when it meets a barrier. The output of LBP $y'_{t,c}$ in temporal location t and class c is formulated as:

$$y'_{t,c} = \frac{y_{t,c} + \sum\limits_{s\in\{-1,+1\}} \sum\limits_{\beta=1}^{L} y_{t+s\cdot\beta,c} \exp(-\alpha \sum\limits_{j=1}^{\beta} b_{t+s\cdot j})}{1 + \sum\limits_{s\in\{-1,+1\}} \sum\limits_{\beta=1}^{L} \exp(-\alpha \sum\limits_{j=1}^{\beta} b_{t+s\cdot j})} \tag{4}$$

where $y_{t,c}$ is the frame-wise confidence score predicted by networks, b_t is barrier strength, α controls decay rate of the weight and the pooling window's length is $2L+1$. The weighted sum in the pooling window is aggregated to two directions as $s \in \{-1, +1\}$ (temporally forward and backward). By setting barriers heuristically, common smoothing methods such as gaussian smoothing and average pooling can be seen as special cases of LBP: i) if all barriers are set to 1, weights will decay at the speed of $e^{-\alpha x}$, thus similar to gaussian smoothing at $e^{-\frac{x^2}{2\sigma}}$; ii) if there are no barriers, weights will be uniform distribution which is identical to average pooling. Different from heuristic smoothing methods, our LBP introduce the boundary information b_t to parameterize its weights, which is sample-dependent. Evaluation on LBP and two heuristic smoothing methods shows that our boundary-aware smoothing operator LBP achieves better performance than its two special cases.

Barrier Generation Module. To provide input-dependent boundary information for LBP, we use temporal evaluation module (TEM) of BSN [19] as Barrier Generation Module (BGM) upon the extracted video feature sequence

$\Phi = \{\phi(x_t)\}_{t=1}^{T}$, which is a binary classification model for boundaries. The original form of TEM operates on a fixed-length feature sequence, yet it makes joint training of SC and BGM unstable. So we adjust two version of BGM (1) 'resized' version: we resize input scale to fixed-length l_w by linear interpolation and use identical network to TEM. We select temporal location t according to confidence score p_t outputted by BGM where $p_t > 0.5$ and is a local maximal ($p_t > p_{t-1}$ and $p_t > p_{t+1}$) as barriers, then resize back to original scale T by nearest neighbor interpolation. (2) 'full-length' version: we use original scale T and replace convolution layers of TEM with dilated convolution (dilation of 2^l, $l \in \{2, 3, 4\}$) to be BGM, then we keep all the p_t as barriers.

We keep all hyper-parameters the same as [19] except '$p_t > 0.5$' (0.9 in [19]) for the following reason: unlike TEM, BGM doesn't seek for best precision of detecting boundaries, but it hopes to provide complete barrier information for LBP to smooth predictions inside action instances and meanwhile do not harm accuracy among boundary regions. So we choose common practice as '$p_t > 0.5$' since our LBP is insensitive to few false positives. Selecting local maximal is used for removing repeated barriers near one boundary. We keep all the p_t in 'full-length' version because of training stability and the same reason above.

3.4 Training BCN

Training our BCN has two parts: pre-training Barrier Generation Module (BGM) and joint training Stage Cascade (SC) and BGM. We first construct action boundary ground-truths on segmentation annotations following [19] and pre-train our BGM by binary classification loss. The purpose of pre-training BGM is to provide accurate boundary predictions at the start of joint training, and to fully optimize BGM's parameters because of different convergence rates of SC and BGM. Then we train SC and BGM jointly on original ground-truths only using frame-wise classification loss, where parameters of BGM can also be fine-tuned by backward gradients because our LBP is differentiable.

Stage Cascade. Follow the baseline [3], our loss functions are (1) cross-entropy loss for classification, which will be adaptively adjusted for each frame following equation (3); (2) truncated mean squared error over the frame-wise log-probabilities for smoothing. Please refer to [3] or our appendix for details.

Barrier Generation Module. Given a ground truth sequence Ψ with temporal length l, we accumulate frame-wise annotations as segments $\varphi_g = (t_s, t_e)$ (in which $d_g = t_e - t_s$) and use the 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. We assign a duration of $\frac{1}{l}$ to each frame and calculate the temporal intersection over union (tIOU) between each frame and boundary regions as the annotation. We use a binary logistic regression loss for classifying boundaries following [19].

Training Details. For LBP, we use a pooling window of 39, 99 and 159 in GTEA, 50Salads and Breakfast dataset respectively. We set $\alpha = 1$ in 'resized' version and $\alpha = 0.2$ for 'full-length' version. Our experiment shows that α has little influence on predictions because BGM will adaptively adjust the value of barriers in joint training. We use threshold $\rho = 0.8$ to train SC and 4 stages in

MS-TCN [3] as the backbone of SC for fair comparison. We set the number of channels to 256, and then keep all other hyper-parameters the same with [3]. For joint training, we use Adam optimizer with a learning rate 10^{-3} and multiply it by 0.3 every 20 epoch for 50Salads and GTEA dataset and a learning rate $5 \cdot 10^{-4}$ and multiply it by 0.3 every 30 epochs for Breakfast dataset. For pre-training BGM, we use learning rate 10^{-3} for 50Salads and GTEA, and $5 \cdot 10^{-4}$ for Breakfast, then multiply them by 0.3 every 100 epochs. Despite larger epochs for training, the computational cost of BGM is much smaller compared to the main network SC.

4 Experiments

Dataset. We evaluate our proposed BCN on three challenging action segmentation datasets: 50Salads [26], GTEA [5] and the Breakfast dataset [11].

The **50Salads** contains 50 videos of 25 people preparing salads in kitchen environment with 17 action classes in the mid-level. Each video contains 9000 to 18000 frames and 20 action instances on average such as `cut tomato`.Although this is a multi-modal dataset, we only use RGB data in videos. We perform 5-fold cross-validation and report the average results for evaluation.

The **GTEA** is composed of 28 egocentric and dynamic-view videos and includes four subjects performing seven daily activities. We utilize 11 action classes including background class and perform 4-fold cross-validation for evaluation.

The **Breakfast** dataset is among the largest dataset for action segmentation task, which has 1,712 videos of cooking breakfast in the kitchen environment with a overall duration of 66.7h. Overall, there are 48 different actions where each video contains 6 action instances on average. We use the standard 4-fold cross-validation for evaluation.

Evaluation Metrics. For all the datasets, we report the following evaluation metrics as in [13]: frame-wise **accuracy**, segmental **edit score** and the segmental **F1 score** at temporal intersection over union (tIoU) thresholds 0.10, 0.25 and 0.5, denoted by $F1@\{10, 25, 50\}$. The commonly used accuracy fails to take the temporal structure of the prediction into account and does not reflect over-segmentation errors, so results with large amount of action segments against temporal continuity in human actions can still score high. So we also adopt edit score proposed by [13] which penalizes over-segmentation errors and F1 score proposed by [15] which is similar to mean average precision (mAP) widely used in detection task.

4.1 Study on SC and LBP

In this section, we demonstrate the ability of our proposed SC and LBP by comparing to their variants and other counterparts. To justify our framework's ability, we specify the following baselines and BCN variants: (1) MS-TCN: backbone with 4 stages and 10 layers per stage; (2) MS-TCN w/ feature: video feature passing into each stage, which provides the same information with BCN. Another

Table 1: Comparison with baseline and BCN's variants on 50Salads (mid). The 1st and 2nd of each criterion are boldfaced and underlined respectively. (* reported in [3])

Methods	F1@{10,25,50}			Edit	Acc
MS-TCN*	76.3	74.0	64.5	67.9	80.7
MS-TCN w/ feature*	56.2	53.7	45.8	47.6	76.8
MS-TCN (5 stages)*	76.4	73.4	63.6	69.2	79.5
MS-TCN (12 layers)*	77.8	75.2	66.9	69.6	80.5
Stage Cascade	56.4	54.3	48.9	52.6	<u>83.4</u>
MS-TCN w/ LBP	78.3	75.9	66.1	68.1	81.5
MS-TCN w/ attention&LBP	<u>78.9</u>	<u>77.2</u>	<u>68.5</u>	<u>71.3</u>	82.7
BCN (SC w/ LBP)	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**

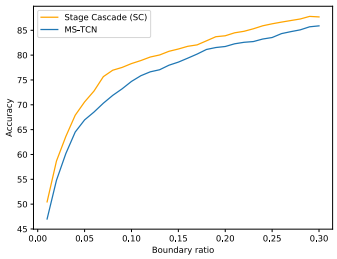


Fig. 4: Stage Cascade's accuracy gain over baseline.

Table 2: LBP and two heuristic smoothing operators on 50Salads (mid).

Smoothing Operators	F1@{10,25,50}			Edit	Acc
Average (all barriers set as '0')	80.1	77.3	69.1	72.7	82.4
Gaussian-like (all barriers set as '1')	77.0	74.6	64.9	68.7	82.5
LBP (barriers from BGM)	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**

two more stronger form of MS-TCN: (3) MS-TCN w/ 5 stages; (4) MS-TCN w/ 12 layers per stage; (5) Stage Cascade: our SC with 3 cascade stage and 1 fusion stage; (6) MS-TCN w/ LBP: (2) with LBP as post-processing; (7) MS-TCN w/ attention&LBP: (6) with traditional attention mechanism where weights are predicted from each stage itself (i.e., additional 1-dim of output).

Comparison between BCN and its important counterparts are summarized in Table. 1. Baselines and variants (1) - (7) are placed in the first seven rows and BCN is placed in the last row. In our experiment, BCN has SC, 1 embedding 'full-length' LBP in joint training and testing, and 4 times of 'resized' LBP as post-processing as default based on our ablation study. We have some observation here from Table. 1. Firstly, MS-TCN with more information (i.e., video feature) passing to higher stages will not lead to performance gain while our SC will. Secondly, SC increases segmentation accuracy and damages F1 score in the same time, implying that it tends to predict over-segment predictions without LBP for challenges shown in Fig. 1 (a) because of enhanced discriminative ability. Thirdly, LBP itself can improve both F1 score and accuracy without SC by correcting over-segmentation errors, which is consistent with our analysis. Furthermore, LBP can compensate the exacerbated over-segmentation problem caused by SC. With the help of LBP, our cascade strategy is superior to commonly used attention mechanism which does not consider the relations between stages and outputs weights by each stage separately.

To reveal the reason behind SC's improvement, we collect the accuracy gain over baseline about distances from boundaries. Similar to the construction of groundtruth in BGM, we represent regions by the incremental part when the boundary ratio increases. Comparison between accuracy of SC and the baseline in Fig. 4 show that SC mainly improves accuracy near boundary regions, which is consistent to our analysis in Sec. 3.2.

Table 3: Study on probability thresholds for stage cascade on 50Salads (mid).

Threshold	F1@{10,25,50}			Edit	Acc
0.5	81.6	79.7	71.5	74.2	83.0
0.6	82.1	80.9	71.9	74.1	83.3
0.7	81.9	80.5	72.4	74.0	83.9
0.8	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**
0.9	80.6	79.1	69.7	73.3	82.9

Table 4: Study on the the number of Cascade Stages on 50Salads dataset.

50Salads (mid)	F1@{10,25,50}			Edit	Acc
MS-TCN (3 stages)	71.5	68.6	61.1	64.0	78.6
MS-TCN (4 stages)	76.3	74.0	64.5	67.9	80.7
BCN (2 cascade stages)	81.2	79.2	71.3	73.5	83.6
BCN (3 cascade stages)	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**
BCN (4 cascade stages)	80.7	78.9	71.5	72.9	83.5

Comparison between LBP and two heuristic temporal regularizer in Table. 2 illustrates that our proposed boundary-aware and input-dependent temporal regularizer LBP is more effective than its special cases which are unable to utilize boundary information thus unsuitable for temporal action segmentation task.

4.2 Ablation Study on Hyper-parameters

We also investigate the effects of threshold ρ, number of stages in SC and LBP's pooling window size. Our ablation studies strictly follows existing works and all the comparisons are *fair*. Experiments show that both SC and LBP are insensitive to most of the hyper-parameters, which proves our framework's robustness. Please refer to appendix for other less important hyper-parameters.

Study on Probability Thresholds. Based on frame-wise confidence score of the previous cascade, SC can update weights adaptively, where threshold ρ controls the frame distribution in SC: smaller ρ encourages more frames to be handled in early stage while larger ρ tends to progressively classify most of frames by the later stages. In extreme cases, weights in SC will monotonically increase when $\rho = 1$ and decrease when $\rho = 0$. As shown in Table 3, BCN achieves the best performance with $\rho = 0.8$, which lead to the distribution about 26:21:53 for 3 cascade stages of SC. Although the value of ρ is dataset-dependent which can be chosen empirically using a validation set, our experiment shows that ρ has little influence on our methods in Table 3 and our methods achieve good results in GTEA and Breakfast datasets with the $\rho = 0.8$ obtained in 50Salads.

Study on the Number of Cascade Stages. Commonly we use 3 cascade stages and 1 fusion stage in our experiments for *fair* comparison with baseline, yet we construct a 2-stage BCN to justify that our performance gain is not obtained from more computational cost or parameters. SC only updates the weights for stage 2 w_t^2 and keep all w_t^1 to be 1 for 2-stage BCN. Table 4 shows our framework with less capacity not only surpasses its baseline by large margin, but also outperforms MS-TCN's best result, which proves the effectiveness of our method. Comparison between BCN with different stages shows that our method's behavior is similar to the backbone: the performance will not be better if we add more stages, just like the result of Table.1 in [3].

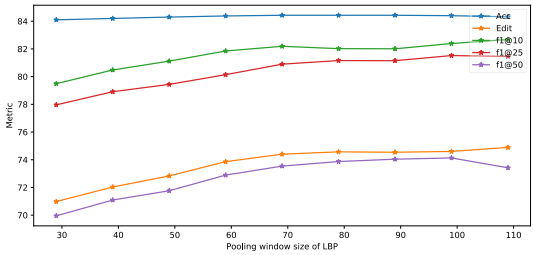


Fig. 5: Metrics as functions of LBP's pooling window size on 50Salads (mid).

Study on LBP's Pooling Window Size. As Fig. 5 shows, larger pooling window size generally benefits all the metrics, which demonstrates that small windows can not provide enough confident temporal neighborhood to correct misclassified frames and a large window is necessary for smoothing predictions due to the existence of long duration actions. We also observe that the performance drops a little when we stretch the window too large. This may be because pooling weights are lowered rather than being zeros across barriers in LBP and then there might be unexpected long-term interactions between action instances.

4.3 Comparison with the State of the Art

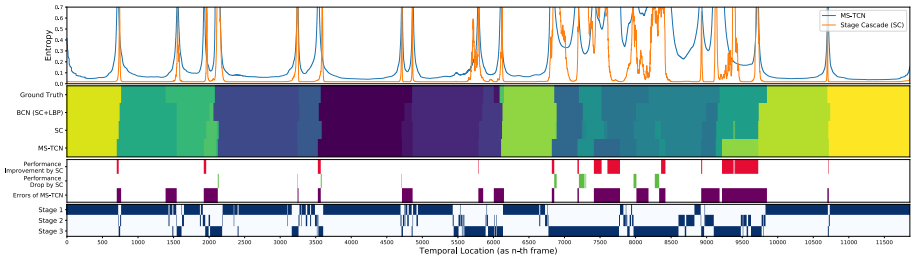
In Table 5, we compare our BCN to the state-of-the-art methods on three challenging benchmarks: 50Salads, GTEA and Breakfast datasets with I3D features (without fine-tune). Our model leads to notable gains compared to previous competitive methods in all datasets, especially in the larger Breakfast dataset. Qualitative results on two datasets are shown in Fig. 6 (please refer to appendix for GTEA dataset), visualizing that our predictions have high accuracy and the weight assignment is consist to our analysis of BCN where hard frames near boundaries are handled by later stages with higher capacity. Our model's frame-wise confidence score's entropy is far below the baseline's, indicating more confident predictions. It's worth noting that there is only a minor extra computational burden in our models compared to the baseline.

Table 5: Comparison with the state-of-the-art on 50Salads, GTEA and Breakfast dataset. (* uses multi-modal data, † obtained from [2])

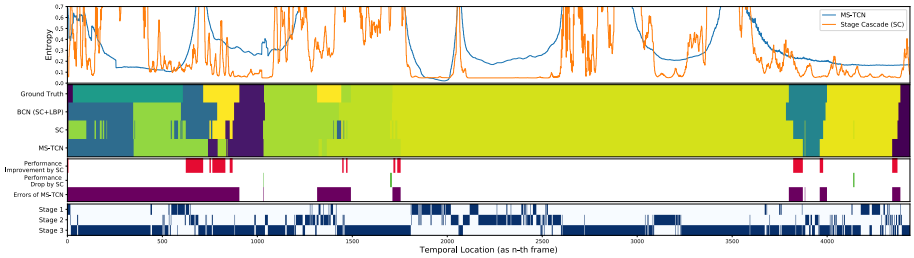
50Salads (mid)	F1@{10,25,50}			Edit	Acc
Spatial CNN [14]	32.3	27.1	18.9	24.8	54.9
IDT+LM [22]	44.4	38.9	27.8	45.8	48.7
Dilated TCN [13]	52.2	47.6	37.4	43.1	59.3
ST-CNN [14]	55.9	49.6	37.1	45.9	59.4
Bi-LSTM [25]	62.6	58.3	47.0	55.6	55.7
ED-TCN [13]	68.0	63.9	52.6	59.8	64.7
TDRN [16]	72.9	68.5	57.2	66.0	68.1
MS-TCN [3]	76.3	74.0	64.5	67.9	80.7
Coupled GAN [6]*	80.1	78.7	71.1	76.9	74.5
BCN	**82.3**	**81.3**	**74.0**	**74.3**	**84.4**

GTEA	F1@{10,25,50}			Edit	Acc
Bi-LSTM [25]	66.5	59.0	43.6	-	55.5
ED-TCN [13]	72.2	69.3	56.0	-	64.0
TDRN [16]	79.2	74.4	62.7	74.1	70.1
MS-TCN [3]	85.8	83.4	69.8	79.0	76.3
Coupled GAN [6]*	80.1	77.9	69.1	72.8	78.5
BCN	**88.5**	**87.1**	**77.3**	**84.4**	**79.8**

Breakfast	F1@{10,25,50}			Edit	Acc
ED-TCN [13]†	-	-	-	-	43.3
TCFPN [2]	-	-	-	-	52.0
HTK (64) [12]	-	-	-	-	56.3
GRU [23]†	-	-	-	-	60.6
MS-TCN (I3D) [3]	52.6	48.1	37.9	61.7	66.3
BCN	**68.7**	**65.5**	**55.0**	**66.2**	**70.4**



(a) rgb-27-1 from 50Salads dataset.



(b) P13_cam01_P13_pancake from Breakfast dataset.

Fig. 6: Qualitative results on 50Salads and Breakfast datasets. There are several rows: (1) Frame-wise entropy of SC vs. baseline [3]; (2) Action segmentation results of groundtruth, BCN, SC and baseline; (3) SC's improvement (red) and performance drop (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.

5 Conclusion

We have presented a new framework called boundary-aware cascade network (BCN) for temporal action segmentation, which consists of two components: a stage cascade module that adaptively adjusts weights to enable later stages to focus on harder frames, and a local barrier pooling to improve the smoothness of predictions by explicitly utilizing semantic boundary information. Our empirical evaluation on the benchmark 50Salads, GTEA and Breakfast demonstrated that BCN outperforms the state-of-the-art models by a large margin. The superior performance of BCN is owe to the fact that it is able to predict more precise action segments and greatly reduce over-segmentation artifacts. It implies merits of our end-to-end learning of stage cascade and local barrier pooling over simply stacking deeper layers of temporal convolutions.

Acknowledgements. This work is supported by Tencent AI Lab Rhino-Bird Focused Research Program (No. JR202025), the National Science Foundation of China (No. 61921006), Program for Innovative Talents and Entrepreneur in Jiangsu Province, and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

1. Carreira, J., Zisserman, A.: Quo vadis, action recognition? A new model and the kinetics dataset. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 4724–4733 (2017)
2. Ding, L., Xu, C.: Weakly-supervised action segmentation with iterative soft boundary assignment. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. pp. 6508–6516 (2018)
3. Farha, Y.A., Gall, J.: MS-TCN: multi-stage temporal convolutional network for action segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019. pp. 3575–3584 (2019)
4. Fathi, A., Rehg, J.M.: Modeling actions through state changes. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013. pp. 2579–2586 (2013)
5. Fathi, A., Ren, X., Rehg, J.M.: Learning to recognize objects in egocentric activities. In: The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011. pp. 3281–3288 (2011)
6. Gammulle, H., Fernando, T., Denman, S., Sridharan, S., Fookes, C.: Coupled generative adversarial network for continuous fine-grained action segmentation. In: IEEE Winter Conference on Applications of Computer Vision, WACV 2019, Waikoloa Village, HI, USA, January 7-11, 2019. pp. 200–209 (2019)
7. Gao, J., Yang, Z., Nevatia, R.: Cascaded boundary regression for temporal action detection. In: British Machine Vision Conference 2017, BMVC 2017, London, UK, September 4-7, 2017 (2017)
8. Gao, Z., Wang, L., Wu, G.: LIP: local importance-based pooling. In: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019. pp. 3354–3363 (2019)
9. Huang, D., Fei-Fei, L., Niebles, J.C.: Connectionist temporal modeling for weakly supervised action labeling. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV. pp. 137–153 (2016)
10. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 2261–2269. IEEE Computer Society (2017)
11. Kuehne, H., Arslan, A.B., Serre, T.: The language of actions: Recovering the syntax and semantics of goal-directed human activities. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014. pp. 780–787 (2014)
12. Kuehne, H., Gall, J., Serre, T.: An end-to-end generative framework for video segmentation and recognition. In: 2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016, Lake Placid, NY, USA, March 7-10, 2016. pp. 1–8 (2016)
13. Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks for action segmentation and detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 1003–1012 (2017)
14. Lea, C., Reiter, A., Vidal, R., Hager, G.D.: Segmental spatiotemporal cnns for fine-grained action segmentation. In: Computer Vision - ECCV 2016 - 14th European

Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III. pp. 36–52 (2016)
15. Lea, C., Vidal, R., Hager, G.D.: Learning convolutional action primitives for fine-grained action recognition. In: 2016 IEEE International Conference on Robotics and Automation, ICRA 2016, Stockholm, Sweden, May 16-21, 2016. pp. 1642–1649 (2016)
16. Lei, P., Todorovic, S.: Temporal deformable residual networks for action segmentation in videos. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. pp. 6742–6751 (2018)
17. Li, H., Lin, Z., Shen, X., Brandt, J., Hua, G.: A convolutional neural network cascade for face detection. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. pp. 5325–5334 (2015)
18. Li, X., Liu, Z., Luo, P., Loy, C.C., Tang, X.: Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 6459–6468 (2017)
19. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: BSN: boundary sensitive network for temporal action proposal generation. In: Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part IV. pp. 3–21 (2018)
20. van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A.W., Kavukcuoglu, K.: Wavenet: A generative model for raw audio. In: The 9th ISCA Speech Synthesis Workshop, Sunnyvale, CA, USA, 13-15 September 2016. p. 125 (2016)
21. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada. pp. 91–99 (2015)
22. Richard, A., Gall, J.: Temporal action detection using a statistical language model. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 3131–3140 (2016)
23. Richard, A., Kuehne, H., Gall, J.: Weakly supervised action learning with RNN based fine-to-coarse modeling. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017. pp. 1273–1282 (2017)
24. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 27, pp. 568–576. Curran Associates, Inc. (2014)
25. Singh, B., Marks, T.K., Jones, M.J., Tuzel, O., Shao, M.: A multi-stream bi-directional recurrent neural network for fine-grained action detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 1961–1970 (2016)
26. Stein, S., McKenna, S.J.: Combining embedded accelerometers with computer vision for recognizing food preparation activities. In: The 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing, UbiComp '13, Zurich, Switzerland, September 8-12, 2013. pp. 729–738 (2013)

27. Toshev, A., Szegedy, C.: Deeppose: Human pose estimation via deep neural networks. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014. pp. 1653–1660 (2014)
28. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). pp. 4489–4497. ICCV '15 (2015)
29. Wang, L., Li, W., Li, W., Gool, L.V.: Appearance-and-relation networks for video classification. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018. pp. 1430–1439. IEEE Computer Society (2018)
30. Wang, L., Qiao, Y., Tang, X.: Action recognition with trajectory-pooled deep-convolutional descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. pp. 4305–4314. IEEE Computer Society (2015)
31. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L.V.: Temporal segment networks: Towards good practices for deep action recognition. In: Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII. pp. 20–36 (2016)
32. Yeung, S., Russakovsky, O., Mori, G., Fei-Fei, L.: End-to-end learning of action detection from frame glimpses in videos. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016. pp. 2678–2687 (2016)
33. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017. pp. 2933–2942 (2017)