Boundary Sensitive Network: Submission to ActivityNet Challenge 2018

Tianwei Lin, Haisheng Su, Xu Zhao*

Department of Automation,
Shanghai Jiao Tong University
{wzmsltw,suhaisheng,zhaoxu}@sjtu.edu.cn

Abstract. In this technical paper, we describe our approach used in the submission to the **temporal action proposal generation (task 1)** and **temporal action localization (detection) (task 2)** of ActivityNet Challenge 2018. Since we believe that the main bottleneck for temporal action localization is the quality of action proposals, we mainly focus on the temporal action proposal generation task and adopt a novel proposal generation method we proposed recently, called Boundary-Sensitive Network (BSN) [1]. To generate high quality proposals, BSN first locates temporal boundaries with high probabilities, then directly combines these boundaries as proposals. Finally, with Boundary-Sensitive Proposal feature, BSN retrieves proposals by evaluating the confidence of whether a proposal contains an action within its region. BSN achieves the state-of-the-art performances on both temporal action proposal generation task and temporal action localization task. The full version of our paper can be found in [1].

Keywords: Temporal action proposal generation · Temporal action detection · Temporal convolution · Untrimmed video

1 Introduction

Nowadays, with fast development of digital cameras and Internet, the number of videos is continuously booming, making automatic video content analysis methods widely required. One major branch of video analysis is action recognition, which aims to classify manually trimmed video clips containing only one action instance. However, videos in real scenarios are usually long, untrimmed and contain multiple action instances along with irrelevant contents. This problem requires algorithms for another challenging task: temporal action detection, which aims to detect action instances in untrimmed video including both temporal boundaries and action classes. It can be applied in many areas such as video recommendation and smart surveillance.

Similar with object detection in spatial domain, temporal action detection task can be divided into two stages: proposal and classification. Proposal generation stage aims to generate temporal video regions which may contain action instances, and classification stage aims to classify classes of candidate proposals. Although classification methods have reached convincing performance, the detection precision is still low in

* Corresponding author.

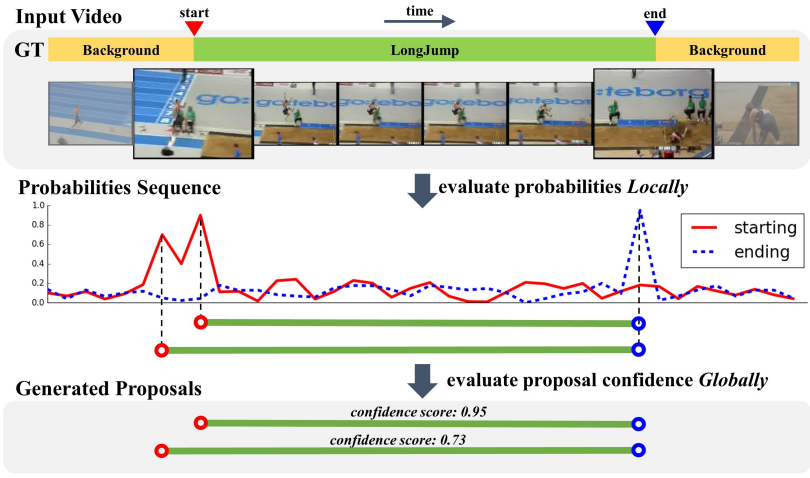


Fig. 1: Overview of our approach. Given an untrimmed video, (1) we evaluate boundaries and actionness probabilities of each temporal location and generate proposals based on boundary probabilities, and (2) we evaluate the confidence scores of proposals with proposal-level feature to get retrieved proposals.

many benchmarks [2, 3]. Thus recently temporal action proposal generation has received much attention [4–7], aiming to improve the detection performance by improving the quality of proposals. High quality proposals should come up with two key properties: (1) proposals can cover truth action regions with both high recall and high temporal overlap, (2) proposals are retrieved so that high recall and high overlap can be achieved using fewer proposals to reduce the computation cost of succeeding steps.

To achieve high proposal quality, a proposal generation method should generate proposals with flexible temporal durations and precise temporal boundaries, then retrieve proposals with reliable confidence scores, which indicate the probability of a proposal containing an action instance. Most recently proposal generation methods [4–6, 8] generate proposals via sliding temporal windows of multiple durations in video with regular interval, then train a model to evaluate the confidence scores of generated proposals for proposals retrieving, while there is also method [7] making external boundaries regression. However, proposals generated with pre-defined durations and intervals may have some major drawbacks: (1) usually not temporally precise; (2) not flexible enough to cover variable temporal durations of ground truth action instances, especially when the range of temporal durations is large.

To address these issues and generate high quality proposals, we propose the Boundary-Sensitive Network (BSN), which adopts "*local to global*" fashion to locally combine high probability boundaries as proposals and globally retrieve candidate proposals using proposal-level feature as shown in Fig 1. In detail, BSN generates proposals in three steps. **First**, BSN evaluates the probabilities of each temporal location in video whether it is inside or outside, at or not at the boundaries of ground truth action instances, to generate starting, ending and actionness probabilities sequences as local information. **Second**, BSN generates proposals via directly combining temporal locations with high

starting and ending probabilities separately. Using this bottom-up fashion, BSN can generate proposals with flexible durations and precise boundaries. **Finally**, using features composed by actionness scores within and around proposal, BSN retrieves proposals by evaluating the confidence of whether a proposal contains an action. These proposal-level features offer global information for better evaluation.

In summary, the main contributions of our work are three-folds:

(1) We introduce a new architecture (BSN) based on "*local to global*" fashion to generate high quality temporal action proposals, which *locally* locates high boundary probability locations to achieve precise proposal boundaries and *globally* evaluates proposal-level feature to achieve reliable proposal confidence scores for retrieving.

(2) Extensive experiments demonstrate that our method achieves significantly better proposal quality than other state-of-the-art proposal generation methods, and can generate proposals in unseen action classes with comparative quality.

(3) Integrating our method with existing action classifier into detection framework leads to significantly improved performance on temporal action detection task.

2 Related work

Action recognition. Action recognition is an important branch of video related research areas and has been extensively studied. Earlier methods such as improved Dense Trajectory (iDT) [9, 10] mainly adopt hand-crafted features such as HOF, HOG and MBH. In recent years, convolutional networks are widely adopted in many works [11–14] and have achieved great performance. Typically, two-stream network [11, 12, 14] learns appearance and motion features based on RGB frame and optical flow field separately. C3D network [13] adopts 3D convolutional layers to directly capture both appearance and motion features from raw frames volume. Action recognition models can be used for extracting frame or snippet level visual features in long and untrimmed videos.

Object detection and proposals. Recent years, the performance of object detection has been significantly improved with deep learning methods. R-CNN [15] and its variations [16, 17] construct an important branch of object detection methods, which adopt "detection by classifying proposals" framework. For proposal generation stage, besides sliding windows [18], earlier works also attempt to generate proposals by exploiting low-level cues such as HOG and Canny edge [19, 20]. Recently some methods [17, 21, 22] adopt deep learning model to generate proposals with faster speed and stronger modelling capacity. In this work, we combine the properties of these methods via evaluating boundaries and actionness probabilities of each location using neural network and adopting "*local to global*" fashion to generate proposals with high recall and accuracy.

Boundary probabilities are also adopted in LocNet [23] for revising the horizontal and vertical boundaries of existing proposals. Our method differs in (1) BSN aims to generate while LocNet aims to revise proposals and (2) boundary probabilities are calculated repeatedly for all boxes in LocNet but only once for a video in BSN.

Temporal action detection and proposals. Temporal action detection task aims to detect action instances in untrimmed videos including temporal boundaries and action classes, and can be divided into proposal and classification stages. Most detection methods [8, 24, 25] take these two stages separately, while there is also method [26, 27] tak-

ing these two stages jointly. For proposal generation, earlier works [28–30] directly use sliding windows as proposals. Recently some methods [4–8] generate proposals with pre-defined temporal durations and intervals, and use multiple methods to evaluate the confidence score of proposals, such as dictionary learning [5] and recurrent neural network [6]. TAG method [25] adopts watershed algorithm to generate proposals with flexible boundaries and durations in *local* fashion, but without *global* proposal-level confidence evaluation for retrieving. In our work, BSN can generate proposals with flexible boundaries meanwhile reliable confidence scores for retrieving.

Recently temporal action detection method [31] detects action instances based on class-wise start, middle and end probabilities of each location. Our method is superior than [31] in two aspects: (1) BSN evaluates probabilities score using temporal convolution to better capture temporal information and (2) "*local to global*" fashion adopted in BSN brings more precise boundaries and better retrieving quality.

3 Our Approach

3.1 Problem Definition

An untrimmed video sequence can be denoted as $X = \{x_n\}_{n=1}^{l_v}$ with l_v frames, where x_n is the n-th frame in X. Annotation of video X is composed by a set of action instances $\Psi_g = \{\varphi_n = (t_{s,n}, t_{e,n})\}_{n=1}^{N_g}$, where N_g is the number of truth action instances in video X, and $t_{s,n}$, $t_{e,n}$ are starting and ending time of action instance φ_n separately. Unlike detection task, classes of action instances are not considered in temporal action proposal generation. Annotation set Ψ_g is used during training. During prediction, generated proposals set Ψ_p should cover Ψ_g with high recall and high temporal overlap.

3.2 Video Features Encoding

To generate proposals of input video, first we need to extract feature to encode visual content of video. In our framework, we adopt two-stream network [12] as visual encoder, since this architecture has shown great performance in action recognition task [32] and has been widely adopted in temporal action detection and proposal generation tasks [25, 26, 33]. Two-stream network contains two branches: spatial network operates on single RGB frame to capture appearance feature, and temporal network operates on stacked optical flow field to capture motion information.

To extract two-stream features, as shown in Fig 2(a), first we compose a snippets sequence $S = \{s_n\}_{n=1}^{l_s}$ from video X, where l_s is the length of snippets sequence. A snippet $s_n = (x_{t_n}, o_{t_n})$ includes two parts: x_{t_n} is the t_n-th RGB frame in X and o_{t_n} is stacked optical flow field derived around center frame x_{t_n}. To reduce the computation cost, we extract snippets with a regular frame interval σ, therefore $l_s = l_v/\sigma$. Given a snippet s_n, we concatenate output scores in top layer of both spatial and temporal networks to form the encoded feature vector $f_{t_n} = (f_{S,t_n}, f_{T,t_n})$, where f_{S,t_n}, f_{T,t_n} are output scores from spatial and temporal networks separately. Thus given a snippets sequence S with length l_s, we can extract a feature sequence $F = \{f_{t_n}\}_{n=1}^{l_s}$. These two-stream feature sequences are used as the input of BSN.

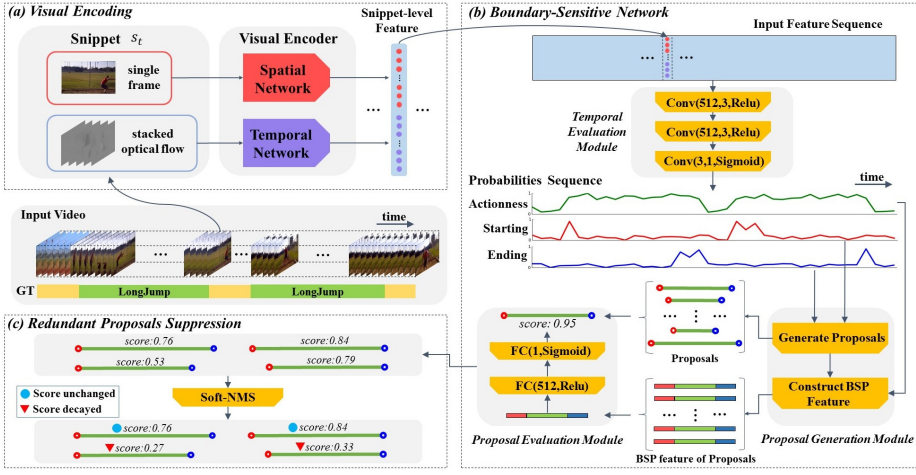


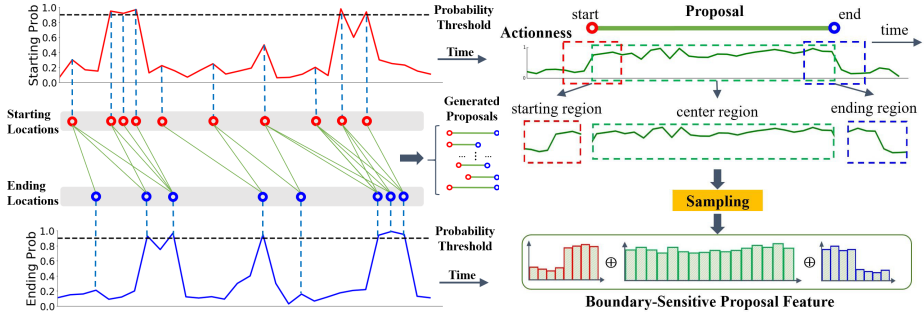
Fig. 2: The framework of our approach. (a) Two-stream network is used for encoding visual features in snippet-level. (b) The architecture of Boundary-Sensitive Network: *temporal evaluation module* handles the input feature sequence, and evaluates starting, ending and actionness probabilities of each temporal location; *proposal generation module* generates proposals with high starting and ending probabilities, and construct Boundary-Sensitive Proposal (BSP) feature for each proposal; *proposal evaluation module* evaluates confidence score of each proposal using BSP feature. (c) Finally, we use Soft-NMS algorithm to suppress redundant proposals by decaying their scores.

3.3 Boundary-Sensitive Network

To achieve high proposal quality with both precise temporal boundaries and reliable confidence scores, we adopt "*local to global*" fashion to generate proposals. In BSN, we first generate candidate boundary locations, then combine these locations as proposals and evaluate confidence score of each proposal with proposal-level feature.

Network architecture. The architecture of BSN is presented in Fig 2(b), which contains three modules: temporal evaluation, proposal generation and proposal evaluation. *Temporal evaluation module* is a three layers temporal convolutional neural network, which takes the two-stream feature sequences as input, and evaluates probabilities of each temporal location in video whether it is inside or outside, at or not at boundaries of ground truth action instances, to generate sequences of starting, ending and actionness probabilities respectively. *Proposal generation module* first combines the temporal locations with separately high starting and ending probabilities as candidate proposals, then constructs Boundary-Sensitive Proposal (BSP) feature for each candidate proposal based on actionness probabilities sequence. Finally, *proposal evaluation module*, a multilayer perceptron model with one hidden layer, evaluates the confidence score of each candidate proposal based on BSP feature. Confidence score and boundary probabilities of each proposal are fused as the final confidence score for retrieving.

Temporal evaluation module. The goal of temporal evaluation module is to evaluate starting, ending and actionness probabilities of each temporal location, where three binary classifiers are needed. In this module, we adopt temporal convolutional layers upon



(a) Generate proposals (b) Construct BSP feature

Fig. 3: Details of proposal generation module. (a) Generate proposals. First, to generate candidate boundary locations, we choose temporal locations with high boundary probability or being a probability peak. Then, we combine candidate starting and ending locations as proposals when their duration satisfying condition. (b) Construct BSP feature. Given a proposal and actionness probabilities sequence, we can sample actionness sequence in starting, center and ending regions of proposal to construct BSP feature.

feature sequence, with good modelling capacity to capture local semantic information such as boundaries and actionness probabilities.

A temporal convolutional layer can be simply denoted as $Conv(c_f, c_k, Act)$, where c_f, c_k and Act are filter numbers, kernel size and activation function of temporal convolutional layer separately. As shown in Fig 2(b), the temporal evaluation module can be defined as $Conv(512, 3, Relu) \rightarrow Conv(512, 3, Relu) \rightarrow Conv(3, 1, Sigmoid)$, where the three layers have same stride size 1. Three filters with sigmoid activation in the last layer are used as classifiers to generate starting, ending and actionness probabilities separately. For convenience of computation, we divide feature sequence into non-overlapped windows as the input of temporal evaluation module. Given a feature sequence F, temporal evaluation module can generate three probability sequences $P_S = \{p^s_{t_n}\}_{n=1}^{l_s}$, $P_E = \{p^e_{t_n}\}_{n=1}^{l_s}$ and $P_A = \{p^a_{t_n}\}_{n=1}^{l_s}$, where $p^s_{t_n}$, $p^e_{t_n}$ and $p^a_{t_n}$ are respectively starting, ending and actionness probabilities in time t_n.

Proposal generation module. The goal of proposal generation module is to generate candidate proposals and construct corresponding proposal-level feature. We achieve this goal in two steps. First we locate temporal locations with high boundary probabilities, and combine these locations to form proposals. Then for each proposal, we construct Boundary-Sensitive Proposal (BSP) feature.

As shown in Fig 3(a), to locate where an action likely to start, for starting probabilities sequence P_S, we record all temporal location t_n where $p^s_{t_n}$ (1) has high score: $p^s_{t_n} > 0.9$ or (2) is a probability peak: $p^s_{t_n} > p^s_{t_{n-1}}$ and $p^s_{t_n} > p^s_{t_{n+1}}$. These locations are grouped into candidate starting locations set $B_S = \{t_{s,i}\}_{i=1}^{N_S}$, where N_S is the number of candidate starting locations. Using same rules, we can generate candidate ending locations set B_E from ending probabilities sequence P_E. Then, we generate temporal regions via combing each starting location t_s from B_S and each ending location t_e from B_S. Any temporal region $[t_s, t_e]$ satisfying $d = t_e - t_s \in [d_{min}, d_{max}]$ is denoted as a candidate proposal φ, where d_{min} and d_{max} are minimum and maximum durations

of ground truth action instances in dataset. Thus we can get candidate proposals set $\Psi_p = \{\varphi_i\}_{i=1}^{N_p}$, where N_p is the number of proposals.

To construct proposal-level feature as shown in Fig 3(b), for a candidate proposal φ, we denote its center region as $r_C = [t_s, t_e]$ and its starting and ending region as $r_S = [t_s - d/5, t_s + d/5]$ and $r_E = [t_e - d/5, t_e + d/5]$ separately. Then, we sample the actionness sequence P_A within r_c as f_c^A by linear interpolation with 16 points. In starting and ending regions, we also sample actionness sequence with 8 linear interpolation points and get f_s^A and f_e^A separately. Concatenating these vectors, we can get Boundary-Sensitive Proposal (BSP) feature $f_{BSP} = (f_s^A, f_c^A, f_e^A)$ of proposal φ. BSP feature is highly compact and contains rich semantic information about corresponding proposal. Then we can represent a proposal as $\varphi = (t_s, t_e, f_{BSP})$.

Proposal evaluation module. The goal of proposal evaluation module is to evaluate the confidence score of each proposal whether it contains an action instance within its duration using BSP feature. We adopt a simple multilayer perceptron model with one hidden layer as shown in Fig 2(b). Hidden layer with 512 units handles the input of BSP feature f_{BSP} with Relu activation. The output layer outputs confidence score p_{conf} with sigmoid activation, which estimates the overlap extent between candidate proposal and ground truth action instances. Thus, a generated proposal can be denoted as $\varphi = (t_s, t_e, p_{conf}, p_{t_s}^s, p_{t_e}^e)$, where $p_{t_s}^s$ and $p_{t_e}^e$ are starting and ending probabilities in t_s and t_e separately. These scores are fused to generate final score during prediction.

3.4 Training of BSN

In BSN, temporal evaluation module is trained to learn local boundary and actionness probabilities from video features simultaneously. Then based on probabilities sequence generated by trained temporal evaluation module, we can generate proposals and corresponding BSP features and train the proposal evaluation module to learn the confidence score of proposals. The training details are introduced in this section.

Temporal evaluation module. Given a video X, we compose a snippets sequence S with length l_s and extract feature sequence F from it. Then we slide windows with length $l_w = 100$ in feature sequence without overlap. A window is denoted as $\omega = \{F_\omega, \Psi_\omega\}$, where F_ω and Ψ_ω are feature sequence and annotations within the window separately. For ground truth action instance $\varphi_g = (t_s, t_e)$ in Ψ_ω, we denote its region as action region r_g^a and its starting and ending region as $r_g^s = [t_s - d_g/10, t_s + d_g/10]$ and $r_g^e = [t_e - d_g/10, t_e + d_g/10]$ separately, where $d_g = t_e - t_s$.

Taking F_ω as input, temporal evaluation module generates probabilities sequence $P_{S,\omega}$, $P_{E,\omega}$ and $P_{A,\omega}$ with same length l_w. For each temporal location t_n within F_ω, we denote its region as $r_{t_n} = [t_n - d_s/2, t_n + d_s/2]$ and get corresponding probability scores $p_{t_n}^s$, $p_{t_n}^e$ and $p_{t_n}^a$ from $P_{S,\omega}$, $P_{E,\omega}$ and $P_{A,\omega}$ separately, 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 r_g^a, r_g^s and r_g^e of all φ_g in Ψ_ω separately, where IoP is defined as the overlap ratio with groundtruth proportional to the duration of this proposal. Thus we can represent information of t_n as $\phi_n = (p_{t_n}^a, p_{t_n}^s, p_{t_n}^e, g_{t_n}^a, g_{t_n}^s, g_{t_n}^e)$, where $g_{t_n}^a$, $g_{t_n}^s$, $g_{t_n}^e$ are maximum matching overlap IoP of action, starting and ending regions separately.

Given a window of matching information as $\Phi_\omega = \{\phi_n\}_{n=1}^{l_s}$, we can define training objective of this module as a three-task loss function. The overall loss function consists of actionness loss, starting loss and ending loss:

$$L_{TEM} = \lambda \cdot L_{bl}^{action} + L_{bl}^{start} + L_{bl}^{end}, \tag{1}$$

where λ is the weight term and is set to 2 in BSN. We adopt the sum of binary logistic regression loss function L_{bl} for all three tasks, which can be denoted as:

$$L_{bl} = \frac{1}{l_w} \sum_{i=1}^{l_w} \left(\alpha^+ \cdot b_i \cdot log(p_i) + \alpha^- \cdot (1 - b_i) \cdot log(1 - p_i)\right), \tag{2}$$

where $b_i = sign(g_i - \theta_{IoP})$ is a two-values function for converting matching score g_i to $\{0, 1\}$ based on threshold θ_{IoP}, which is set to 0.5 in BSN. Let $l^+ = \sum g_i$ and $l^- = l_w - l^+$, we can set $\alpha^+ = \frac{l_w}{l^-}$ and $\alpha^- = \frac{l_w}{l^+}$, which are used for balancing the effect of positive and negative samples during training.

Proposal evaluation module. Using probabilities sequences generated by trained temporal evaluation module, we can generate proposals using proposal generation module: $\Psi_p = \{\varphi_n = (t_s, t_e, f_{BSP})\}_{n=1}^{N_p}$. Taking f_{BSP} as input, for a proposal φ, confidence score p_{conf} is generated by proposal evaluation module. Then we calculate its Intersection-over-Union (IoU) with all φ_g in Ψ_g, and denote the maximum overlap score as g_{iou}. Thus we can represent proposals set as $\Psi_p = \{\varphi_n = \{t_s, t_e, p_{conf}, g_{iou}\}\}_{n=1}^{N_p}$. We split Ψ_p into two parts based on g_{iou}: Ψ_p^{pos} for $g_{iou} > 0.7$ and Ψ_p^{neg} for $g_{iou} < 0.3$. For data balancing, we take all proposals in Ψ_p^{pos} and randomly sample the proposals in Ψ_p^{neg} to insure the ratio between two sets be nearly 1:2.

The training objective of this module is a simple regression loss, which is used to train a precise confidence score prediction based on IoU overlap. We can define it as:

$$L_{PEM} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} (p_{conf,i} - g_{iou,i})^2, \tag{3}$$

where N_{train} is the number of proposals used for training.

3.5 Prediction and Post-processing

During prediction, we use BSN with same procedures described in training to generate proposals set $\Psi_p = \left\{\varphi_n = (t_s, t_e, p_{conf}, p_{t_s}^s, p_{t_e}^e)\right\}_{n=1}^{N_p}$, where N_p is the number of proposals. To get final proposals set, we need to make score fusion to get final confidence score, then suppress redundant proposals based on these score.

Score fusion for retrieving. To achieve better retrieving performance, for each candidate proposal φ, we fuse its confidence score with its boundary probabilities by multiplication to get the final confidence score p_f:

$$p_f = p_{conf} \cdot p_{t_s}^s \cdot p_{t_e}^e. \tag{4}$$

After score fusion, we can get generated proposals set $\Psi_p = \{\varphi_n = (t_s, t_e, p_f)\}_{n=1}^{N_p}$, where p_f is used for proposals retrieving. In section 4.2, we explore the recall performance with and without confidence score generated by proposal evaluation module.

Redundant proposals suppression. Around a ground truth action instance, we may generate multiple proposals with different temporal overlap. Thus we need to suppress redundant proposals to obtain higher recall with fewer proposals.

Soft-NMS [34] is a recently proposed non-maximum suppression (NMS) algorithm which suppresses redundant results using a score decaying function. First all proposals are sorted by their scores. Then proposal φ_m with maximum score is used for calculating overlap IoU with other proposals, where scores of highly overlapped proposals is decayed. This step is recursively applied to the remaining proposals to generate re-scored proposals set. The Gaussian decaying function of Soft-NMS can be denoted as:

$$p'_{f,i} = \begin{cases} p_{f,i}, & iou(\varphi_m, \varphi_i) < \theta \\ p_{f,i} \cdot e^{-\frac{iou(\varphi_m, \varphi_i)^2}{\varepsilon}}, & iou(\varphi_m, \varphi_i) \geq \theta \end{cases} \tag{5}$$

where ε is parameter of Gaussian function and θ is pre-fixed threshold. After suppression, we get the final proposals set $\Psi'_p = \left\{\varphi_n = (t_s, t_e, p'_f)\right\}_{n=1}^{N_p}$.

4 Experiments

In the full version of BSN paper [1], we conduct experiments on both ActivityNet-1.3 and THUMOS-14 datasets including many ablation studys, where BSN achieves great performance on both datasets. In this challenge report, we mainly introduce new improvements and experiments of BSN on ActivityNet Challenge 2018. And for convenience, we denote BSN introduced in [1] as BSN-baseline.

4.1 Dataset and setup

Dataset. ActivityNet-1.3 [2] is a large dataset for general temporal action proposal generation and detection, which contains 19994 videos with 200 action classes annotated and was used in the ActivityNet Challenge 2017 and 2018. ActivityNet-1.3 is divided into training, validation and testing sets by ratio of 2:1:1.

Evaluation metrics. In temporal action proposal generation task, Average Recall (AR) calculated with multiple IoU thresholds is usually used as evaluation metrics. Following conventions, we use IoU thresholds set $[0.5 : 0.05 : 0.95]$. To evaluate the relation between recall and proposals number, we evaluate AR with Average Number of proposals (AN) on both datasets, which is denoted as AR@AN. Area under the AR vs. AN curve (AUC) is also used as metrics, where AN varies from 0 to 100.

In temporal action detection task, mean Average Precision (mAP) is used as evaluation metric, where Average Precision (AP) is calculated on each action class respectively. On ActivityNet-1.3, mAP with IoU thresholds $\{0.5, 0.75, 0.95\}$ and average mAP with IoU thresholds set $[0.5 : 0.05 : 0.95]$ are used.

Implementation details. Here, we mainly introduce the implementation details adopted in BSN-baseline, the improvement details will be introduced later. For visual feature encoding, we use the two-stream network [12] with architecture described in [35], where BN-Inception network [36] is used as temporal network and ResNet network [37] is

Table 1: Comparison between our method with other state-of-the-art proposal generation methods on ActivityNet-1.3 in terms of AR@AN and AUC.

Method	AR@10 (val)	AR@100 (val)	AUC (val)	AUC (test)
Uniform Random	29.02	55.71	44.88	-
Zhao et al. [25]	-	63.52	53.02	-
Dai et al. [41]	-	-	59.58	61.56
Yao et al. [42]	-	-	63.12	64.18
Lin et al. [39]	52.50	73.01	64.40	64.80
BSN-baseline [1]	-	74.16	66.17	66.26
+ improvement A	54.78	74.62	66.26	-
+ improvement B	55.23	75.62	67.17	67.34
+ improvement C	55.12	76.10	67.53	67.46
+ improvement D	55.66	76.45	67.88	67.99
+ improvement E	**56.91**	**77.30**	**68.92**	**69.30**

used as spatial network. Two-stream network is implemented using Caffe [38] and pretrained on ActivityNet-1.3 training set. During feature extraction, the interval σ of snippets is set to 16 on ActivityNet-1.3.

Since the duration of videos are limited, we follow [39] to rescale the feature sequence of each video to new length $l_w = 100$ by linear interpolation, and the duration of corresponding annotations to range [0,1]. In BSN, temporal evaluation module and proposal evaluation module are both implemented using Tensorflow [40]. Temporal evaluation module is trained with batch size 16 and learning rate 0.001 for 10 epochs, then 0.0001 for another 10 epochs, and proposal evaluation module is trained with batch size 256 and same learning rate. For Soft-NMS, we set the threshold θ to 0.8 by empirical validation, while ε in Gaussian function is set to 0.75 on both datasets.

4.2 Temporal Proposal Generation

The proposal performance on ActivityNet-1.3 of our method and previous state-of-the-art methods are shown in Table 1. Our method (BSN-baseline) has significantly better performance than previous method, and we further improve BSN in many aspects to achieve better performance. These improvements are introduced in the following, where each improvement is conducted based on the last improvement.

(A) *Threshold in proposal generation module*: In BSN-baseline, while generating proposals, we choose temporal locations as candidate boundary locations where boundary probability is higher than a threshold or is a probability peak. Here, we modify the threshold from 0.9 to $0.5 \cdot p_{max}$, where p_{max} is the maximum boundary probability in the video.

(B) *Video feature*: In BSN-baseline, we adopt two-stream network [35] pretrained on ActivityNet-1.3 for video feature extraction. Here, we further adopt two-stream network [32] and pseudo-3d network [43] pretrained on Kinetics-400 dataset for video feature extraction. To fuse these features, we train temporal evaluation mod-

Table 2: Action detection results on validation and testing set of ActivityNet-1.3 in terms of mAP@$tIoU$ and average mAP, where our proposals are combined with video-level classification results generated by [48].

	validation				testing
Method	0.5	0.75	0.95	Average	Average
Wang et al. [44]	42.28	3.76	0.05	14.85	14.62
SCC [45]	40.00	17.90	4.70	21.70	19.30
CDC [46]	43.83	25.88	0.21	22.77	22.90
TCN [41]	-	-	-	-	23.58
SSN [47]	39.12	23.48	5.49	23.98	28.28
Lin et al. [39]	44.39	29.65	7.09	29.17	32.26
BSN-baseline [1]	46.45	29.96	8.02	30.03	32.84
BSN-improved	**51.48**	**34.12**	**9.37**	**34.15**	**37.18**

ule using these features separately, then average the output of temporal evaluation module trained with different features.

(C) *Ensemble with SSAD-prop* [26, 39]: For better evaluating the confidence of proposals, we ensemble the results of BSN with the results of SSAD-prop [39]. For each proposal φ_{bsn} generated by BSN, we find a proposal φ_{ssad} generated by SSAD-prop which has maximum IoU. Then we fuse the confidence score of φ_{ssad} and φ_{bsn} via $p'_{bsn} = p_{bsn} \cdot p_{ssad}$, where p'_{bsn} is new confidence score of φ_{bsn}.

(D) *Prediction with original video duration*: In BSN-baseline, for convenience, we rescale the feature length to a fix new length $l_w = 100$. However, there are many short action instances, where the ratio between action duration and video duration is even lower than 0.01. To better capture these short action instances, during prediction, we use the original feature sequence instead of rescaled feature sequence.

(E) *Ensemble of original and fix video duration*: During analysis, we found that make prediction using original video length can benefit the recall performance of short action instances, however, can also damage the recall performance of long action instances. Thus, we make a combination between fix-scale and original-scale predictions: for fix-scale predictions, we take all proposals with duration larger than 25 seconds; for original-scale predictions, we take all proposals with duration smaller than 25 seconds. And we conduct Soft-NMS on combined proposals and output final results.

The experiment results of these improvements are shown in Table 1, which suggest that these improvements can bring salient performance promotion. With improved BSN, we finally achieve 69.30 of AUC in testing set, and win the second place of temporal action proposal generation task in ActivityNet Challenge 2018.

4.3 Action Localization with Our Proposals

To conduct temporal action localization, we put BSN proposals into "detection by classifying proposals" temporal action localization framework with state-of-the-art action

classifier, where temporal boundaries of detection results are provided by our proposals. We use top-2 video-level class generated by classification model [48] [1] for all proposals in a video and keep BSN confidence scores of proposals for retrieving. And we use 100 proposals per video during temporal action detection.

In ActivityNet Challenge 2018, comparing with BSN-baseline, we also adopt improvements introduced above but with two differences: (1) first, we use rescaled video feature with $l_w = 64$ during prediction; (2) second, we set θ in Soft-NMS to 0 here. So why we make these adjustments? Since the localization metric (mAP) mainly depends on first several proposals (as discussed in our previous notebook [39]) and the proposal metric (AUC) depends on first 100 proposals, improvements or configurations which benefit proposal performance may harm localization performance. Thus, as in our previous notebook [39], we suggest that AR with small proposals amount should has higher weight in evaluation metric of proposal generation.

Experiment results shown in Table 2 suggest that our proposed method (BSN-baseline) has significantly better performance than previous state-of-the-art methods, and our new improvements can bring further performance promotion. With improved BSN, we finally achieve 38.52% of mAP in testing set, and win the first place of temporal action localization task in ActivityNet Challenge 2018.

5 Conclusion

In this challenge notebook, we have introduced our recent work: the Boundary-Sensitive Network (BSN) for temporal action proposal generation. Our method can generate proposals with flexible durations and precise boundaries via directly combing locations with high boundary probabilities, and make accurate retrieving via evaluating proposal confidence score with proposal-level features. Thus BSN can achieve high recall and high temporal overlap with relatively few proposals. And we also introduce the improvements we conducted during ActivityNet Challenge 2018, these improvements bring further performance promotion, and can also reveal the direction of how to make better temporal action proposal generation and localization.

[1] Previously, we adopted classification results from result files of [44]. Recently we found that the classification accuracy of these results are unexpected high. Thus we replace it with classification results of [48] and updated all related experiments accordingly.

References

1. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: Bsn: Boundary sensitive network for temporal action proposal generation. arXiv preprint arXiv:1806.02964 (2018)
2. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 961–970
3. Jiang, Y.G., Liu, J., Zamir, A.R., Toderici, G., Laptev, I., Shah, M., Sukthankar, R.: Thumos challenge: Action recognition with a large number of classes. In: ECCV Workshop. (2014)
4. Buch, S., Escorcia, V., Shen, C., Ghanem, B., Niebles, J.C.: SST: Single-stream temporal action proposals. In: IEEE International Conference on Computer Vision. (2017)
5. Caba Heilbron, F., Carlos Niebles, J., Ghanem, B.: Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1914–1923
6. Escorcia, V., Heilbron, F.C., Niebles, J.C., Ghanem, B.: Daps: Deep action proposals for action understanding. In: European Conference on Computer Vision, Springer (2016) 768–784
7. Gao, J., Yang, Z., Sun, C., Chen, K., Nevatia, R.: Turn tap: Temporal unit regression network for temporal action proposals. arXiv preprint arXiv:1703.06189 (2017)
8. Shou, Z., Wang, D., Chang, S.F.: Temporal action localization in untrimmed videos via multi-stage cnns. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1049–1058
9. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Action recognition by dense trajectories. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE (2011) 3169–3176
10. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Proceedings of the IEEE International Conference on Computer Vision. (2013) 3551–3558
11. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1933–1941
12. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in Neural Information Processing Systems. (2014) 568–576
13. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 4489–4497
14. Wang, L., Xiong, Y., Wang, Z., Qiao, Y.: Towards good practices for very deep two-stream convnets. arXiv preprint arXiv:1507.02159 (2015)
15. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2014) 580–587
16. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448
17. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. (2015) 91–99
18. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE transactions on pattern analysis and machine intelligence **32**(9) (2010) 1627–1645
19. Uijlings, J.R., van de Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. International journal of computer vision **104**(2) (2013) 154–171

20. Zitnick, C.L., Dollár, P.: Edge boxes: Locating object proposals from edges. In: European Conference on Computer Vision, Springer (2014) 391–405
21. Kuo, W., Hariharan, B., Malik, J.: Deepbox: Learning objectness with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2479–2487
22. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. arXiv preprint arXiv:1612.03144 (2016)
23. Gidaris, S., Komodakis, N.: Locnet: Improving localization accuracy for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 789–798
24. Singh, G., Cuzzolin, F.: Untrimmed video classification for activity detection: submission to activitynet challenge. arXiv preprint arXiv:1607.01979 (2016)
25. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Lin, D., Tang, X.: Temporal action detection with structured segment networks. arXiv preprint arXiv:1704.06228 (2017)
26. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. In: Proceedings of the 25nd ACM international conference on Multimedia. (2017)
27. Buch, S., Escorcia, V., Ghanem, B., Fei-Fei, L., Niebles, J.C.: End-to-end, single-stream temporal action detection in untrimmed videos. In: Proceedings of the British Machine Vision Conference. (2017)
28. Karaman, S., Seidenari, L., Del Bimbo, A.: Fast saliency based pooling of fisher encoded dense trajectories. In: ECCV THUMOS Workshop. (2014)
29. Oneata, D., Verbeek, J., Schmid, C.: The lear submission at thumos 2014. ECCV THUMOS Workshop (2014)
30. Wang, L., Qiao, Y., Tang, X.: Action recognition and detection by combining motion and appearance features. THUMOS14 Action Recognition Challenge **1** (2014) 2
31. Yuan, Z., Stroud, J.C., Lu, T., Deng, J.: Temporal action localization by structured maximal sums. arXiv preprint arXiv:1704.04671 (2017)
32. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: towards good practices for deep action recognition. In: European Conference on Computer Vision, Springer (2016) 20–36
33. Gao, J., Yang, Z., Nevatia, R.: Cascaded boundary regression for temporal action detection. arXiv preprint arXiv:1705.01180 (2017)
34. Bodla, N., Singh, B., Chellappa, R., Davis, .L.S.: Improving object detection with one line of code. arXiv preprint arXiv:1704.04503 (2017)
35. Xiong, Y., Wang, L., Wang, Z., Zhang, B., Song, H., Li, W., Lin, D., Qiao, Y., Gool, L.V., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2016. arXiv preprint arXiv:1608.00797 (2016)
36. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning. (2015) 448–456
37. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
38. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia, ACM (2014) 675–678
39. Lin, T., Zhao, X., Shou, Z.: Temporal convolution based action proposal: Submission to activitynet 2017. arXiv preprint arXiv:1707.06750 (2017)
40. Abadi, M., Agarwal, A., Barham, P., et al.: Tensorflow: Large-scale machine learning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467 (2016)

41. Dai, X., Singh, B., Zhang, G., Davis, L.S., Chen, Y.Q.: Temporal context network for activity localization in videos. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE (2017) 5727–5736
42. Ghanem, B., Niebles, J.C., Snoek, C., Heilbron, F.C., Alwassel, H., Khrisna, R., Escorcia, V., Hata, K., Buch, S.: Activitynet challenge 2017 summary. arXiv preprint arXiv:1710.08011 (2017)
43. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: 2017 IEEE International Conference on Computer Vision (ICCV), IEEE (2017) 5534–5542
44. Wang, R., Tao, D.: Uts at activitynet 2016. AcitivityNet Large Scale Activity Recognition Challenge **2016** (2016) 8
45. Heilbron, F.C., Barrios, W., Escorcia, V., Ghanem, B.: Scc: Semantic context cascade for efficient action detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Volume 2. (2017)
46. Shou, Z., Chan, J., Zareian, A., Miyazawa, K., Chang, S.F.: Cdc: Convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. arXiv preprint arXiv:1703.01515 (2017)
47. Xiong, Y., Zhao, Y., Wang, L., Lin, D., Tang, X.: A pursuit of temporal accuracy in general activity detection. arXiv preprint arXiv:1703.02716 (2017)
48. Zhao, Y., Zhang, B., Wu, Z., Yang, S., Zhou, L., Yan, S., Wang, L., Xiong, Y., Lin, D., Qiao, Y., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2017. arXiv preprint arXiv:1710.08011 (2017)