YoTube: Searching Action Proposal via Recurrent and Static Regression Networks

Hongyuan Zhu⋆, Romain Vial⋆, Shijian Lu,
Yonghong Tian, Xianbin Cao

Abstract—In this paper, we present *YoTube*-a novel network fusion framework for searching action proposals in untrimmed videos, where each action proposal corresponds to a spatial-temporal video tube that potentially locates one human action. Our method consists of a recurrent *YoTube* detector and a static *YoTube* detector, where the recurrent *YoTube* explores the regression capability of RNN for candidate bounding boxes predictions using learnt temporal dynamics and the static *YoTube* produces the bounding boxes using rich appearance cues in a single frame. Both networks are trained using rgb and optical flow in order to fully exploit the rich appearance, motion and temporal context, and their outputs are fused to produce accurate and robust proposal boxes. Action proposals are finally constructed by linking these boxes using dynamic programming with a novel trimming method to handle the untrimmed video effectively and efficiently. Extensive experiments on the challenging UCF-101 and UCF-Sports datasets show that our proposed technique obtains superior performance compared with the state-of-the-art.

I. Introduction

Video activity analysis has attracted increasing attention in recent years due to its wide application in video surveillance and human computer interaction. Most works focus on action classification [1], [2], which aims to assign a global category label to a video sequence. On the other side, action detection [3]–[5] which targets to localize the spatial-temporal extent of human actions is equally important by providing localized action information for tasks requiring precise positioning e.g. assistive agents and autonomous vehicles. Similar to the object detection where fast and reliable object proposal can greatly improve the performance of higher level understanding tasks, an efficient action proposal can greatly facilitate video activity analysis [6]–[11] as studied by Yu *et al.* [12].

Action proposal is a challenging task due to significant variations in human pose, illumination, occlusion, blur and background clutter. In addition, the commonly available untrimmed videos bring additional noises for accurate localization of the actions. Existing works have advanced the performance by segmentation-and-merging [3], [4], [13], dense trajectories clustering [14], human detection [12] and deep learning [15]. Most of these methods handle trimmed videos based on image object proposal or their variants, which either use sensitive low-level cues or separate the spatial appearance and temporal motion information learning into two isolated processes, where the later accumulates errors and also hinders the end-to-end optimization.

This paper presents a novel network fusion framework called *YoTube* that locates spatially compact and temporally smooth action proposals within untrimmed videos. A flowmap of our method is shown in Fig. 1. Specifically, our framework consists of a recurrent *YoTube* detector and a static *YoTube* detector. The recurrent *YoTube* explores to produce frame bounding boxes which exploits temporal contexts by empowering recurrent neural network (RNN) with the regression capability. The static *YoTube* is trained to better exploit the rich global appearance cues within each individual frame. These two networks are end-to-end optimized using complementary information of RGB and Flow, and their outputs are fused to produce accurate bounding boxes in each video frame. To this end, our proposed network framework overcomes the limitation of existing works by fusing the appearance, motion and temporal context simultaneously.

The action proposals are finally constructed by linking the candidate action boxes using dynamic programming with a novel path trimming technique. In the first pass, action paths encompassing the whole video are generated by considering their actionness score and overlap in spatial temporal domain. In the second pass, a novel temporal path trimming method is designed which exploits the actioness and background score transition pattern and is capable of handling the paths spanning the whole video.

Our key contributions are three folds. First, we propose a novel network fusion framework which learns appearance, motion and temporal context simultaneously. Second, we explore the regression capability of RNN for spatial-temporal action proposal for the first time. Third, we design an efficient path trimming technique that is capable of handling untrimmed videos without requiring time-consuming techniques of existing methods.

H. Zhu and S. Lu are with Institute for Infocomm Research, A*Star, Singapore (email: {zhuh, slu}@i2r.a-star.edu.sg).

R. Vial is with the Mines ParisTech, France (e-mail:romain.vial@mines-paristech.fr).

Y. Tian is with National Engineering Laboratory for Video Technology (NELVT), School of EECS, Peking University, Beijing, China (email:yhtian@pku.edu.cn)

X. Cao is with School of Automation Science and Electrical Engineering, Beihang University, Beijing, China (email:xbcao@buaa.edu.cn)

II. Related Work

Recurrent Neural Network: Traditional recurrent neural network is designed to incorporate temporal dynamics by
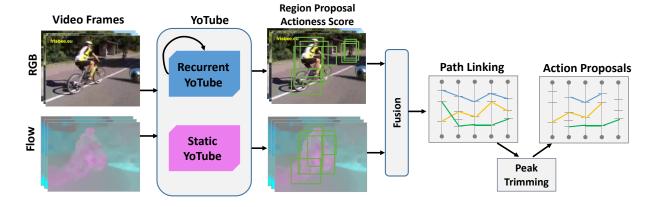
Fig. 1. Conceptual illustration of our method: we explore the regression capability of RNN and CNN to directly regress sequences of bounding boxes. The located bounding boxes are then seamed into longer action proposals with path linking and trimming.

utilizing a hidden state in each recurrent cell. The unit works like a dynamic memory which can be changed according to the previous states. The recurrent update process of RNN can be modeled as follows:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$
$$z_t = \sigma(W_{hz}h_t = b_z)$$
(1)

where h_t is the output hidden state and g_t is the output at time t, and σ is an element-wise non-linearity.

However, conventional RNN is difficult to incorporate long-range information due to the inflating or decaying of the back-propagation error over time. Hence Long-short Term Memory (LSTM) [16] is proposed to incorporate memory that has explicitly control of when to 'forget' and when to 'update' given new information. Recently, LSTM has been actively applied to action classification and video description with state-of-the-art performance [1]. The LSTM unit we use in our work is similar to the one in [16]:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$
$$g_t = \sigma(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot tanh(c_t)$$
(2)

where i_t is the input gate, f_t is the forget gate, o_t is the output gate, g_t is the input modulation gate, and c_t is the sum of previous memory cell c_{t-1} which is modulated by forget gate f_t and modulation gate g_t. An illustration of LSTM is shown in Fig.2.

Regression based Object Detection: Recent advances in deep learning have lifted the performance of state-of-the-art object detection. There are two paradigms: one is based on unsupervised proposal techniques such as selective search [17] and EdgeBox [18], which have boosted initial ConvNet detectors, such as Fast-RCNN. However, hand-crafted object proposal is not directly correlated with the final recognition task which may undermine the performance of object recognition. Hence researchers proposed to use CNN to directly regress the bounding boxes of object proposal using Region
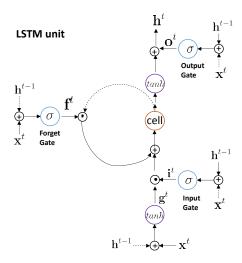

Fig. 2. A diagram of Long-short Term Memory [16].

Proposal Network (RPN) [19] recently. This further boosted the development of other regression based detection methods (e.g. YOLO [20] and SSD [21]) that have demonstrated higher accuracy and speed. YOLO performs inference with global image descriptor, hence can better exploit the context information in whole image to avoid the influence from background. RPN and SSD use local image patches for bounding boxes regression with faster speed, but at a cost of more bounding boxes.

However, these methods are for static images which neglects the useful temporal context among adjacent frames. Our work explores to extend the regression based detectors to spatial temporal domain using RNN for the first time. Our study reveals that, RNN can capture as important information as CNN, and their combination yields superior performance as compared with either component alone.

Video Action Proposal: Early successes in action detection are based on exhaustive search using sliding cuboids [22]–[24]. However, the rigid cuboid is difficult to capture the versatile shape of the human actions. Besides cuboid search, Tran *et al.* [25] explored structure output regression to detect spatial-

temporal action tubes, but they can only search the best action path with fixed size window. Although these early attempts have solved action localization problem to some extent, the tremendously large search space leads to great computational cost. Action proposal has great potential to significantly reduce the search space by generating sequences of bounding boxes with good localization of candidate human actions in spatio-temporal domain. Due to the large volume of literatures, we only review the works which are most directly related with our approach.

Unsupervised image based object proposals [17][26] have been directly extended for video action proposal. Jain *et al.* [3] extend selective search [17] to produce action proposal by clustering the video into voxels, which are then hierarchically merged into action proposals. Similarly, Oneata *et al.* [4] extends the work in [26] by introducing a randomized supervoxel segmentation method for proposal generation. Inspired by the video segmentation method by Brox and Malik [27], Jain *et al.* [14] propose to generation action proposals by clustering long term point trajectories with improved speed and accuracy.

Supervised frame-level human detection has also been introduced to further improve the performance. Yu *et al.* [12] use human detector and generate the action proposal by using max sub-path search. Inspired by the success of deep learning, Gkioxari and Malik [5] propose to train two stream R-CNN networks [28] with selective search to detect action regions. They link the high scored action boxes to form action tubes. Detection-and-tracking methods have also been used for action localization and action proposal. Weinzaepfel *et al.* [29] train a two-stream R-CNN to detect action regions and they also train another instance-level detectors to track the regions with Spatio-Temporal Motion Histogram. Li *et al.* [15] train a single stream RPN [19] to replace R-CNN in [29] for proposal boxes generation and use an improved method of [12] to generate action proposal. The miss detections are remedied by tracking-by-detection and achieved the state-of-the-art performance.

Although aforementioned methods have greatly advanced the quality of action proposal, they still have limitations. Specifically, most of these works [3]–[5], [12], [14] either produce action proposals frame-by-frame individually which ignores the interplay between appearance, motion and temporal context among adjacent frames or arrange the spatial information learning and temporal context learning into isolated processes [15], [29] which produce less satisfactory results. Moreover, most of these methods work on trimmed videos [3]–[5], [14]. To handle untrimmed video, extra detectors need to be trained using low-level features which further accumulates errors [15], [29].

Our method belongs to the two-stream deep learning based approach. Similar to these prior works, we train a static *YoTube* detector to detect frame-level action candidates, but we perform reasoning using global image features to reduce the interference from background clutter. In addition, we trained a recurrent *YoTube* detector to capture the long-term dependency and context among adjacent frames which are largely neglected in those existing methods. The two networks can be optimized end-to-end by simultaneously integrating appearance, motion and temporal dynamics, and their outputs are further fused for accurate candidate action box prediction. Moreover, we design a novel and efficient path trimming technique which is capable of handling untrimmed videos directly without requiring time-consuming techniques of existing methods

III. METHODOLOGY

The proposed method takes an untrimmed video as input and outputs the action proposal accordingly as illustrated in Fig. 1. Our framework consists of two steps: (1) sequential candidate action bounding boxes prediction using recurrent and static *YoTube*; (2) action path linking and trimming. Further details of our method are elaborated in the following sections.
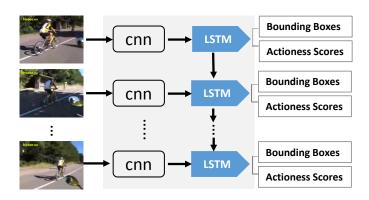

Fig. 3. With a video snippet as input, the proposed recurrent YoTube detector first extracts discriminative features from each frame and then apply the LSTM to regress the coordinates of the bounding boxes. The bounding boxes of each frame is estimated by considering the rich spatial and temporal context in forward direction.

A. YoTube for Action Candidate Boxes Generation

One limitation of existing deep learning based action proposals methods is they either process a video frame-by-frame [5] or separate spatial and temporal information learning into two isolated processes [15], [29]. On the other hand, temporal dynamics and contexts among adjacent frames have been proven to be useful for recent state-of-the-arts in action classification and video description [1]. Inspired by this idea, we design a network fusion framework which can incorporate apperance, motion and temporal context learning in a unified and end-to-end optimizable manner.

We firstly describe the ***recurrent YoTube***, which is a Long-term Recurrent Convolutional Network (LRCN) [1] that combines CNN and RNN for sequence bounding boxes prediction. Note LRCN was initially designed for action classification, which maps a sequence of input feature vectors into sequence of frame lables. In this work, we adapt LRCN to map the sequence of feature vectors to sequence of tensors encoding the object bounding boxes information, as inspired by the recent popular regression based object detectors [19], [20].

Fig. 3 depicts the architecture of recurrent *YoTube* which works by passing each video frame f_t at time t into a CNN to produce a fixed-length feature x_t. Then x_t is passed into the recurrent LSTM, which maps the input x_t and previous time step hidden state h_{t-1} to a new hidden state h_t and bounding

boxes o_t as in Eq. 2. The inference is conducted sequentially from top to bottom as illustrated in Fig. 3, hence the context in earlier frames t_l ($t_l < t$) can be propagated to the current frame t.

The output o_t for frame t is a $K \times K \times (B \times 5 + |S|)$ tensor encodes the output bounding boxes information. Specifically, it means to divide the image into $K \times K$ grids. Each grid cell will predict B bounding boxes which are parameterized by (x, y, w, h, c) where (x, y) represents the center of the box relative to the bounds of the cell. The width w (or height h) is normalized with respect to the image width (height). The confidence c predicts the IoU between the predicted box and any ground-truth box. Moreover, each cell will also predict a score tuple $S = (s_{ac}, s_{bg})$, where s_{ac} and s_{bg} is an actionness score and a background score for the given cell, respectively.

The loss function to be minimized is defined as a sum-squared error between prediction o_t and ground-truth $\hat{o}_t$ for optimization simplicity [20]:

$$
\begin{aligned}
&\lambda_{coord} \sum_{i=0}^{K^2} \sum_{j=0}^{B} 1_{ij}^{obj} \|(x_i, y_i) - (\hat{x}_i, \hat{y}_i)\|^2 \\
&+ \lambda_{coord} \sum_{i=0}^{K^2} \sum_{j=0}^{B} 1_{ij}^{obj} \|(\sqrt{h_i}, \sqrt{w_i}) - (\sqrt{\hat{h}_i}, \sqrt{\hat{w}_i})\|^2 \\
&+ \sum_{i=0}^{K^2} \sum_{j=0}^{B} 1_{ij}^{obj} (c_i - \hat{c}_i)^2 \\
&+ \lambda_{noobj} \sum_{i=0}^{K^2} \sum_{j=0}^{B} 1_{ij}^{noobj} (c_i - \hat{c}_i)^2 \\
&+ \sum_{i=0}^{K^2} 1_i^{obj} \sum_{k \in \{ac, bg\}} (s_k^i - \hat{s}_k^i)^2
\end{aligned} \tag{3}
$$

where $\hat{o}_t^i = (\hat{x}_i, \hat{y}_i, \hat{h}_i, \hat{w}_i, \hat{c}_i, \hat{s}_{ac}^i, \hat{s}_{bg}^i)$ is the cell i of the ground-truth $\hat{o}_t$, 1_i^{obj} denotes if object appears in cell i, 1_{ij}^{obj} denotes that the jth bounding box predictor in cell i is responsible for the prediction (i.e. has the higher IoU with the ground truth between the B boxes) and 1_{ij}^{noobj} denotes that the jth bounding box predictor in cell i is not responsible for the prediction or that there is no ground truth boxes in cell i.

The first two terms penalize coordinates error only when the prediction is responsible for the ground truth box. As deviation in the predicted coordinates matters more for small boxes than large boxes, we take the square root of width and height. The third and fourth terms penalize confidence score error, reduced by a factor λ_{noobj} when the prediction is not responsible for the ground truth box. As most of the grid cells don't contain object, it limits to push confidence score towards zero. The final term penalizes classification as "action" or "background" error only when there is an object in the cell. In this work, we set $\lambda_{coord} = 5$ and $\lambda_{noobj} = 0.5$.

Recurrent *YoTube* is doubly deep in spatial-temporal domain, which can learn the temporal action dynamics. To further exploit the rich RGB and Flow cues in individual frame, we train a ***static YoTube*** which shares the same architecture as the recurrent *YoTube*, but replaces the last LSTM layer with a

fully-connected layer of same number of neurons to regress the coordinates. These two networks complement each other and their outputs are combined to further improve the performance.

B. Path Linking and Trimming

At the end of the detection process (Sec. III-A), we have a set of bounding boxes for each frame of the video $\mathbf{B} = \{\{b_i^{(j)}, j \in [1 \dots N_{b_i}]\}, i \in [1 \dots T]\}$ where T is the length of the video and N_{b_i} is the number of predicted boxes in frame i. For each box $b_i^{(j)}$ we have its confidence score $s_c(b_i^{(j)})$, actionness score $s_{ac}(b_i^{(j)})$ and background score $s_{bg}(b_i^{(j)})$. The next objective is to create a set of proposal paths $\mathbf{P} = \{p_i = \{b_{m_i}, b_{m_{i+1}} \dots b_{n_i}\}, i \in [1 \dots |\mathbf{P}|]\}$ where m_i and n_i are the starting and ending frame of path p_i, respectively.

1) Action Path Linking: In order to link frame-level boxes into coherent path, we firstly define a score for each path given its confidence scores s_c of each box and the IoU of successive boxes:

$$
S(p) = \underbrace{\sum_{i=1}^{T} s_c(b_i)}_{unary} + \lambda_0 \times \underbrace{\sum_{i=2}^{T} IoU(b_i, b_{i-1})}_{pairwise} \tag{4}
$$

$S(p)$ is high for path whose detection boxes have high confidence scores and overlap significantly. λ_0 is a trade-off factor to balance the two terms.

Maximizing Eqn. 4 helps find paths whose detection box scores are high and consecutive detection boxes overlap significantly in spatial and temporal domain. The path which maximizes the energy $\hat{p}_c = \underset{p_c}{argmax} E(p_c)$ can be found with the Viterbi algorithm [5]. Once an optimal path has been found, we remove the bounding boxes in previous path from the frames to construct next path until certain frame doesn't have any boxes.
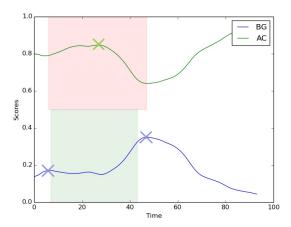

Fig. 4. Illustration of the proposed peak trimming method on one UCF-101 videos: Blue and green curves represent the background and actionness scores, respectively, where blue and green crosses denote score peaks. Green patches represent the ground-truth paths and red patches represent paths that are extracted by using the proposed peak trimming method. It can be observed that the proposed method is capable of trimming the predicted paths accurately thanks to certain action and background score transition patterns.

2) Action Paths Trimming: The generated action paths as described in the last subsection spans the entire video as it greedily optimizes all confidence scores across the paths. On the other hand, human actions typically take up a fraction of it for untrimmed video. It is therefore necessary to perform trimming to remove those boxes that are unlikely belong to the action regions. Mathmetically, we would like to assign each box b_t in a path p with a binary label $y_t \in \{0, 1\}$ (where 'zero' and 'one' represent the 'background' and 'action' class respectively), such that the boxes which are near (or far) from the valid action regions should be assigned to 'action' (or 'background') class as much as possible in final path labeling $\hat{Y}_p = [\hat{y}_0, \hat{y}_1, ..., \hat{y}_T]$.

We also noticed that a transition in background scores typically signifies a change between action and non-action frames. In addition, the boxes within a valid action region often have high actioness scores. Hence, detecting peaks in actioness scores helps find a potential action region, while finding peaks in background score helps define the start and end of the action regions.

Hence, we propose a new method by looking at the transition pattern in the actionness and background scores for the path trimming as illustrated in Fig.4. We first smooth the scores by computing their running average to reduce the influence of noisy classifier scores. All the peaks in both scores are then detected where a peak is defined as a local maximum among at least n neighbors:

$$
\begin{aligned}
peaks_{ac} &= \{t, s_{ac}(b_t) = \max(V_n^{(ac)}(t))\} \\
peaks_{bg} &= \{t, s_{bg}(b_t) = \max(V_n^{(bg)}(t))\}
\end{aligned}
\tag{5}
$$

where $V_n^{(k)}(t) = \{s_k(b_i), i \in [t - n \ldots t + n]\}, k \in \{ac, bg\}$.

Once we have found the peaks, we can select all subsequences to generate the final action proposals by applying the following algorithm:

Algorithm 1 Action paths trimming using actioness and background score peaks.

Input: actioness score peaks $peaks_{ac}$ and background score peaks $peaks_{bg}$.
Output: set $subseq$ consists of trimmed paths.

$subseq = \emptyset$
for $p \in peaks_{ac}$ **do**
    $s = \max(peaks_{bg} < p)$
    $e = \min(peaks_{bg} > p)$
    add path $\{b_s \ldots b_e\}$ to $subseq$
end for

IV. IMPLEMENTATION AND BENCHMARKING

In this section, we discuss the details of implementation and benchmarking, including the dataset and evaluation metrics.

A. Training

The CNN architecture we use for feature extraction in *YoTube* is adapted from [20] which has 24 convolution layers and 2 fully-connected layer. We firstly replacing the last two fully connected layer with a locally connected layer with 256 filters with a 3×3 kernel. On top of the locally connected layer, we train an LSTM layer with 588 neurons which directly regresses the bounding boxes' coordinates. We choose locally connected layer to stablize training and improve convergence. The number of neurons in last layer means to divide the image into 7×7 grids and each grid predicts 2 bounding boxes.

For the RGB stream, the convolutional part of our model is pretrained on the ImageNet 1000-class dataset [30]. For the Flow stream, the convolutional part is pretrained with the weights of the RGB stream. The top layers are initialized using the method in [31]. We found no problem in convergence by initializing the weights of our Flow model with the weights of the RGB model despite the notable difference between the images distribution.

We make an extensive use of data augmentation to prevent over-fitting. This part is non negligible due to important correlation between frames of the same video. In addition to mirroring, we use corner cropping and center cropping. It means that we take a 224×224 crop from the 320×240 frame in each corner and the center. Then we resize this crop to the input size of 448×448. This method permits to increase the size of the dataset by a factor 12.

We use Adam [32] optimizer during training with default parameters. When training the static *YoTube*, we use a batch size of 32 frames from different videos during 100 epochs with an initial learning rate of 10^{-4} decaying at 10^{-5} after the 20th epoch. When training the recurrent *YoTube*, we freeze the weighs of the convolutional layers to avoid a catastrophic forgetting. We then use a batch size of 10 sequences of 10 frames from different videos during 50 epochs. The same learning rate planning is used.

B. Datasets

UCF-101. The UCF-101 dataset is a large action recognition dataset containing 101 action categories with more than 13,000 videos and an average of 180 frames per video. A subset of 24 categories are used for the localization task with bounding box annotation corresponding to 3,204 videos. Part of these videos are untrimmed (around 25%), permitting to validate the efficiency of our trimming methods. Each video contains one or more instances of same action class. It has large variations in terms of appearance, scale, motion, etc with much diversity in terms of actions. Three train/test splits are provided with the dataset, and we perform experiments on the first split with 2,290 training videos and 914 testing videos.

UCF-Sports. This dataset contains 150 sport broadcast videos with realistic actions captured in dynamic and cluttered environments. It is challenging considering many actions with large displacement and intra-class variation. These videos have been trimmed to contain a single action instance without interruption. There are ten categories in the dataset, e.g. diving, swinging bench, horse riding, etc. We used the train-test split of videos suggested in [14] (103 for training and 47 for testing). The ground truth is provided as sequences of bounding boxes enclosing the actions.
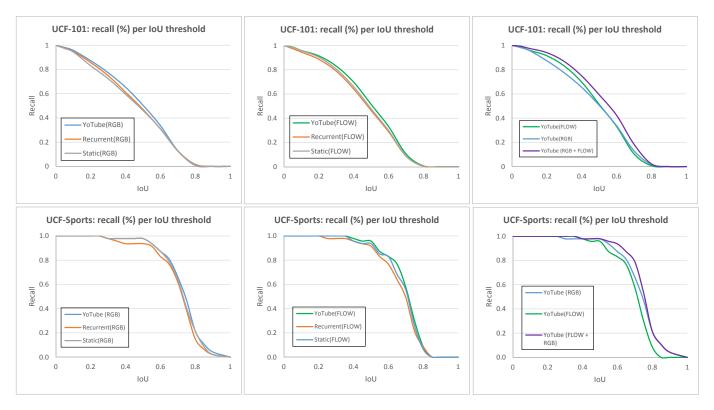
Fig. 5. Ablation study of two stream's static YoTube and recurrent YoTube on the UCF-101 (top-row) and UCF-Sports (bottom row) datasets, left column shows RGB stream, middle column shows the Flow stream and right column shows their ensemble.

C. Evaluation metrics

ABO, MABO: We use two common metrics as in [14] to report overall performance, namely Average Best Overlap (ABO) and Mean ABO (MABO). The overlap (OV) between a path $\mathbf{d} = \{d_s \ldots d_e\}$ and a ground truth path $\mathbf{g} = \{g_s \ldots g_e\}$ is defined as follows:

$$OV(\mathbf{d}, \mathbf{g}) = \frac{1}{|\mathbf{d} \bigcup \mathbf{g}|} \times \sum_{i \in \mathbf{d} \bigcap \mathbf{g}} \frac{d_i \bigcap g_i}{d_i \bigcup g_i}$$

$$|\mathbf{d} \bigcup \mathbf{g}| = \max(d_e, g_e) - \min(d_s, g_s)$$

$$\mathbf{d} \bigcap \mathbf{g} = [\max(d_s, g_s) \ldots \min(d_e, g_e)]$$

where d_s and d_e are the detected bounding boxes in the starting and ending frame of a path, g_s and g_e are the bounding boxes in the starting and ending frame of the ground-truth path.

ABO measures the best localization from the set of action proposals $D = \{d_j | j = 1...m\}$ for the each ground-truth G, where ABO(c) is the ABO computed for the ground-truth G_c of class c. The mean ABO (MABO) measures the average performance across all classes.

$$\text{ABO} = \frac{1}{|\mathbf{G}|} \sum_{\mathbf{g} \in \mathbf{G}} \max_{\mathbf{d} \in \mathbf{D}} OV(\mathbf{d}, \mathbf{g})$$

$$\text{ABO}(c) = \frac{1}{|\mathbf{G^c}|} \sum_{\mathbf{g} \in \mathbf{G^c}} \max_{\mathbf{d} \in \mathbf{D}} OV(\mathbf{d}, \mathbf{g})$$

$$\text{MABO} = \frac{1}{|\mathbf{C}|} \sum_{c \in \mathbf{C}} \text{ABO}(c)$$

UCF101	ABO	MABO	Recall	#Prop.
RGB Stream				
Static (S)	44.94	45.42	46.13	**10**
Recurrent (R)	45.85	45.83	47.05	21
YoTube (RGB)	**47.60**	**47.78**	**50.61**	35
Flow Stream				
Static (S)	46.87	47.02	47.63	33
Recurrent (R)	46.09	46.45	46.3	**9**
YoTube (FLOW)	**48.61**	**48.83**	**51.29**	42
Ensemble				
YoTube-RGB+FLOW (NO TRIM)	45.36	46.42	52.03	80
YoTube-RGB+FLOW	**52.45**	**52.92**	**59.19**	73

TABLE I
ABLATION STUDY ON THE UCF-101 DATASET.

where $\mathbf{C}$ is the set of action classes, $\mathbf{G}$ is the set of ground truth paths and $\mathbf{G}^c$ is the set of ground truth paths for action class c.

Recall vs IoU: Another commonly used metric is the Recall vs IoU [18], which measures the fraction of ground-truths detected in a set of overlap threshold. An instance of action, g_i is correctly detected by an action proposal d_j if the overlap score is higher than a threshold η i.e.: $OV(d_j, g_i) \geq \eta$ where $\eta \in [0, 1]$. In our work, we target to maximize the recall at a 0.5 threshold as other works [14], [15].

V. EXPERIMENTAL RESULTS

In our experiments, we first compare recurrent *YoTube* and static *YoTube* in rgb and optical flow streams. Then, we study the impact of path-trimming on the detecction performance. Finally, we compare the *YoTube* with the state-of-the-art.
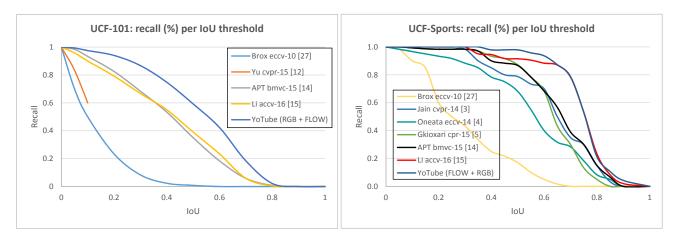
Fig. 6. Comparison with other state-of-the-arts on UCF-101 and UCF-Sports dataset, performance is measured by recall for various IoU thresholds.
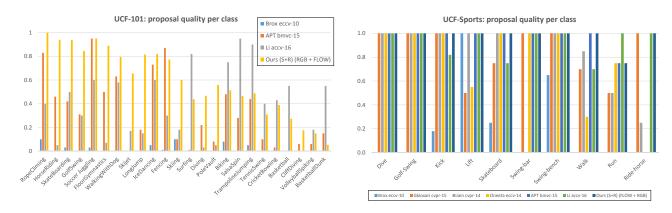

Fig. 7. Comparison with other state-of-the-arts on UCF-101 and UCF-Sports, performance is measured by the recall on each action class.

UCF-Sports	ABO	MABO	Recall	#Prop.
RGB Stream				
Static (S)	71.64	72.54	**97.87**	**20**
Recurrent (R)	70.08	71.5	93.62	**20**
YoTube (RGB)	**72.45**	**73.54**	**97.87**	30
Flow Stream				
Static (S)	68.08	68.98	93.62	**20**
Recurrent (R)	66.22	66.91	91.49	**20**
YoTube (FLOW)	**69.08**	**69.95**	95.74	30
Ensemble				
YoTube-RGB+FLOW (NO TRIM)	**74.44**	**75.31**	**97.87**	30
YoTube-RGB+FLOW	**74.44**	**75.31**	**97.87**	30

TABLE II
ABLATION STUDY ON THE UCF-SPORTS DATASET.

A. Recurrent YoTube vs Static YoTube

The ablation comparison between recurrent and static *YoTube* for two streams in UCF-101 and UCF-Sports are shown in Fig. 5, Table I and II. In UCF-101, the performance of recurrent version is slightly better than the static version in RGB stream with around 1% improvements in recall and the ensemble model (YoTube(RGB)) achieves another 3.56% improvement as shown in Table I. These results prove that the recurrent model and static model are complementary. We conjecture this is due to the RNN which captures the temporal dynamics among adjacent frames. For the results of Flow stream in Table I, the recall of static version is 1.3% better than the

recurrent version and the ensemble model (YoTube(FLOW)) achieves another 4% improvement than the static version, which further confirms the complementariness of two methods. The slightly inferior result of recurrent version is probably caused by lacking training data. In addition, the ensemble flow stream (YoTube(FLOW)) performs slightly better than the ensemble rgb stream (YoTube(RGB)). This is probably because the flow field eliminates the interference from the background. The final model (YoTube (RGB+FLOW)) which combines two streams achieves another 8% improvement than the flow stream in recall, which demonstrates that the RGB and Flow streams also complement each other.

For UCF-Sports dataset, the static version has 4% better recall than the recurrent version as shown in Table II, whereas the ensemble model have the same recall as the static version, it has higher ABO and MABO for better localization. For the flow stream, the static version has 2.2% better recall than the recurrent version in the interval, the ensemble model is 2% better recall than the static version. The slightly inferior performance of recurrent YoTube could be caused by the small number of training sample. On the other hand, the higher ABO and MABO of the ensembles between static and recurrent model in both streams still proves that the two models capture complementary information to improve localization.

Moreover, we also show the result of the model without

path trimmming (NO TRIM) in both Table I and II. As UCF-101 dataset contains un-trimmed video, the performance of our method without path trimming is nearly 7% lower in recall and also contains more noisy paths. This shows that the proposed path trimming is effective for the untrimmed video. For the UCF-Sports dataset which consists of trimmed videos, the performance with or without path trimming is identical. This result also proves that our method is adaptive to the video content.

B. Comparison to state-of-the-arts

We compare our method with state-of-the-arts on UCF-101 and UCF-Sports datasets. The recall-vs-IoU and recall-per-class curves for both datasets are shown in Fig. 6 and Fig. 7.

For UCF-101 dataset, our method out-performs the state-of-the-art [15] by 20% or more in all range of IoU. Although Li *et al.* use deep network (RPN) [19], they use only one stream and their performance is only 4% better than the unsupervised method APT [14] in terms of the recall as shown in Table III. Notwithstanding, the recall of our single stream design in RGB and Flow out-performs Li *et al.*[15] by 11% and 12% respectively, which proves the superiority of *YoTube* by using spatial-temporal modeling and two-stream design. Yu *et al.* [12] is based on a human detector with low-level features, which is difficult to handle large dynamic changes in the scene. The work in [27] is mainly designed for non-overlap segmentation and using low-level features, hence it is sub-optimal for the task and has the lowest recall. According to the per-class recall curve in Fig. 7, our method is better than Li *et al.* [15] in many classes, especially for classes that have large motion change (e.g. 'skijet', 'floor gymnastics' and 'long jump').

For UCF-Sports dataset, our method also out-performs Li *et al.* [15] by nearly 6% in terms of recall in Table IV. Moreover, the deep learning based approaches (our method and Li *et al.* [15]) also largely out-perform the unsupervised methods, which proves the effectiveness of the deep networks' discriminative features. The per-class recall curve for all methods is also provided in Fig. 7.

We also evaluate our methods in terms of other metrics, e.g. ABO, MABO, number of proposals which are also listed in Table III and IV. Our method produces the highest MABO, Recall using slightly higher number of proposal than Li *et al.* [15], while still relatively smaller than [14] and [12]. Although [15] achieves the highest ABO in both datasets, there are big differences between ABO and MABO. Actually, these two measurements should be in the similar scale according to the formula in Sec. IV-C, i.e. MABO is the mean of ABO for all classes.

VI. CONCLUSION

We propose a novel framework for video action proposal. Given an untrimmed video as input, our method produces a small number of spatially compact and temporally smooth action proposals. The proposed method explores the regression capability of RNN and CNN to produce frame-level candidate action boxes using rgb, flow and temporal contexts among

UCF101	ABO	MABO	Recall	#Prop.
Brox & Malik [27]	13.28	12.82	1.40	**3**
Yu *et al.* [12]	n.a	n.a	0.0	10,000
APT [14]	40.77	39.97	35.45	2299
Li *et al.* [15]	**63.76**	40.84	39.64	18
YoTube-RGB	47.60	47.78	50.61	35
YoTube-FLOW	48.61	48.83	51.29	42
YoTube-RGB+FLOW	52.45	**52.92**	**59.19**	73

TABLE III
QUANTITATIVE COMPARISON ON THE UCF-101 DATASET. RECALL IS
COMPUTED AT A PRECISION THRESHOLD OF 0.5.

UCF Sports	ABO	MABO	Recall	#Prop.
Brox & Malik [27]	29.84	30.90	17.02	**4**
Jain *et al.* [3]	63.41	62.71	78.72	1642
Oneata *et al.* [4]	56.49	55.58	68.09	3000
Gkioxari *et al.* [5]	63.07	62.09	87.23	100
APT [14]	65.73	64.21	89.36	1449
Li *et al.* [15]	**89.64**	74.19	91.49	12
YoTube-RGB	72.45	73.54	97.87	30
YoTube-Flow	69.08	69.95	95.74	30
YoTube-RGB+Flow	74.44	**75.31**	**97.87**	30

TABLE IV
QUANTITATIVE COMPARISON ON THE UCF-SPORTS DATASET. RECALL IS
COMPUTED AT A PRECISION THRESHOLD OF 0.5.

frames, hence greatly increases the accuracy and meanwhile reduces the number of false positives. The action proposals are constructed using dynamic with a novel path trimming methods. The experiments on the UCF-Sports and UCF-101 datasets highlight the effectiveness of our proposed modeling.

REFERENCES

[1] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venu-gopalan, T. Darrell, and K. Saenko, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[2] Y. Wang, J. Song, L. Wang, L. Van Gool, and O. Hilliges, "Two-stream sr-cnns for action recognition in videos," in *BMVC*, 2016.

[3] M. Jain, J. C. van Gemert, H. Jégou, P. Bouthemy, and C. G. M. Snoek, "Action localization with tubelets from motion," in *CVPR*, 2014.

[4] D. Oneata, J. Revaud, J. J. Verbeek, and C. Schmid, "Spatio-temporal object detection proposals," in *ECCV*, 2014.

[5] G. Gkioxari and J. Malik, "Finding action tubes," in *CVPR*, 2015.

[6] G. Yu, J. Yuan, and Z. Liu, "Action search by example using randomized visual vocabularies," *IEEE Trans. Image Processing*, vol. 22, no. 1, pp. 377–390, 2013.

[7] J. Fan, X. Shen, and Y. Wu, "What are we tracking: A unified approach of tracking and recognition," *IEEE Trans. Image Processing*, vol. 22, no. 2, pp. 549–560, 2013.

[8] G. Zhao, J. Yuan, G. Hua, and J. Yang, "Topical video object discovery from key frames by modeling word co-occurrence prior," *IEEE Trans. Image Processing*, vol. 24, no. 12, pp. 5739–5752, 2015.

[9] Y. Jiang, J. Meng, J. Yuan, and J. Luo, "Randomized spatial context for object search," *IEEE Trans. Image Processing*, vol. 24, no. 6, pp. 1748–1762, 2015.

[10] K. R. Jerripothula, J. Cai, and J. Yuan, "CATS: co-saliency activated tracklet selection for video co-localization," in *ECCV*, 2016.

[11] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," in *CVPR*, 2016.

[12] G. Yu and J. Yuan, "Fast action proposals for human action detection and search," in *CVPR*, 2015.

[13] S. Ma, J. Zhang, N. Ikizler-Cinbis, and S. Sclaroff, "Action recognition and localization by hierarchical space-time segments," in *ICCV*, 2013.

[14] J. C. van Gemert, M. Jain, E. Gati, and C. G. M. Snoek, "APT: action localization proposals from dense trajectories," in *BMVC*, 2015.

[15] N. Li, D. Xu, Z. Ying, and G. L. Zhihao Li, "Search action proposals via spatial actionness estimation and temporal path inference and tracking," in *ACCV*, 2016.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[18] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014.

[19] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," in *ECCV*, 2016.

[22] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *ICCV*, 2005.

[23] T. Lan, Y. Wang, and G. Mori, "Discriminative figure-centric models for joint action localization and recognition," in *ICCV*, 2011.

[24] Y. Tian, R. Sukthankar, and M. Shah, "Spatiotemporal deformable part models for action detection," in *CVPR*, 2013.

[25] D. Tran and J. Yuan, "Max-margin structured output regression for spatio-temporal action localization," in *NIPS*, 2012.

[26] S. Manen, M. Guillaumin, and L. J. V. Gool, "Prime object proposals with randomized prim's algorithm," in *ICCV*, 2013.

[27] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010.

[28] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[29] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *ICCV*, 2015.

[30] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *ICCV*, 2015.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980