YoTube: Searching Action Proposal Via Recurrent and Static Regression Networks

Hongyuan Zhu, *Member, IEEE*, Romain Vial, Shijian Lu, Xi Peng, Huazhu Fu, Yonghong Tian, *Senior Member, IEEE*, and Xianbin Cao, *Senior Member, IEEE*

***Abstract*— In this paper, we propose *YoTube*—a novel deep learning framework for generating action proposals in untrimmed videos, where each action proposal corresponds to a spatial-temporal tube that potentially locates one human action. Most of the existing works generate proposals by clustering low-level features or linking image proposals, which ignore the interplay between long-term temporal context and short-term cues. Different from these works, our method considers the interplay by designing a new recurrent *YoTube* detector and static *YoTube* detector. The recurrent *YoTube* detector sequentially regresses candidate bounding boxes using Recurrent Neural Network learned long-term temporal contexts. The static *YoTube* detector produces bounding boxes using rich appearance cues in every single frame. To fully exploit the complementary appearance, motion, and temporal context, we train the recurrent and static detector using RGB (Color) and flow information. Moreover, we fuse the corresponding outputs of the detectors to produce accurate and robust proposal boxes and obtain the final action proposals by linking the proposal boxes using dynamic programming with a novel path trimming method. Benefiting from the pipeline of our method, the untrimmed video could be effectively and efficiently handled. Extensive experiments on the challenging UCF-101, UCF-Sports, and JHMDB datasets show superior performance of the proposed method compared with the state of the arts.**

***Index Terms*— Image sequence analysis, object detection, activity recognition.**

Manuscript received May 4, 2017; revised December 7, 2017; accepted January 16, 2018. Date of publication February 14, 2018; date of current version March 12, 2018. The work of X. Peng was supported by the National Nature Science Foundation of China under Grant 61432012 and Grant U1435213 and in part by the Fundamental Research Funds for the Central Universities under Grant YJ201748. The work of Y. Tian was supported by the National Natural Science Foundation of China under Contract U1611461, Contract 61390515, and Contract 61425025. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Aydin Alatan. *(Hongyuan Zhu and Romain Vial contributed equally to this work.) (Corresponding authors: Hongyuan Zhu; Romain Vial; Xi Peng.)*

H. Zhu and H. Fu are with the Institute for Infocomm Research, A*Star, Singapore 138632 (e-mail: zhuh@i2r.a-star.edu.sg; fuhz@i2r.a-star.edu.sg).

R. Vial is with the Mines ParisTech, 75006 Paris, France (e-mail: romain.vial@mines-paristech.fr).

S. Lu is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: shijian.lu@ntu.edu.sg).

X. Peng is with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: pangsaai@gmail.com).

Y. Tian is with the National Engineering Laboratory for Video Technology, School of EECS, Peking University, Beijing 100871, China (e-mail: yhtian@pku.edu.cn).

X. Cao is with the School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China (e-mail: xbcao@buaa.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2806279

I. Introduction

ACTION proposal aims to extract a small number of spatial-temporal paths to cover all potential regions corresponding to human actions. As it could significantly reduce the size of search space, there is increasing attention in video analytics tasks [1]–[6]. Different from action recognition [7]–[13], action proposal mines all potential action regions from one video rather than classifying the whole video into existing categories. Comparing with action detection [14]–[16], action proposal generates generic action regions instead of focusing on the specific action defined by training data. A conceptual illustration of the difference among action proposal, action recognition, and action detection could be found in Fig. 1.

Despite the success of object proposals in images [17], [18], it is extremely challenging to generate action proposals in videos due to following reasons. First, the extensively investigated image object proposal only relies on appearance and spatial cues, whereas action proposal takes appearance, motion and temporal information into consideration. What is more challenging is learning effective actioness cues to differentiate human actions from commonly occurred background clutters and other dynamic motion, given the diversity and variations of human actions. Second, the search space of action proposals is exponentially larger than image object proposal since the former is with the additional temporal dimension. In practice, it is infeasible to enumerate all possible candidates to pick action proposals. Third, the raw video content is generally untrimmed, which brings in temporal noises and needs further elaborate post-processing to trim the action paths.

To holistically address the above challenges, we propose a novel deep learning framework called *YoTube* that generates spatially compact and temporally smooth action proposals for untrimmed videos by simultaneously considering the appearance, motion and temporal information. In details, our framework consists of a novel recurrent *YoTube* detector and a static *YoTube* detector. The recurrent *YoTube* detector is based on a novel recurrent regression network that sequentially predicts the bounding boxes using adjacent frame temporal contexts in one-shot. The static *YoTube* detector is designed for better exploiting the rich global appearance and motion cues within each individual frame. The outputs of these two networks are further fused to exploit the complementary information between short-term and long-term contexts. After that, our method gives initial action proposals by linking the candidate

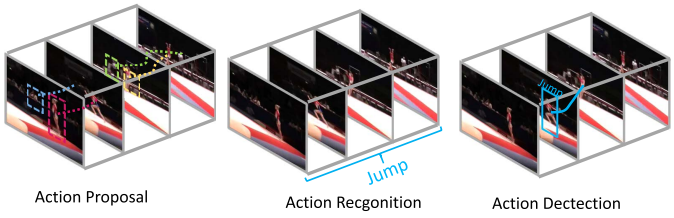


Fig. 1. Conceptual illustration of the difference among action proposal, action recognition, and action detection. Action proposal tries to extract all possible human actions which are highlighted using dash-lines. Action recognition aims to classify the whole video into specific categories, e.g., 'Jump'. Action detection aims at detecting when and where a specific action appears (highlighted using solid-lines).

action boxes in terms of their actionness score and overlap in the spatial-temporal domain. Furthermore, we propose a novel temporal path trimming method to handle the untrimmed videos by utilizing the actioness and background score transition pattern.

The contributions of our paper lie in following aspects:

- A novel deep learning framework for action proposal is proposed, which learns discriminative actioness cues from video by considering the short-term appearance, motion information, and the long-term temporal information.
- A novel recurrent regression network is introduced to capture the long-term temporal information for action proposal, which is ignored in recent works.
- An efficient and accurate path trimming technique is proposed to deal with untrimmed videos.
- Extensive experiments are carried out on UCF-101, UCF-Sports and JHMDB dataset, which demonstrate the superior performance of our method.

A preliminary conference version of our work appeared in [19], which only adopts the RGB (Color) information and applies a less efficient path trimming method. Moreover, the experiment was conducted only on the UCF-101 dataset. This paper extends the conference work by additionally considering the motion information from flow images and proposing a more efficient and accurate path trimming method to trim the action proposal. Furthermore, this work performs more detailed analysis and extensive experiments involving more state-of-the-art methods on three challenging datasets (UCF-101, UCF-Sports, and JHMDB dataset).

II. Related Work

A. Recurrent Neural Network

Recurrent Neural Network (RNN), especially Long-Short Term Memory (LSTM) [20] has become popular for sequence generation and prediction tasks. A detailed survey of recent RNN models and applications could be found in [21]. In this work, we mainly discuss RNNs based action recognition and detection.

Veeriah *et al.* [22] proposed a differential gating scheme to capture the changes between two successive frames. Donahue *et al.* [23] developed a novel recurrent convolutional architecture and successfully applied it to video recognition, image description and video narration. Wu *et al.* [24], [25] and Ng *et al.* [13] demonstrated that an average fusion of LSTMS with appearance and flow boosts the prediction performance. Although our work also employs LSTM to learn long-term temporal contexts, it is different from existing works since we employ the LSTM to predict bounding boxes instead of class labels. Moreover, we trained a CNN using a larger resolution images for feature extraction, which facilitates the action proposal task.

Recently, RNN has been applied to refine tracking-by-detection result. For example, Ni *et al.* [26] applied an LSTM to refine the tracked objects or human parts captured in each image frame. Stewart *et al.* [27] applied an LSTM to aggregate contexts from adjacent detections so that the detected face in the image is progressively refined. Comparing with these methods, the proposed recurrent regression network is designed for video action proposal, which predicts bounding boxes in one-shot without a usage of other detectors and multiple pass regime, thus embracing computational efficiency.

B. Regression Based Object Detection

Most recent deep learning detection methods perform detection by classifying object proposals (e.g. selective search [17], EdgeBox [28]) or directly regressing the coordinates of bounding boxes based on local features. The typical methods include but not limited to Region Proposal Network [29] and SSD [30]. Recently, Redmon *et al.* [31] propose YOLO to perform inference using global image feature and achieves impressive results, which exploits the context information of the whole image to avoid the influence from the background.

To reduce the influence from the background, our work also exploits the global image features for bounding box regression. The architecture of our CNN is different from that of YOLO. More specifically, we replace the last two fully connected layers of YOLO with a locally connected dense layer, thus reducing the computational overhead. Experimental result verifies that such a difference improves the accuracy of our method by nearly 5%. Moreover, YOLO and other detectors are mainly designed for image object detection, which neglects the useful temporal context information. In contrast, our work explores the regression capability of RNN for video action proposal. The extensive experimental study reveals that the combination of RNN and CNN yields superior performance over either one of them alone.

C. Action Recognition

Following the impressive performance of CNNs in image recognition, deep learning approaches were applied to action recognition. A thorough survey on recent deep action recognition methods could be found in [32] and [33]. Here we summarize the influential works according to different pipelines.

Existing deep action recognition architectures could be divided into three groups according to [33]. To be exact, 1) image based action recognition methods directly extract the off-the-shelf CNN features pre-trained on the ImageNet and then pass the features through a learned SVM classifier [7], [8]; 2) end-to-end snippet learning methods learn video features using the appearance cues of short video snippets. For example, Ji *et al.* [34] introduced the 3D-CNN

that operates on the stacked video frames. Karpathy *et al.* [9] compared several similar 3D-CNN architectures on the large-scale video classification task. Tran *et al.* [10] proposed the C3D model which has inspired numerous works, such as R-C3D [35] and Segment-CNN [36]. Recently, Simonyan and Zisserman [11] propose a two-stream approach to break down the video feature learning into the learning of separate spatial and temporal clues, thus greatly reducing the learning overload in C3D and improving the recognition performance. In [12], the two-stream approach has been extended with dense trajectories. 3) long-term temporal modeling overcomes the limitations of short-snippet learning method by using RNN/LSTM to capture the long range temporal contexts. For example, Donahue *et al.* [23] and Wu *et al.* [24] trained an LSTM on the top of CNN for video recognition. Ng *et al.* [13] further stacked multiple layers of LSTM and compared different pooling strategies. The technical difference with the long-term temporal modeling is discussed in the section of Recurrent Neural Network.

Our work explores to combine the advantages of short-snippet modeling and long-term temporal modeling in action recognition for action proposal. We show that both networks are complementary with each other and could deliver significant performance improvement.

D. Action Proposal

To reduce the search space, action proposal generates sequences of bounding boxes with good localization of candidate human actions in the spatio-temporal domain, which avoids the rigid structure requirement of early works [37]–[40].

Unsupervised image proposals [17], [18] have been extended for video action proposal. Jain *et al.* [41] extend Selective Search [17] by clustering the videos into voxels and then hierarchically merging the voxels to produce action proposals. Similarly, Oneata *et al.* [42] extend [18] by introducing a randomized supervoxel segmentation method for proposal generation. Inspired by the video segmentation method in [43], van Gemert *et al.* [44] propose to generate action proposals by clustering long term point trajectories with improved speed and accuracy. However, they are based on low-level features which make difficulties in handling videos with rich motion.

Supervised detectors have also been introduced to the action proposal by learning actioness cues using labels. Yu and Yuan [45] use human detector and generate the action proposal by using max sub-path search. Inspired by the success of deep learning, Gkioxari and Malik [46] propose to train two stream R-CNN networks [47] which learns actioness cues with selective search to detect action regions. They link the high scored action boxes to form action tubes. In class-specific action detection, Saha *et al.* [15] and Peng and Schmid [16] propose to use region proposal networks (RPN) to generate frame proposals and then classify these regions are by Fast-RCNN. Detection-and-tracking methods have also been used for action localization and action proposal. Weinzaepfel *et al.* [14] train a two-stream R-CNN to detect action regions and an additional instance-level detector to track the regions with Spatio-Temporal Motion Histogram. Inspired by these works, Li *et al.* [48] also train RPN [29] to replace R-CNN in [14] for generating proposal boxes, Moreover, their method uses an improved method of [45] to generate action proposals. The missed detections are remedied by tracking-by-detection and their method has achieved the state-of-the-art performance.

Most of these works [41], [42], [44]–[46] generate proposals for each frame individually without considering the temporal contexts or just considering the contexts in very short snippets [48]. Moreover, they generally work on trimmed videos [41], [42], [44], [46]. To handle untrimmed video, extra detectors need to be trained using low-level features, thus leading to errors accumulation and higher time consumption [14]–[16], [48].

Different from the above works, we propose a 4-way network fusion scheme to combine the short-term and long-term information given by the recurrent and static regression networks respectively. Our method uses a novel recurrent regression network to capture the long-term temporal context which is largely neglected in recent works of action proposal. In addition, we use the global features instead of local features to perform inference, thus reducing the interference from background clutter. Finally, we design an efficient path trimming technique which is capable of handling untrimmed videos directly without requiring time-consuming techniques of existing methods [14]–[16], [48]

E. Information Fusion

Fusing information from multiple sources has shown effectiveness in various tasks. A comprehensive survey on this topic could be found in [49] and [50]. Here, we mainly focus on the works related to object detection, action/event recognition and action detection based on neural networks.

A popular approach is to perform fusion at the input level. Some static object detection methods stack images generated from multiple-sensors such as multi-spectral camera [51] and RGB-D camera [52]. To achieve action recognition, C3D [10], [34] applies the 3D-Convolution on stacked image frames. Another popular approach is to perform fusion at the feature level. In RGB-D object/scene-recognition, Wang *et al.* [53], [54] and Zhu *et al.* [55] proposed to use auto-encoder and correlation analysis to fuse the features from RGB and Depth, respectively. Wu *et al.* [24] and Feichtenhofer *et al.* [56] proposed to directly concatenate the RGB and Flow features for action recognition. The third approach is to perform fusion at the output level. Recently, Zhang *et al.* [57] propose using an auto-encoder to perform late fusion on the results of dense trajectories generation, scene classification and object detection. Zhang *et al.* [58] also proposed dynamically fusing the motion and image cues for video description. Simonyan and Zisserman [11] proposed a late fusion scheme to train a two-stream network for the purpose of extracting features from RGB and Flow frames.

Our model adopts the late fusion since [11], [15], [16], [24] has validated its effectiveness in class-specific action recognition and action detection. The major difference between our work and existing ones is that we perform 4-way fusion by considering the recurrent and static regression networks

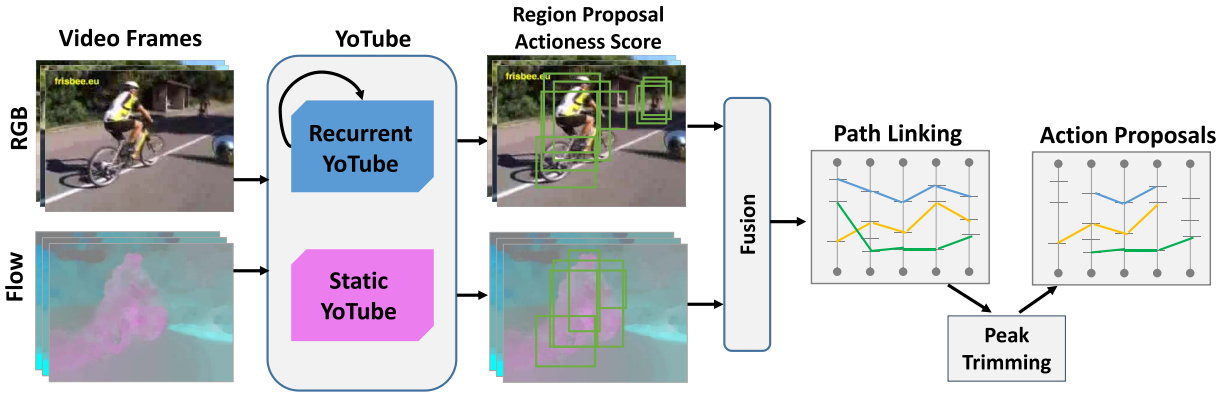


Fig. 2. Conceptual illustration of our method: we utilize the regression capability of RNN and CNN to directly regress the sequences of bounding boxes and then seam the boxes into longer action proposals using path linking and trimming.

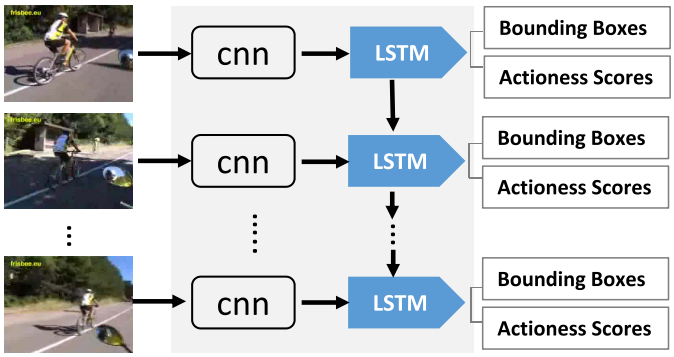


Fig. 3. Using a video snippet as the input, the proposed recurrent YoTube detector first extracts discriminative features from each frame and then applies the LSTM to regress the coordinates of the bounding boxes. The bounding boxes of each frame is estimated by considering the rich spatial and temporal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

trained on RGB and Flow frames for action proposal. Another advantage of our model is that it does not require any external multi-channel sensors such as [51]–[55]. As a result, our method is easier to deploy. It should be pointed out that [59] is is proposed for object tracking, in which the "fusion" concept is different from the score/modality fusion concept discussed in our work. The fusion concept here is more about "model updating", i.e., updating the old model by using newly classified examples.

III. Methodology

The proposed method inputs an untrimmed video and outputs the action proposal accordingly. Fig. 2 illustrates the flowchart of our framework which consists of two steps: 1) using recurrent and static *YoTube* to predict sequential candidate action bounding boxes; 2) linking and trimming action path.

A. YoTube for Action Candidate Boxes Generation

One limitation of existing deep action proposal methods is that they either process a video frame-by-frame [46] or separate spatial and temporal information learning into two isolated processes [14], [48]. Moreover, temporal dynamics and contexts between two adjacent frames have been proven to be useful in action classification and video description [23]. In this paper, we design a neural network based fusion framework to incorporate the appearance, motion and temporal context learning in an end-to-end optimizable manner.

We first describe the ***recurrent YoTube*** — a recurrent regression network, which is used to predict the bounding boxes for each frame by leveraging the temporal contexts from adjacent frames. Fig. 3 depicts the architecture of recurrent *YoTube* detector which passes the frame f_t at time t into a CNN to produce a fixed-length feature x_t. Then a recurrent LSTM maps x_t and the previous time step hidden state h_{t-1} into a new hidden state h_t and bounding boxes o_t. The inference is sequentially conducted from top to bottom as illustrated in Fig. 3. Benefit from the used network, the context in earlier frames t_l ($t_l < t$) can be propagated into the current frame t.

The output o_t for the frame t is a $K \times K \times (B \times 5 + |S|)$ tensor which encodes the output bounding boxes information. Specifically, the image is divided into $K \times K$ grids and each grid cell will predict B bounding boxes 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 (height h) is normalized with respect to the image width (height). The confidence c predicts the IoU between the predicted box and the ground-truth. 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 is defined as a sum-squared error between the prediction o_t and the ground-truth $\hat{o}_t$ for the simplicity in optimization [31]:

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

where $\hat{o}_t^i = (\hat{x}_i, \hat{y}_i, \hat{h}_i, \hat{w}_i, \hat{c}_i, \hat{s}_{ac}^i, \hat{s}_{bg}^i)$ denotes the cell i of the ground-truth $\hat{o}_t$. 1_i^{obj} indicates that the object appears in cell i, and 1_{ij}^{obj} indicates that the j^{th} bounding box predictor in cell i is responsible for the prediction (i.e., has a higher IoU with the ground truth between the B boxes). In contrast, 1_{ij}^{noobj} denotes that the j^{th} 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. Since the deviation in the predicted coordinates more better for smaller boxes than large ones, 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. Since most of the grid cells do not contain objects, it pushes 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 empirically 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 RGB and Flow cues in each single frame, we propose ***static YoTube*** which shares the same architecture as the recurrent *YoTube* with only one difference, *i.e.*, the static YoTube replace the last LSTM layer of the recurrent YoTube with a fully-connected layer with the same number of neurons. These two networks complement each other and thus combining their outputs could further improve the performance.

B. Path Linking and Trimming

Once the detectors (Sec. III-A) output a set of bounding boxes for each frame (denoted by $\mathbf{B} = \{\{b_i^{(j)}, j \in [1 \dots N_{b_i}]\}, i \in [1 \dots T]\}$), we compute the confidence score $s_c(b_i^{(j)})$, actionness score $s_{ac}(b_i^{(j)})$ and background score $s_{bg}(b_i^{(j)})$ for each box $b_i^{(j)}$, where T is the length of the video and N_{b_i} is the number of predicted boxes in frame i. After that, we could 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. The details are as follows.

1) Action Path Linking: In order to link frame-level boxes into the 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:

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

$S(p)$ will be high for path if the corresponding detection box is assigned with a high confidence score and overlap. λ_0 is a trade-off factor to balance these two terms.

Maximizing Eqn. 2 helps find paths whose detection box scores are high and consecutive detection boxes significantly overlap in spatial and temporal domain.

To solve $\hat{p}_c = \underset{p_c}{argmax} S(p_c)$, we employ the Viterbi algorithm [46]. Once the optimal path is calculated, we remove the bounding boxes in previous path from the frames to construct the next path until a certain frame does not contain any boxes.

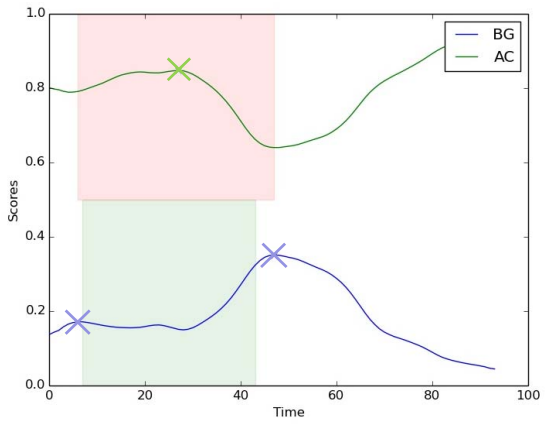


Fig. 4. Illustration of the proposed peak trimming method on one UCF-101 videos: Blue and green curves represent the background and actionness scores, respectively. Blue and green crosses denote score peaks. Green patches represent the ground-truth paths and red patches represent paths that are extracted 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 described in the last subsection span the entire video since it greedily optimizes all confidence scores across the paths. On the other hand, human actions typically take up a fraction for untrimmed video. Therefore, It is necessary to perform trimming for removing those boxes that are unlikely belong to the action regions. Mathematically, each box b_t in a path p is assigned by a binary label $y_t \in \{0, 1\}$ (where "zero" and "one" represent the "background" and "action" class respectively). With such a scheme, the boxes are near to (or far from) the valid action regions which should be assigned to the "action" (or "background") class as much as possible in the final path labeling $\hat{Y}_p = [\hat{y}_0, \hat{y}_1, \dots, \hat{y}_T]$.

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

As illustrated in Fig.4, we propose a new method by looking at the transition pattern in the actionness and background scores for the path trimming. We first smooth the scores by adopting averaging approach to reduce the influence of noisy classifier scores, and then detect all the peaks in both scores. Note that, a peak is defined as a local maximum among at least n neighbors:

$$\begin{aligned} peaks_{ac} &= \{t, s_{ac}(b_t) = \max(V_n^{(ac)}(t))\} \\ peaks_{bg} &= \{t, s_{bg}(b_t) = \max(V_n^{(bg)}(t))\} \end{aligned} \tag{3}$$

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

Once the peaks have been found, 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 $subseq$ as a set of trimmed paths.

$subseq = \emptyset$
for $p \in peaks_{ac}$ **do**
  $s = \max(peaks_{bg} < p)$
  $e = \min(peaks_{bg} > p)$
  add path $\{b_s \dots b_e\}$ into $subseq$
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 used CNN architecture for feature extraction in *YoTube* is similar to that used in [31]. In details, it consists of 24 convolution layers and 2 fully-connected layers. We firstly replace the last two fully connected layers with a locally connected layer consisting of 256 filters with a 3×3 kernel. On the top of the locally connected layer, we train an LSTM layer with 588 neurons to directly regress the coordinates of the bounding boxes. We employ a locally connected layer to stabilize the training process and improve the convergence. The number of neurons in the last layer means dividing the image into 7×7 grids of which each predicts 2 bounding boxes.

For the RGB stream, the convolutional part of our model is pretrained on the ImageNet 1000-class dataset [60]. 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 [61]. We found no problem in convergence by initializing the weights of our Flow model with the weights of the RGB model despite of the notable difference between the images distribution.

We perform data augmentation to prevent over-fitting. This part is non-negligible due to important correlation among frames of the same video. Besides mirroring, we use corner cropping and center cropping. In details, 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. Such an operation permits increasing the size of the dataset by a factor of 12.

We use the Adam [62] optimizer for training. When training the static *YoTube* detector, we use a batch size of 32 frames from different videos during 100 epochs with an initial learning rate of 10^{-4}. After 20 epochs, the learning rate decay by 10^{-5}. When training the recurrent *YoTube*, we freeze the weighs of the convolutional layers to avoid a catastrophic forgetting. We set the batch size to 10 sequences and each sequence consists of 10 frames from different videos during 50 epochs. The same learning rate scheduling is used.

B. Datasets

1) UCF-101: The UCF-101 dataset is a large action recognition dataset which contains 101 action categories with more than 13,000 videos and each video contains about 180 frames. In our experiments, we use a subset which consists of 3,204 videos over 24 categories for the localization task. About 25% videos have been untrimmed, which permits to validate the efficiency of our methods of trimming videos. Each video contains one or more instances of the same action class. It has large variations in terms of appearance, scale, motion, etc with much diversity in terms of actions. Three default training/testing splits are provided with the dataset, and we perform experiments on the first split which consists of 2,290 training videos and 914 testing videos.

2) UCF-Sports: The dataset contains 150 sport broadcast videos with realistic actions captured under dynamic and cluttered environments. The dataset considers 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 training-testing split suggested in [44], where the training and testing partition consist of 103 and 47 videos, respectively. The ground truth is provided as the sequences of bounding boxes enclose the actions.

3) JHMDB: This dataset consists of 928 videos for 21 different actions such as brush hair, swing baseball or jump. Video clips are trimmed to the duration of the action. Each clip contains between 15 and 40 frames. There are 3 training/testing splits and evaluation averages the results over the three splits.

C. Evaluation Metrics

1) ABO, MABO: We use two popular metrics as in [44] to report the overall performance, namely Average Best Overlap (ABO) and Mean ABO (MABO). The overlap (OV) between a path $\mathbf{d} = \{d_s \dots d_e\}$ and a ground truth path $\mathbf{g} = \{g_s \dots g_e\}$ is defined as follows:

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

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

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

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 $D = \{d_j | j = 1 \dots m\}$ for the 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.

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

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

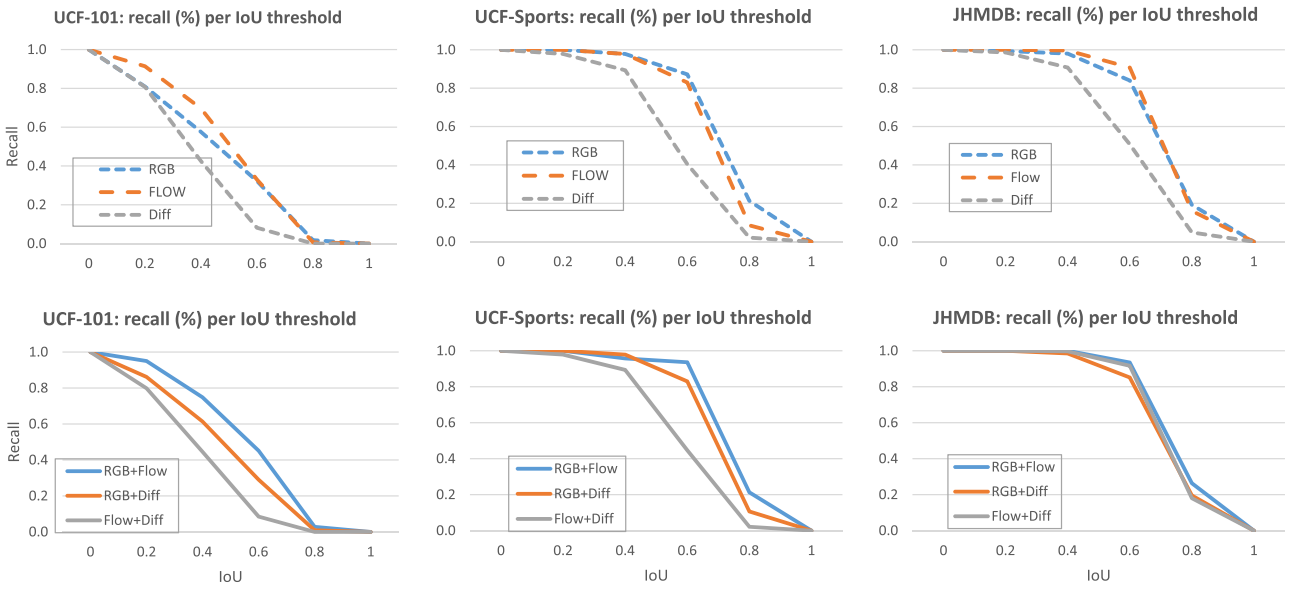


Fig. 5. Comparison on using RGB, Flow, Frame Difference and their combinations as input. From left to right column, the results are on the UCF-101, UCF-Sports, and the JHMDB dataset.

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

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.

2) Recall Vs. IoU: Another popular metric is the Recall vs. IoU [28], 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 the threshold η *i.e.*, $OV(d_j, g_i) \geq \eta$ and $\eta \in [0, 1]$. In our work, we aim to maximize the recall at the threshold of 0.5 like [44] and [48].

3) Precision Vs Recall: The precision-recall is a metric commonly used for action detection. In experiments, we also evaluate the precision-recall curve for action proposal to reflect a detector's tradeoff between precision and recall. The recall is similarly defined as in Recall vs IoU. The Precision describes how many detected actions are matched with respect to the total number of detected tubes.

V. Experimental Results

Our experiments could be divided into following parts: component analysis, generalization analysis, parameter sensitivity analysis, run-time analysis and comparison with other methods.

A. Component Analysis

1) RGB Vs. Flow Vs. Frame Difference: In recent action analysis studies, RGB and Flow have shown their complementary role in achieving state-of-the-art performance [11]. Their success could attribute to that RGB conveys rich object information and scene context and the Flow images capture salient object motions. Furthermore, in surveillance, frame difference has also been used as a kind of inputs to speed-up analysis thanks to its computation efficiency. Fig. 5 demonstrates the performance of our method with different inputs. One can see that on UCF-101 and JHMDB, our method using flow image achieves better performance than the case of RGB inputs. One underlying reason is that Flow image could eliminate the influence of the background clutter in RGB. Moreover, we found in UCF-Sports, using RGB images achieves a better result than using Flow images, which is resulted from the contents of dataset. In words, UCF-Sports contains many videos with salient actors, hence the information from RGB images is discriminative enough.

The performance of using Frame difference is relatively lower than that of using RGB and Flow, a possible explanation is that both UCF-Sports and UCF-101 contain background clutters/motion, simply calculating the difference between frames will result in noisy responses from background.

2) AlexNet Vs GoogleNet Vs C3D: Our method is compatible with recent popular Convolutional Neural Networks such as AlexNet [63], VGG [64], GoogleNet [65] and C3D [10] for feature extraction. In LRCN, Donahue *et al.* [23] apply CaffeNet (a variant of AlexNet) for feature extraction. In our experiment, we choose GoogleNet [65] as our base feature extractor to balance speed and accuracy since it gives comparable performance with VGG [64], while using a smaller number of parameters. On the counter-part, the C3D structure [10] has shown good performance in action recognition and temporal action detection by learning 3D convolutional filter. Hence, we conduct experiments using AlexNet, GoogleNet and C3D for feature extraction.

The comparison result is shown in Fig.6. The performance of GoogleNet is about 3% and 6% higher than C3D and AlexNet respectively. There are two possible reasons for the performance gain: 1) the GoogleNet is deeper than the C3D and AlexNet, which can help extract more discriminative features; 2) our GoogleNet is trained with an input image

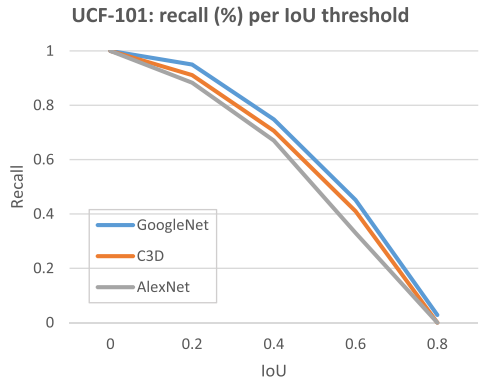


Fig. 6. Comparison by using AlexNet, GoogleNet and C3D for feature extraction.

TABLE I
COMPONENT ANALYSIS ON THE UCF-101 DATASET

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 (Trim with [19])	47.02	46.89	54.30	75
YoTube-RGB+FLOW	**52.45**	**52.92**	**59.19**	73

TABLE II
COMPONENT ANALYSIS ON THE UCF-SPORTS DATASET

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 (Trim with [19])	**74.44**	**75.31**	**97.87**	30
YoTube-RGB+FLOW	**74.44**	**75.31**	**97.87**	30

of size 448 × 448, while the input to C3D and AlexNet is only 160 × 160 and 224 × 224 respectively. The low-resolution input to C3D and AlexNet is less desirable for detection related tasks. Actually, we would point out that our architecture could further enjoy the progress of deep neural networks.

3) Recurrent YoTube Vs Static YoTube: The component analysis between recurrent and static *YoTube* for two streams in UCF-101, UCF-Sports and JHMDB are shown in Fig. 7, Table I, II and III. 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 about 3.56% improvement as shown in Table I. These results prove that the recurrent model and static model are complementary. We conjecture this since the RNN could capture 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. The result 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 proposed model (YoTube (RGB+FLOW)) combines two streams, which achieves 8% improvement than the flow stream in recall. The result demonstrates that the RGB and Flow streams also complement each other.

TABLE III
COMPONENT ANALYSIS ON THE JHMDB DATASET

JHMDB	ABO	MABO	Recall	#Prop.
RGB Stream				
Static (S)	68.36	67.55	92.34	**20**
Recurrent (R)	56.43	56.07	91.07	**20**
YoTube (RGB)	**70.52**	**69.72**	**94.03**	30
Flow Stream				
Static (S)	70.46	70.08	96.81	**20**
Recurrent (R)	71.76	71.36	96.17	**20**
YoTube (FLOW)	**72.55**	**72.17**	**97.65**	30
Ensemble				
YoTube-RGB+FLOW (NO TRIM)	**74.72**	**74.21**	**99.31**	30
YoTube-RGB+FLOW (Trim with [19])	73.41	72.65	97.08	35
YoTube-RGB+FLOW	**74.72**	**74.21**	**99.31**	30

For UCF-Sports dataset, the static version outperforms the recurrent version by 4% (see Table II), whereas the ensemble model performs similar to the static version in terms of recall, but achieves higher ABO and MABO for better localization. For the flow stream, the static version gives a performance gain of 2.2% over the recurrent version in terms of recall, and the ensemble model is 2% better than the static version.

For JHMDB dataset, the static version is 12%, 11% and 1.3% better than the recurrent version in RGB stream in terms of the ABO, MABO and Recall metrics. The combination of static and recurrent version yields a further 2% performance improvement. In Flow stream, the recurrent version is about 1.3% better than the static version in terms of ABO and MABO, their combination yields around 1.5% improvement. Further ensemble yields a recall of 99.31%. This confirms the complementariness of recurrent and static networks using RGB and Flow information.

4) Comparison Between Different Path Trimming Methods: We also compare our path trimming method with the method proposed in our conference work [19], one can observe that the new method is 5% and 2% better than [19] on UCF-101 and JHMDB with less amount of tubes in (see Tables I–III). Our method and [19] have the identical result in UCF-Sports in Table II, because the classifier scores in UCF-Sports is strong enough to indicate the start and end of the video.

Moreover, we investigate the performance of the model without path trimming (NO TRIM) in Table I, II and III. 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 and JHMDB dataset which consists of trimmed videos, the method with and without path trimming

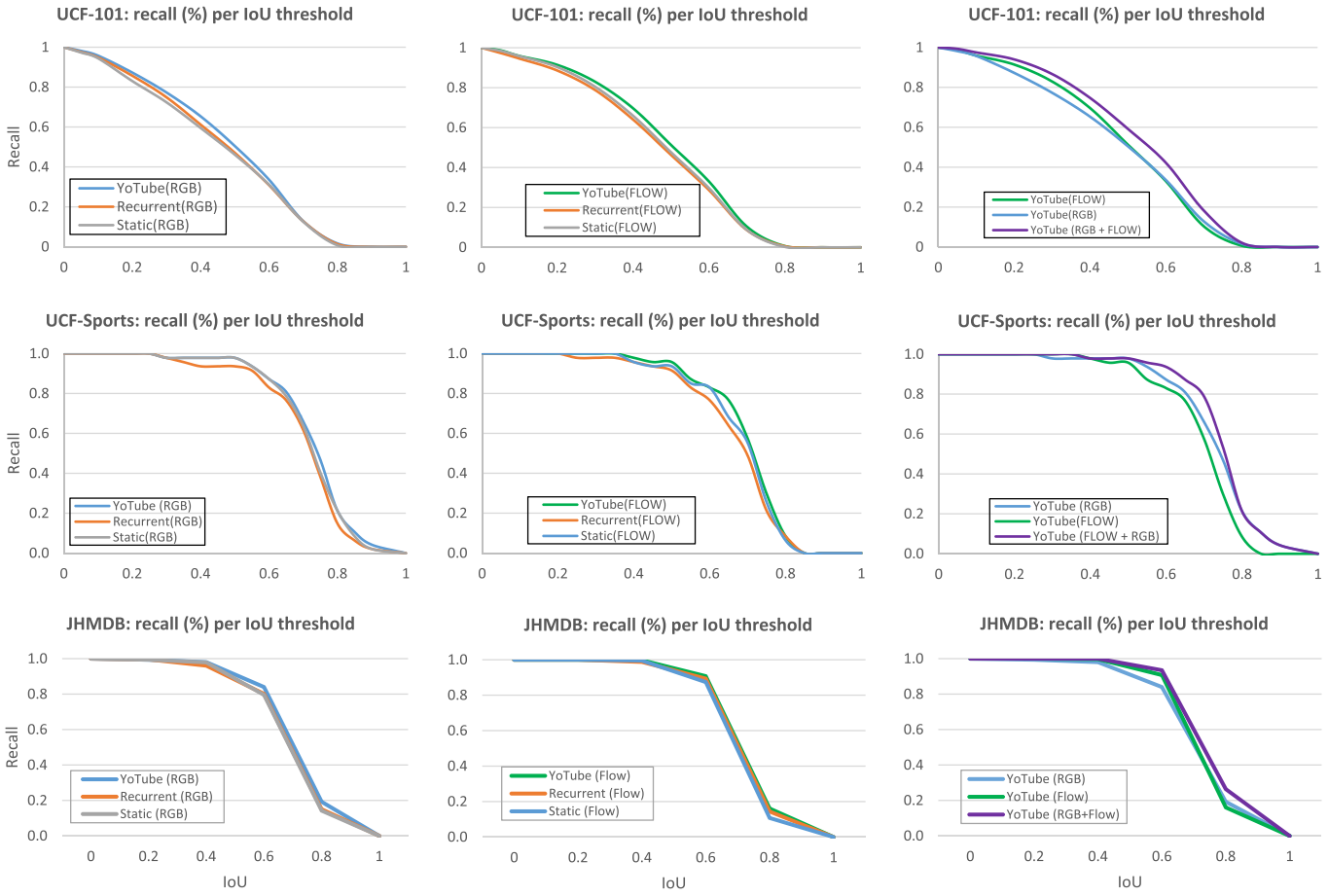


Fig. 7. Comparison between two stream's static YoTube and recurrent YoTube on the UCF-101 (top-row), UCF-Sports (middle row) and JHMDB (bottom row); left column shows RGB stream, middle column shows the Flow stream and right column shows their ensemble.

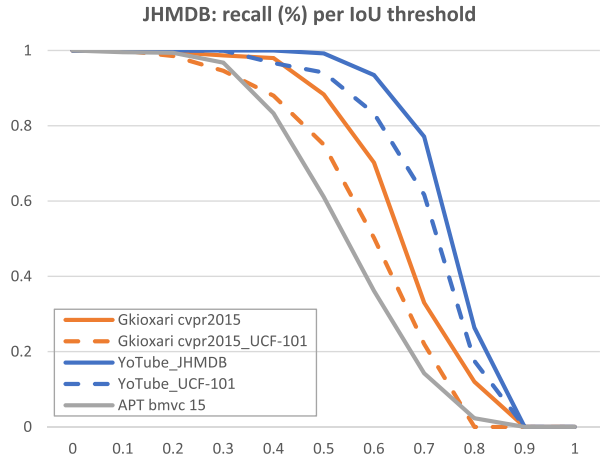


Fig. 8. Generalization Analysis: the models trained on UCF-101 is tested on JHMDB to demonstrate generalization capability. The dashed lines represent the results using the models trained on UCF-101. The solid lines represent the results trained in JHMDB.

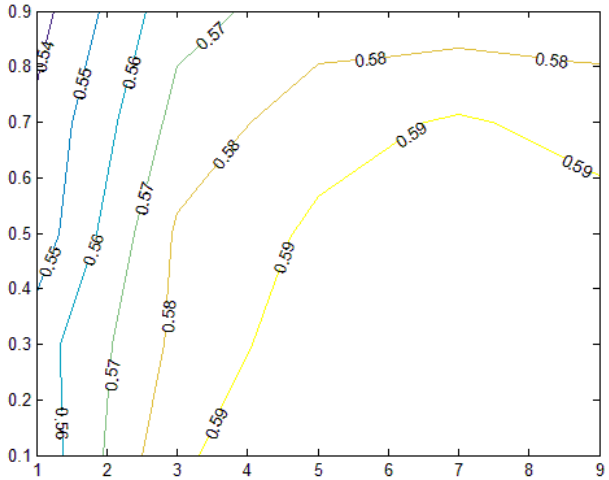


Fig. 9. Parameter sensitivity test: the magnitude of the contour map is the recall rate in UCF-101, the x-axis is the test value of λ_{obj} and y-axis is the test range of λ_{noobj}.

achieve the same result. This proves that our method is adaptive to the video content.

5) Run Time Analysis: Finally, we compared the running speed of our method with the APT [44] and Gkioxari [46]. Our method runs at 20 FPS, which is 15x faster than APT and 60x faster than Gkioxari. Our path linking and trimming method runs at 0.01s and is 20x faster than [19], which is quite computational efficient.

B. Generalization Analysis

We test the generalization ability of our method by pre-training the models on UCF-101 and transferring on JHMDB.

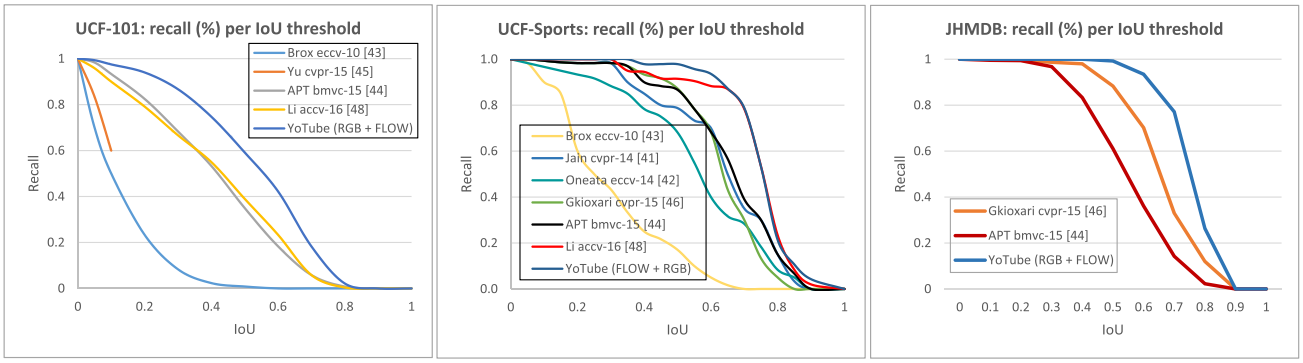


Fig. 10. Comparison with some state of the arts on UCF-101, UCF-Sports and JHMDB dataset in terms of recall with different IoU thresholds.

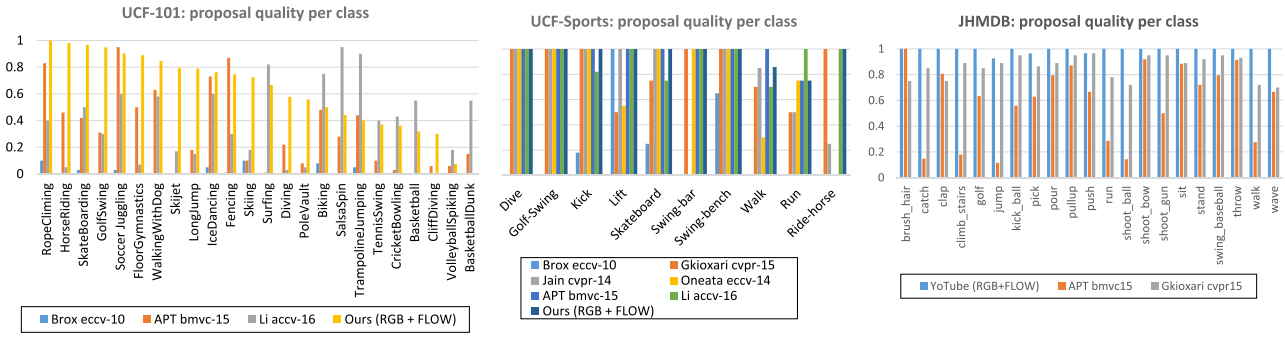


Fig. 11. Comparison with some state-of-the-art methods on UCF-101, UCF-Sports and JHMDB dataset. The performance is measured by the recall on each action class.

Note that, there are little overlap between these two datasets. The corresponding result is shown in Fig.8. One can observe that our model and [46] pre-trained on UCF-101 experience nearly 15% performance degradation when the JHMDB dataset is used for evaluation. Further analysis reveals that the loss mainly results from the flow stream. One possible reason is that the appearance of flow images in UCF-101 is corrupted by noises, which is difficult to generalize to JHMDB as it consists of small motions.

TABLE IV

QUANTITATIVE COMPARISON ON THE UCF-101 DATASET. RECALL IS COMPUTED AT AN IOU THRESHOLD OF 0.5

UCF101	ABO	MABO	Recall	#Prop.
Brox & Malik [43]	13.28	12.82	1.40	**3**
Yu *et al.* [45]	n.a	n.a	0.0	10,000
APT [44]	40.77	39.97	35.45	2299
Li *et al.* [48]	**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

C. Parameter Sensitivity Test

The hyper-parameters λ_{obj} and λ_{noobj} in Eqn.1 balance the role of objects and no-objects, whose values are empirically chosen. In experiment, we investigate the influence of one of parameters by fixing the other parameters. The sensitivity testing result is shown in Fig.8. One can observe that our method has a relative stable performance when $\lambda_{coord} \in \{5, 9\}$ and $\lambda_{noobj} \in \{0.1, 0.5\}$. One explanation for such parameter selection is that as one image typically contains a few action regions, hence the bounding boxes from background should be with small weights.

D. Comparison to State-of-the-Arts

We compare our method with state of the arts on UCF-101, UCF-Sports and JHMDB datasets. The recall-vs-IoU and recall-per-class curves for the testing datasets are shown in Fig. 10 and Fig. 11, respectively.

For the UCF-101 dataset, our method outperforms the state of the art [48] by at least 20% in all the range of IoU. Although Li *et al.* [48] proposed using RPN [29], their model only involves one stream and the achieved performance is only 4% better than the unsupervised method (APT [44]) in terms of the recall as shown in Table IV. Notwithstanding, the recall of our single stream based method in RGB and Flow remarkably outperforms [48] by 11% and 12%, respectively. The result shows the superiority of the proposed *YoTube* which employs spatial-temporal modeling and two-stream design. Reference [45] is a low-level features based human detector and the experimental result shows that it cannot effectively handle large dynamic changes in the scene. The work in [43] is also a low-level features based method, which is designed for non-overlap segmentation. These two characteristics make it sub-optimal for the task and achieving the lowest recall. According to the per-class recall curve in Fig. 11, our method is better than Li *et al.* [48] in many cases, especially when the data set contains large motion changes (e.g. "skijet", "floor gymnastics" and "long jump").

For the UCF-Sports dataset, our method also outperforms Li *et al.* [48] by nearly 6% in terms of recall in Table V. Moreover, the deep learning based approaches (our method

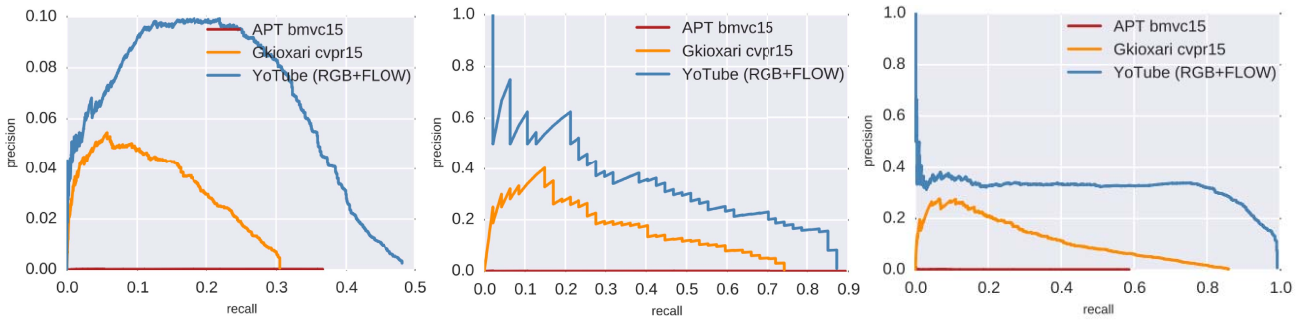


Fig. 12. Comparison with some state-of-the-arts methods on UCF-101, UCF-Sports and JHMDB dataset. The performance is measured by the precision vs. recall.

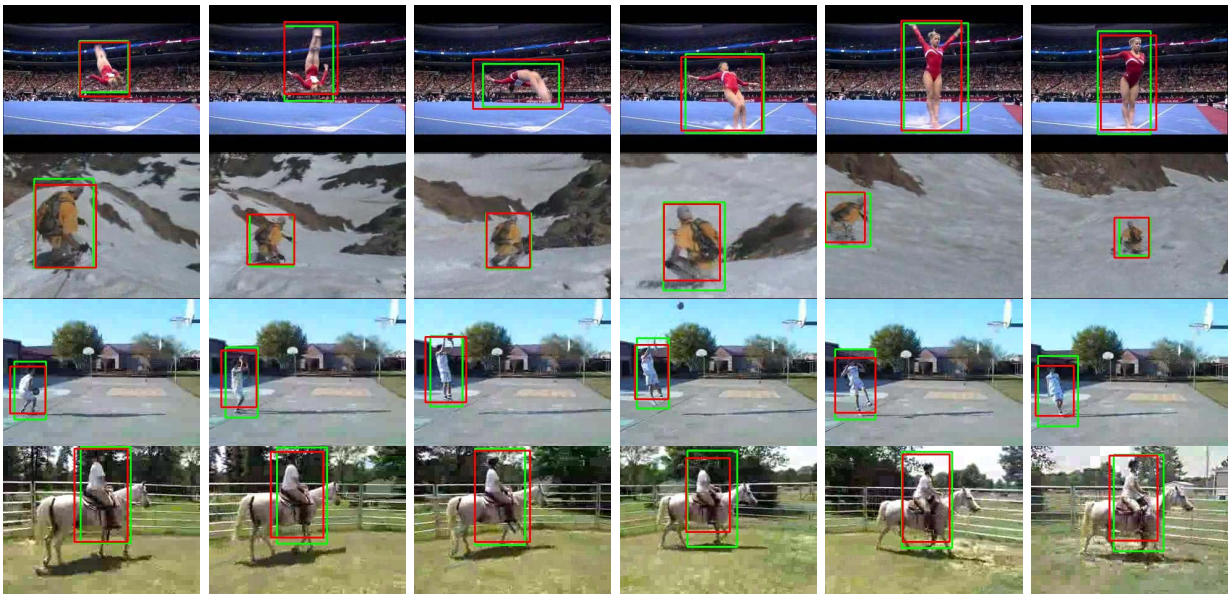


Fig. 13. Examples of our method on 4 videos from UCF101. Green boxes are ground truth and red boxes are from the best predicted path. Best viewed in colors.

TABLE V

QUANTITATIVE COMPARISON ON THE UCF-SPORTS DATASET. RECALL IS COMPUTED AT AN IOU THRESHOLD OF 0.5

UCF Sports	ABO	MABO	Recall	#Prop.
Brox & Malik [43]	29.84	30.90	17.02	**4**
Jain *et al.* [41]	63.41	62.71	78.72	1642
Oneata *et al.* [42]	56.49	55.58	68.09	3000
Gkioxari *et al.* [46]	63.07	62.09	87.23	100
APT [44]	65.73	64.21	89.36	1449
Li *et al.* [48]	**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 VI

QUANTITATIVE COMPARISON ON THE JHMDB DATASET. RECALL IS COMPUTED AT AN IOU THRESHOLD OF 0.5

JHMDB	ABO	MABO	Recall	#Prop.
Gkioxari *et al.* [46]	63.07	62.09	86.34	125
APT [44]	54.16	53.37	59.55	2400
YoTube-RGB	70.52	69.72	94.03	30
YoTube-FLOW	72.55	72.17	97.65	30
YoTube-RGB+FLOW	**74.72**	**74.21**	**99.31**	30

and [48]) also remarkably outperform the unsupervised methods, which shows the effectiveness of the deep networks based methods. The per-class recall curve for all methods is also provided in Fig. 11.

For the JHMDB dataset, Table VI shows that our method outperforms [44] and [46] by nearly 13% and 40% in terms of recall. For a comprehensive evaluation, we also report the Recall vs. Precision Curve in Fig.12. The unsupervised method [44] achieves a relative low precision because they generate many tubes by using low-level cues. Our method performs better than Gkioxari and Malik [46] since we simultaneously consider the appearance and temporal contexts.

We also evaluate our methods in terms of other metrics, *i.e.*, ABO, MABO, and number of proposals. The results are shown in Table IV, V and VI. From the results, our method produces the highest MABO. Note that, although Li *et al.* [48] achieve the highest ABO, there are big differences between ABO and MABO. Actually, these two measurements should

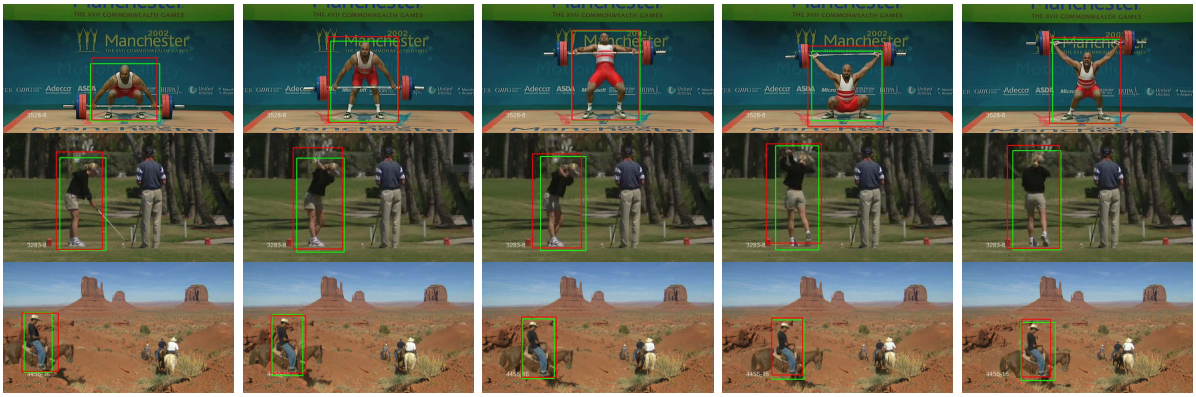


Fig. 14. Examples of our method on 3 videos from UCF Sports. Green boxes are ground truth and red boxes are from the best predicted path. Best viewed in colors.

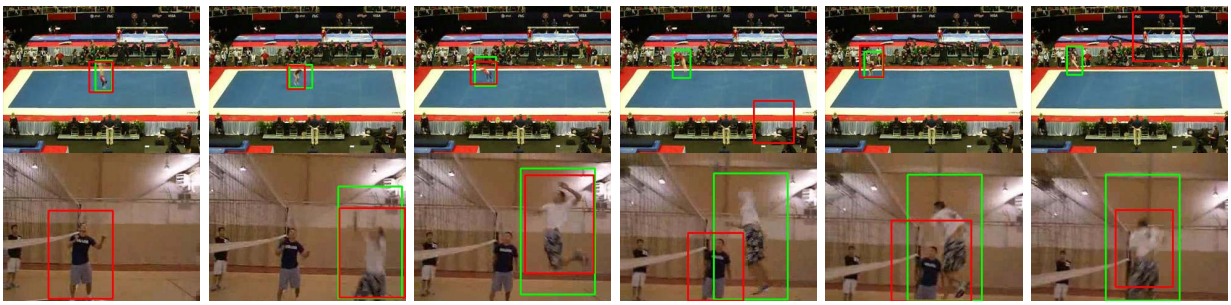


Fig. 15. Mistakes of our method on 3 videos. Green boxes are ground truth and red boxes are from the best predicted path. Best viewed in colors.

be in the similar scale according to the formula in Sec. IV-C, i.e. MABO is the mean of ABO for all classes.

Some visual examples are shown in Fig.13 and 14. The illustrations demonstrate that our method can produce candidate paths which produce good localization of the human actions.

E. Discussion

The proposed method has some failed cases as follows. First, it may loss frame-level localization for medium- and small-scale objects, especially in cluttered environments and the video with severe motion blurs (e.g. row 1 and 2 of Fig. 15). One possible reason is that our method performs inference on a dense fully connected layer which cannot capture the small scale changes due to its relatively coarse receptive field.

Another failed case is that the localization of the generated action proposal might be interfered when the detection quality is unsatisfactory or objects have large overlap (see Fig. 15, row 2). The reason is that the path is generated by the greedy dynamic programming which accumulates errors during optimization. How to improve the robustness against these failed cases will be explored in our future work.

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 approach enjoys the regression capability of RNN and representation learning capability of CNN, thus producing frame-level candidate action boxes jointly using RGB, Flow and temporal contexts among frames. The action proposals are constructed using dynamic programming with a novel path trimming method. Incorporating the long-term temporal context from LSTM helps reduce the ambiguities in each single frame. The proposed path trimming method can help trim the path for untrimmed videos. The superior results on UCF-101, UCF-Sports and JHMDB datasets highlight the effectiveness of our framework.

In the future, we plan to investigate how to use network compression technique to prune the redundant parameters so that the method could run an on-board device such as drone. In addition, we will also explore how to use a deeper network such as ResNet [66] for higher accuracy.

Acknowledgment

This work was mainly done when R. Vial was interned with the Institute for Infocomm Research.

References

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

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

[3] 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 Process.*, vol. 24, no. 12, pp. 5739–5752, Dec. 2015.

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

[5] K. R. Jerripothula, J. Cai, and J. Yuan, "CATS: Co-saliency activated tracklet selection for video co-localization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 187–202.

[6] K. Kang, W. Ouyang, H. Li, and X. Wang, "Object detection from video tubelets with convolutional neural networks," in *Proc. CVPR*, Jun. 2016, pp. 817–825.

[7] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov, "Exploiting image-trained CNN architectures for unconstrained video classification," in *Proc. BMVC*, 2015, p. 1.

[8] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative CNN video representation for event detection," in *Proc. CVPR*, Jun. 2015, pp. 1798–1807.

[9] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. CVPR*, Jun. 2014, pp. 1725–1732.

[10] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. ICCV*, Dec. 2015, pp. 4489–4497.

[11] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. NIPS*, 2014, pp. 568–576.

[12] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. CVPR*, Jun. 2015, pp. 4305–4314.

[13] J. Y.-H. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. CVPR*, Jun. 2015, pp. 4694–4702.

[14] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *Proc. ICCV*, Dec. 2015, pp. 3164–3172.

[15] S. Saha, G. Singh, M. Sapienza, P. H. S. Torr, and F. Cuzzolin, "Deep learning for detecting multiple space-time action tubes in videos," in *Proc. BMVC*, 2016, p. 1.

[16] X. Peng and C. Schmid, "Multi-region two-stream R-CNN for action detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 744–759.

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

[18] S. Manen, M. Guillaumin, and L. van Gool, "Prime object proposals with randomized Prim's algorithm," in *Proc. ICCV*, Dec. 2013, pp. 2536–2543.

[19] R. Vial, H. Zhu, Y. Tian, and S. Lu, "Search video action proposal with recurrent and static YOLO," in *Proc. ICIP*, 2017, pp. 1–5.

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

[21] F. Wang and D. M. J. Tax. (2016). "Survey on the attention based RNN model and its applications in computer vision." [Online]. Available: https://arxiv.org/abs/1601.06823

[22] V. Veeriah, N. Zhuang, and G.-J. Qi, "Differential recurrent neural networks for action recognition," in *Proc. ICCV*, 2015, pp. 1–9.

[23] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. CVPR*, 2015, pp. 1–10.

[24] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, and X. Xue, "Modeling spatial-temporal clues in a hybrid deep learning framework for video classification," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 461–470.

[25] Z. Wu, Y.-G. Jiang, X. Wang, H. Ye, and X. Xue, "Multi-stream multi-class fusion of deep networks for video classification," in *Proc. ACM Multimedia Conf.*, 2016, pp. 791–800.

[26] B. Ni, X. Yang, and S. Gao, "Progressively parsing interactional objects for fine grained action detection," in *Proc. CVPR*, Jun. 2016, pp. 1020–1028.

[27] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proc. CVPR*, Jun. 2016, pp. 2325–2333.

[28] C. L. Zitnick and P. Dollar, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 391–405.

[29] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. NIPS*, 2015, pp. 91–99.

[30] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 21–37.

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. CVPR*, Jun. 2016, pp. 779–788.

[32] S. Herath, M. T. Harandi, and F. Porikli. (2016). "Going deeper into action recognition: A survey." [Online]. Available: https://arxiv.org/abs/1605.04988

[33] Z. Wu, T. Yao, Y. Fu, and Y.-G. Jiang. (2016). "Deep learning for video classification and captioning." [Online]. Available: https://arxiv.org/abs/1609.06782

[34] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," in *Proc. ICML*, 2010, pp. 1–14.

[35] H. Xu, A. Das, and K. Saenko. (2017). "R-C3D: Region convolutional 3D network for temporal activity detection." [Online]. Available: https://arxiv.org/abs/1703.07814

[36] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage CNNs," in *Proc. CVPR*, Jun. 2016, pp. 1049–1058.

[37] Y. Ke, R. Sukthankar, and M. Hebert, "Efficient visual event detection using volumetric features," in *Proc. ICCV*, Oct. 2005, pp. 166–173.

[38] T. Lan, Y. Wang, and G. Mori, "Discriminative figure-centric models for joint action localization and recognition," in *Proc. ICCV*, Nov. 2011, pp. 2003–2010.

[39] Y. Tian, R. Sukthankar, and M. Shah, "Spatiotemporal deformable part models for action detection," in *Proc. CVPR*, Jun. 2013, pp. 2642–2649.

[40] D. Tran and J. Yuan, "Max-margin structured output regression for spatio-temporal action localization," in *Proc. NIPS*, 2012, pp. 350–358.

[41] M. Jain, J. C. van Gemert, H. Jégou, P. Bouthemy, and C. G. M. Snoek, "Action localization with tubelets from motion," in *Proc. CVPR*, Jun. 2014, pp. 740–747.

[42] D. Oneata, J. Revaud, J. Verbeek, and C. Schmid, "Spatio-temporal object detection proposals," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2014, pp. 737–752.

[43] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 282–295.

[44] J. C. van Gemert, M. Jain, E. Gati, and C. G. M. Snoek, "APT: Action localization proposals from dense trajectories," in *Proc. BMVC*, 2015, pp. 1–12.

[45] G. Yu and J. Yuan, "Fast action proposals for human action detection and search," in *Proc. CVPR*, Jun. 2015, pp. 1302–1311.

[46] G. Gkioxari and J. Malik, "Finding action tubes," in *Proc. CVPR*, Jun. 2015, pp. 759–768.

[47] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, Jun. 2014, pp. 580–587.

[48] N. Li, D. Xu, Z. Ying, Z. Li, and G. Li, "Searching action proposals via spatial actionness estimation and temporal path inference and tracking," in *Proc. Asian Conf. Comput. Vis.*, 2016, pp. 384–399.

[49] T. Baltrušaitis, C. Ahuja, and L.-P. Morency. (2017)."Multimodal machine learning: A survey and taxonomy." [Online]. Available: https://arxiv.org/abs/1705.09406

[50] Y. Li, M. Yang, and Z. Zhang. (2016). "Multi-view representation learning: A survey from shallow methods to deep methods." [Online]. Available: https://arxiv.org/abs/1610.01206

[51] J. Wagner, V. Fischer, M. Herman, and S. Behnke, "Multispectral pedestrian detection using deep fusion convolutional neural networks," in *Proc. Eur. Symp. Artif. Neural Netw., Comput. Intell. Mach. Learn.*, 2016, pp. 1–6.

[52] L. Qu, S. He, J. Zhang, J. Tian, Y. Tang, and Q. Yang, "RGBD salient object detection via deep fusion," *IEEE Trans. Image Process.*, vol. 26, no. 5, pp. 2274–2285, May 2017.

[53] A. Wang, J. Lu, J. Cai, G. Wang, and T.-J. Cham, "Unsupervised joint feature learning and encoding for RGB-D scene labeling," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4459–4473, Nov. 2015. [Online]. Available: https://doi.org/10.1109/TIP.2015.2465133

[54] A. Wang, J. Lu, J. Cai, T.-J. Cham, and G. Wang, "Large-margin multi-modal deep learning for RGB-D object recognition," *IEEE Trans. Multimedia*, vol. 17, no. 11, pp. 1887–1898, Nov. 2015.

[55] H. Zhu, J.-B. Weibel, and S. Lu, "Discriminative multi-modal feature fusion for RGBD indoor scene recognition," in *Proc. CVPR*, Jun. 2016, pp. 2969–2976.

[56] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal residual networks for video action recognition," in *Proc. NIPS*, 2016, pp. 3468–3476.

[57] X. Zhang *et al.*, "Deep fusion of multiple semantic cues for complex event recognition," *IEEE Trans. Image Process.*, vol. 25, no. 3, pp. 1033–1046, Mar. 2016. [Online]. Available: https://doi.org/10.1109/TIP.2015.2511585

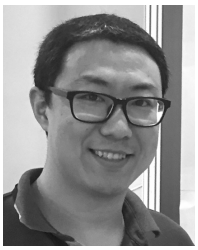
[58] X. Zhang, K. Gao, Y. Zhang, D. Zhang, J. Li, and Q. Tian, "Task-driven dynamic fusion: Reducing ambiguity in video description," in *Proc. CVPR*, Jul. 2017, pp. 6250–6258.
[59] N. D. Doulamis and A. D. Doulamis, "Fast and adaptive deep fusion learning for detecting visual objects," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2012, pp. 345–354.
[60] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[61] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. ICCV*, Dec. 2015, pp. 1026–1034.
[62] D. P. Kingma and J. Ba. (2014). "Adam: A method for stochastic optimization." [Online]. Available: https://arxiv.org/abs/1412.6980
[63] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[64] K. Simonyan and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[65] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. CVPR*, Jun. 2015, pp. 1–9.
[66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Jun. 2016, pp. 770–778.

Hongyuan Zhu (S'13–M'14) received the B.S. degree in software engineering from the University of Macau, Macau, China, in 2010, and the Ph.D. degree in computer engineering from Nanyang Technological University, Singapore, in 2014. He is currently a Research Scientist with the Institute for Infocomm Research, A*STAR, Singapore. His research interests include multimedia content analysis and segmentation, specially image segmentation/cosegmentation, object detection, scene recognition, and saliency detection.

Romain Vial received the double degree in MVA Master Programme with ENS Paris Saclay, with a focus on machine learning and computer vision. He is currently pursuing the M.Sc. degree in applied probabilities with MINES ParisTech. His research interests include the use of machine learning and deep learning to process multimodal data, such as text and images.

Shijian Lu received the Ph.D. degree in electrical and computer engineering from the National University of Singapore. He is currently an Assistant Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His major research interests include image and video analytics, visual intelligence, and machine learning. He has co-authored over 10 patents in these research areas. He has published over 100 international refereed journal and conference papers. He is a Senior Program Committee Member of the International Joint Conference on Artificial Intelligence in 2018. He is the Area Chair of the International Conference on Pattern Recognition in 2016, the International Conference on Document Analysis and Recognition in 2017, and the IEEE Winter Conference on Applications of Computer Vision in 2017.

Xi Peng received the B.Eng. degree in electronic engineering and the M.Eng. degree in computer science from the Chongqing University of Posts and Telecommunications, China, in 2005 and 2008, respectively, and the Ph.D. degree in computer science from Sichuan University, China, in 2013. From 2014 to 2017, he was a Research Scientist with the Institute for Infocomm, Agency for Science, Technology and Research, Singapore. He is currently a Research Professor with the College of Computer Science, Sichuan University, Chengdu, China. His current research interests include machine intelligence and computer vision. He has authored over 30 articles in these areas.



Huazhu Fu received the B.S. degree in mathematical sciences from Nankai University in 2006, the M.E. degree in mechatronics engineering from the Tianjin University of Technology in 2010, and the Ph.D. degree in computer science from Tianjin University, China, in 2013. He was a Research Fellow with Nanyang Technological University, Singapore, for two years. He is currently a Research Scientist with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore. His research interests include computer vision, image processing, and medical image analysis. He is an Associate Editor of the *BMC Medical Imaging*.

Yonghong Tian (S'00–M'06–SM'10) received the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. He is currently a Full Professor with the National Engineering Laboratory for Video Technology and the Cooperative Medianet Innovation Center, School of Electronics Engineering and Computer Science, Peking University, Beijing. He has authored or co-authored over 160 technical articles in refereed journals and conferences. He has owned over 55 Chinese and U.S. patents. His research interests include machine learning, computer vision, and multimedia big data.

Xianbin Cao (M'08–SM'10) is the Dean and a Professor with the School of Electronic and Information Engineering, Beihang University, Beijing, China. His current research interests include intelligent transportation systems, airspace transportation management, and intelligent computation. He currently serves as an Associate Editor for IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING and the *Neurocomputing*.