Recurrent Tubelet Proposal and Recognition Networks for Action Detection

Dong Li[1], Zhaofan Qiu[1], Qi Dai[2], Ting Yao[3], and Tao Mei[3]

[1] University of Science and Technology of China, Hefei, China
[2] Microsoft Research, Beijing, China
[3] JD AI Research, Beijing, China
{dongli1995.ustc,zhaofanqiu,tingyao.ustc}@gmail.com
qid@microsoft.com, tmei@live.com

Abstract. Detecting actions in videos is a challenging task as video is an information intensive media with complex variations. Existing approaches predominantly generate action proposals for each individual frame or fixed-length clip independently, while overlooking temporal context across them. Such temporal contextual relations are vital for action detection as an action is by nature a sequence of movements. This motivates us to leverage the localized action proposals in previous frames when determining action regions in the current one. Specifically, we present a novel deep architecture called Recurrent Tubelet Proposal and Recognition (RTPR) networks to incorporate temporal context for action detection. The proposed RTPR consists of two correlated networks, i.e., Recurrent Tubelet Proposal (RTP) networks and Recurrent Tubelet Recognition (RTR) networks. The RTP initializes action proposals of the start frame through a Region Proposal Network and then estimates the movements of proposals in next frame in a recurrent manner. The action proposals of different frames are linked to form the tubelet proposals. The RTR capitalizes on a multi-channel architecture, where in each channel, a tubelet proposal is fed into a CNN plus LSTM to recurrently recognize action in the tubelet. We conduct extensive experiments on four benchmark datasets and demonstrate superior results over state-of-the-art methods. More remarkably, we obtain mAP of 98.6%, 81.3%, 77.9% and 22.3% with gains of 2.9%, 4.3%, 0.7% and 3.9% over the best competitors on UCF-Sports, J-HMDB, UCF-101 and AVA, respectively.

Keywords: Action Detection · Action Recognition.

1 Introduction

Action detection with accurate spatio-temporal location in videos is one of the most challenging tasks in video understanding. Compared to action recognition, this task is more difficult due to complex variations and large spatio-temporal search space. The solutions to this problem have evolved from handcrafted feature-based methods [18,34,40] to deep learning-based approaches [7]. Promising progresses have been made recently [22,28,36,39] with the prevalence of deep Convolutional Neural Networks (CNN) [10,16,30].

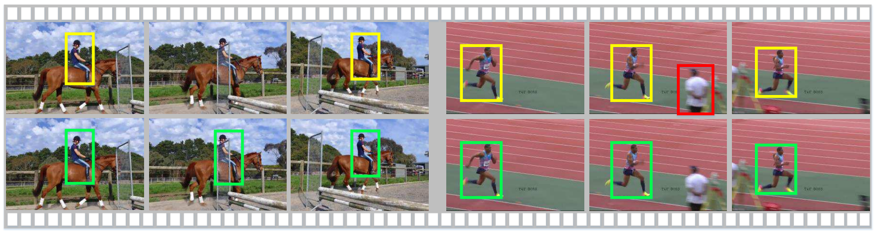


Fig. 1. Action detection comparisons on traditional method (first row) and ours (second row). Traditional method extracts per-frame proposals independently, which may result in some failures. In the example of *horse riding* (left), it fails when the person is partially blocked by the fence. In the example of *long jump* (right), an unwanted person (red bounding box) is also detected. In contrast, our approach can solve these problems by utilizing the temporal context across frames.

Inspired by the recent advances of CNN based image object detection methods [4, 6, 26], previous action detection approaches first detect either frame-level [22, 39] or clip-level proposals [11] independently. Then these fragmental proposals are associated to generate a complete action proposal by linking or tracking based approaches. However, such methods rarely exploit the temporal relations across frames or clips, which may result in unstable proposals when the single detection is unreliable. Figure 1 illustrates two examples of such limitations. In the example of *horse riding*, detection fails in the second frame where the person is partially blocked by the fence. In the other example of *long jump*, an unwanted person (red bounding box) is also detected, bringing in noises for future proposal recognition. Such noise is long-standing and inevitable due to independent detection on each frame or clip. One possible way to solve the above problems is to model the action by leveraging temporally contextual relations. For example, when the person is blocked in current frame, we could leverage the proposals in previous frames to infer the current ones. Motivated by this idea, we consider exploiting the action proposals in previous frames plus the corresponding contextual information when determining the action regions in current one, instead of detecting proposals from each frame or clip independently. Through involving the temporal context, the inevitable failures in per frame or clip proposal generation scheme could be mostly alleviated.

In this paper we present Recurrent Tubelet Proposal and Recognition (RTPR) networks — a novel architecture for action detection, as shown in Figure 2. Our proposed RTPR networks consist of two components: Recurrent Tubelet Proposal (RTP) networks and Recurrent Tubelet Recognition (RTR) networks. The RTP produces action proposals in a recurrent manner. Specifically, it initializes action proposals of the start frame through a Region Proposal Network (RPN) [26] on the feature map. Then the movement of each proposal in next frame is estimated from three inputs: feature maps of both current and next frames, and the proposal in current frame. Simultaneously, a sibling proposal classifier is utilized to infer the actionness of the proposal. To form the tubelet proposals, action proposals in two consecutive frames are linked by taking both their actionness and overlap ratio into account, followed by the temporal trimming on it. The RTR capitalizes on a multi-channel architecture for tubelet proposal recognition. For each proposal, we extract three different semantic-level features, i.e., the features on proposal-cropped image, the features with RoI pooling on proposal, and the features on whole frame. These features implicitly encode the spatial context and scene information, which enhance the recognition capability on specific categories. After that, each of them is fed into a Long Short-Term Memory (LSTM) network to model the temporal dynamics. With both RTP and RTR, our approach can generate better tubelets and boost recognition, thus leading to promising detection results as shown in Figure 1.

The main contribution of this work is the proposal of RTPR networks for addressing the issue of action detection. The solution also leads to the elegant views of what kind of temporal context should be exploited and how to model the temporal relations in a deep learning framework particularly for the task of action detection, which are problems not yet fully understood in the literature.

2 Related Work

Object detection in images is a fundamental computer vision task and has been studied in a plethora of publications. Most object detection techniques are developed based on region mechanism, such as R-CNN [5], Fast R-CNN [4], and Faster R-CNN [26]. These methods first generate a set of object proposals and then do per-proposal classification and bounding box regression. To overcome the speed limit of such two-stage frameworks, YOLO [25] and SSD [21] are proposed to directly classify and regress the anchor boxes in only one step. In our method, we utilize RPN [26] introduced by Faster R-CNN to initialize action proposals in the start frame. It outputs a set of action proposals, each with an actionness score. Besides, RoI pooling [4] is exploited to extract feature for each proposal.

Video action recognition has been extensively studied recently due to its importance in many application areas, such as video surveillance and robotics. Many progresses have been achieved by leveraging the recent advances of CNN, including discriminative feature learning (e.g., Fusion-CNN [15], C3D [35], FV-VAE [23], P3D [24]) and effective architecture design (e.g., Two-Stream CNN [29], SR-CNN [37]). In particular, LSTM is employed in several works [1,19,20,33,41] to model the long-term temporal clues in videos. In our work, we also exploit LSTM model in RTR for tubelet proposal recognition. In addition, both [37] and our RTR capitalize on a multi-channel architecture to capture different semantic-level information. However, ours uniquely designs a *human only* channel, which is potentially more effective than [37] on recognizing human-driven actions.

Video action detection aims to spatio-temporally localize a recognized action within a video. Most recent approaches rely on object detectors which are trained to discriminate human action classes at frame level, including R-CNN based methods [7, 38], Faster R-CNN based methods [22, 28], and SSD based methods [31]. Typically, 2D action regions are detected in each frame, upon which 3D action volumes are generated [2]. For example, TrackLocalization [38] tracks current proposals to obtain anchor ones in next frame by leveraging optical flow, and selects the best regions in the neighborhood of anchors using a sliding window. However, distinguishing actions from single frame could be ambiguous. To address this issue, ACT [14] takes as input a sequence of frames and outputs tube proposals instead of operating on single frames. T-CNN [11] further extends 2D Region-of-Interest pooling to 3D Tube-of-Interest (ToI) pooling with 3D convolution. It directly generates tube proposals on each fixed-length clip.

The aforementioned action detection methods treat each frame or clip independently, while ignoring the temporally contextual relations. Instead, our approach generates the tubelet proposals in a recurrent manner, which fully leverages the temporal information in videos. The most closely related work is CPLA [39], which solely relies on the detected proposals of current frame to estimate the proposal movements in the next frame. Ours is different from [39] in the way that we effectively model the temporal correlations of proposals between two consecutive frames to predict movements. Moreover, Faster R-CNN is required for each frame in [39], while our approach only runs RPN at initialization. In addition, our work also contributes by reliably capturing different semantic-level information and long-term temporal dynamics for recognition.

3 Recurrent Tubelet Proposal and Recognition Networks

In this section we present our proposed Recurrent Tubelet Proposal and Recognition (RTPR) networks for video action detection. An overview of our framework is shown in Figure 2. It consists of two main components: Recurrent Tubelet Proposal (RTP) networks and Recurrent Tubelet Recognition (RTR) networks. In RTP, we first utilize CNNs for feature extraction. The RPN is applied on the feature map of the start frame for proposal initialization. Our RTP then generates proposals of subsequent frames in a recurrent manner. In particular, given action proposals in current frame, RTP estimates the movements of them to produce proposals in the next frame. Furthermore, action proposals from consecutive frames are linked according to their actionness and overlap ratio to form video-level tubelet proposals, followed by temporal trimming on tubelets. The obtained tubelets are finally fed into RTR for recognition. RTR employs a multi-channel architecture to capture different semantic-level information, where an individual LSTM is utilized to model temporal dynamics on each channel.

3.1 Recurrent Tubelet Proposal Networks

The RTP aims to generate action proposals for all frames. Instead of producing proposals in each frame independently, we exploit the localized proposals in previous frame when determining the action regions in current one. Such scheme could help avoid the failures caused by unreliable single detection. Note that RTP only indicates the locations where an action exists irrespective of the category.

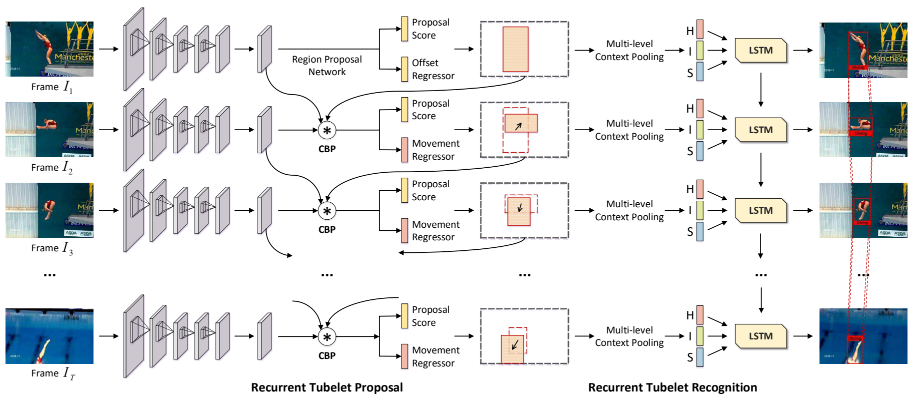


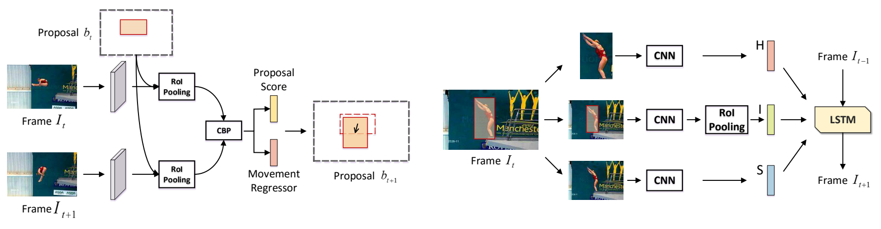
Fig. 2. The overview of RTPR networks. It consists of two components: RTP networks and RTR networks. CNNs are first utilized in RTP for feature extraction. Then RPN is applied on the start frame for proposal initialization. RTP estimates the movements of proposals in current frame to produce proposals in next frame. After proposal linking and temporal trimming, obtained tubelet proposals are fed into RTR. The RTR employs a multi-channel architecture to capture different semantic-level information, where an individual LSTM is utilized to model temporal dynamics on each channel.

Architecture. The RTP begins with the initial anchor action proposals obtained by RPN on the start frame, and then produces the action proposals of subsequent frames in a recurrent manner. Given the video frame I_t and its proposal set $B_t = \{b_t^i | i = 1, ..., N\}$ at time t, RTP aims to generate the proposal set B_{t+1} for the next frame I_{t+1}. Let $b_t^i = (x_t^i, y_t^i, w_t^i, h_t^i)$ denote the i-th proposal at time t, where x, y, w and h represent two coordinates of the proposal center, width and height of it. As shown in Figure 3(a), two consecutive frames, I_t and I_{t+1}, are first fed into a shared CNN to extract features. To predict the i-th proposal b_{t+1}^i at time t+1, we need to estimate the movement $m_{t+1}^i = (\varDelta x_{t+1}^i, \varDelta y_{t+1}^i, \varDelta w_{t+1}^i, \varDelta h_{t+1}^i)$ between b_{t+1}^i and b_t^i, which is defined as

$$\begin{aligned} \varDelta x_{t+1}^i &= (x_{t+1}^i - x_t^i)/w_t^i, & \varDelta y_{t+1}^i &= (y_{t+1}^i - y_t^i)/h_t^i, \\ \varDelta w_{t+1}^i &= \log(w_{t+1}^i / w_t^i), & \varDelta h_{t+1}^i &= \log(h_{t+1}^i / h_t^i). \end{aligned} \tag{1}$$

Instead, to estimate this movement, visual features of proposals b_{t+1}^i and b_t^i are required. It is a chicken-and-egg problem, as we do not have the exact location of b_{t+1}^i in advance. Observing the fact that the receptive fields of deep convolutional layers are generally large enough to capture possible movements, we then solve the problem by simply performing RoI pooling at the same proposal location as b_t^i, as shown in Figure 3(a).

Formally, suppose F_t^i and $F_{t+1}^i \in \mathbb{R}^{W \times H \times D}$ denote the RoI pooled features of I_t and I_{t+1} w.r.t. the same location of proposal b_t^i, where W, H and D are the width, height and channel numbers. The objective is to estimate the proposal movement m_{t+1}^i based on F_t^i and F_{t+1}^i. The movement between two consecutive



(a) RTP (b) RTR

Fig. 3. (a) RTP networks. Two consecutive frames I_t and I_{t+1} are first fed into CNNs. Given the proposal b_t in current frame I_t, we perform RoI pooling on both I_t and I_{t+1} w.r.t. the same proposal b_t. The two pooled features are fed into a CBP layer to generate the correlation features, which are used to estimate the movement of proposal b_t and the actionness score. **(b)** RTR networks. We capitalize on a multi-channel network for tubelet recognition. Three different semantic clues, i.e., *human only* (H), *human-object interaction* (I), and *scene* (S), are exploited, where the features on proposal-cropped image, the features with RoI pooling on the proposal, and the features on whole frame are extracted. Each of them is fed into an LSTM to model the temporal dynamics.

frames could be characterized by the comparison between feature maps of two frames on the same spatial region. Specifically, we adopt Compact Bilinear Pooling (CBP) [3] to capture the pairwise correlations and model spatial interactions between frames, which can be formulated with the kernelized comparison as

$$CBP(F_t^i, F_{t+1}^i) = \sum_{j=1}^{S} \sum_{k=1}^{S} \left\langle \phi(F_{t,j}^i), \phi(F_{t+1,k}^i) \right\rangle \ , \tag{2}$$

where $S = W \times H$ is the size of the feature map, $\phi(\cdot)$ is a low dimensional projection function, and $\langle \cdot \rangle$ is the second order polynomial kernel. Finally, the outputs of CBP are fed into two sibling fully connected layers. One is the regression layer that predicts the movement m_{t+1}^i, and the other is the classification layer that predicts the actionness confidence score of b_{t+1}^i.

Training Objective. When training RTP, the network inputs include two consecutive frames I_t and I_{t+1}, the proposal set B_t of I_t obtained by RPN, and the ground-truth bounding boxes $\hat{B}_{t+1}$. Assuming that the action movement across two consecutive frames is not big, those correctly extracted proposals in B_t will have large Intersection-over-Union (IoU) ratios with one ground-truth proposal in $\hat{B}_{t+1}$. Consequently, we assign a positive label to the proposals b_t^i if: (a) b_t^i has an IoU overlap higher than 0.7 with any ground-truth box in $\hat{B}_{t+1}$, or (b) b_t^i is with the highest IoU overlap with a ground-truth box in $\hat{B}_{t+1}$. Otherwise, we assign a negative label to b_t^i if its IoU ratio is lower than 0.3 with all ground-truth boxes in $\hat{B}_{t+1}$. The network is jointly optimized with classification loss $\mathcal{L}_{cls}$ and regression loss $\mathcal{L}_{reg}$. For classification, the standard log loss $\mathcal{L}_{cls}(b_t^i)$ is utilized. It outputs an actionness score in range $[0, 1]$ for the output proposal. For regression, the smooth L_1 loss [4] $\mathcal{L}_{reg}(m_{t+1}^i)$ is exploited.

It forces the proposal b_t^i to move towards a nearby ground-truth proposal in the next frame. The overall objective $\mathcal{L}$ is formulated as

$$\mathcal{L} = \frac{1}{N}\sum_i \mathcal{L}_{cls}(b_t^i) + \lambda \frac{1}{N_{reg}} \sum_i y_t^i \mathcal{L}_{reg}(m_{t+1}^i) \ , \tag{3}$$

where N is the number of proposals in B_t, N_{reg} is the number of positive proposals in B_t, λ is the parameter for balancing classification and regression, and y_t^i is an indicator that $y_t^i = 1$ if b_t^i is assigned a positive label, otherwise 0.

Prediction. Given a video, we initialize action proposals of its start frame by utilizing RPN on the feature map. Among all the proposals, we keep the top N_1 proposals $B_1 = \{b_1^i\}_{i=1}^{N_1}$ according to their confidence scores. Then, our RTP generates the proposals frame by frame. At time $t+1$, we predict the movements of N_t proposals of frame I_t, and also obtain N_t regions on frame I_{t+1}. Similar in spirit, we only keep N_{t+1} positive ones (actionness score > 0.7), which are further exploited in the next iteration. The process is repeated until the proposal number $N_{t'}$ at time t' is smaller than N_{min}, which indicates that there are not enough action regions to track. In this case, we utilize RPN to re-initialize anchor proposals for t'-th frame and restart RTP on the next frames till the end, making RTP robust to bad initialization. Finally, a set of proposals can be obtained for each frame, which will be utilized in the tubelet generation.

3.2 Action Tubelet Generation

Given frame-level proposals with associated actionness scores, linking them in spatial and temporal dimensions is essential for generating action tubelets. It has two steps: i) linking action proposals based on their actionness scores and spatial overlaps in between to form tubelet proposals, which span the entire video duration, ii) temporally trimming tubelets to identify their temporal boundaries.

Tubelet linking. We formulate the linking problem as a path finding problem, which is to produce K connected paths across the whole video and K is the minimum number of action proposals in one frame. The linking score between two temporally consecutive proposals b_t^i and b_{t+1}^j is given by

$$S(b_t^i, b_{t+1}^j) = \{a_t^i + a_{t+1}^j + \gamma \ iou(b_t^i, b_{t+1}^j)\} \cdot \psi(iou), \tag{4}$$

where a_t^i and a_{t+1}^j are the actionness scores of b_t^i and b_{t+1}^j, $iou(\cdot)$ is the IoU overlap ratio of two proposals, and γ is a scalar parameter for balancing the actionness scores and overlaps. We define $\psi(iou)$ as the following threshold function:

$$\psi(iou) = \begin{cases} 1, & \text{if} \ \ iou(b_t^i, b_{t+1}^j) > \tau, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

According to Equation (4), two proposals b_t^i and b_{t+1}^j will be linked if their spatial regions significantly overlap and their actionness scores are both high.

To find the optimal path across the video, we maximize the linking scores over the duration T of the video, which is calculated by

$$P^* = \arg\max_{P} \frac{1}{T-1} \sum_{t=1}^{T-1} S(b_t, b_{t+1}). \tag{6}$$

We solve it with Viterbi algorithm, whose complexity is $\mathrm{O}(T \times N^2)$, where N is the average number of proposals per frame. Once an optimal path is found, we remove all the proposals in it and seek the next one from the remaining proposals.

Temporal trimming. The above tubelet linking produces tubelets spanning the whole video. In realistic videos, human actions typically occupy only a fraction. In order to determine the temporal extent of an action instance within the tubelet, we employ a similar temporal trimming approach as in [28], which solves an energy maximization problem via dynamic programming. The idea behind is to restrict consecutive proposals to have smooth actionness scores. Note that temporal trimming is only performed on untrimmed datasets in this work.

3.3 Recurrent Tubelet Recognition Networks

Recent detection works always exploit RoI pooling features for action classification directly, which have two main drawbacks. First, the long-term temporal information is not incorporated for action recognition. Second, some action-related semantic clues, such as scene context, are neglected by only utilizing RoI features. To address the issues, we propose Recurrent Tubelet Recognition (RTR) networks with multiple semantic channels to recognize the generated tubelets.

RTR capitalizes on a three-channel architecture for recognition, as illustrated in Figure 3(b). It leverages three different semantic-level information, i.e., *human only*, *human-object interaction*, and *scene*. ***Human only*** (H) channel takes proposal-cropped images as input. It focuses on human region and is expected to be capable of classifying "body motion only" actions, such as "walking" and "jumping." ***Human-object interaction*** (I) channel exploits the RoI pooling layer to extract a fixed-length feature vector for each proposal. Since the receptive fields of deep convolutional layers are generally large, this channel is able to incorporate surrounding contexts. Such information can be utilized to model the relationships between human and nearby objects, which potentially improves the performance for actions involving objects such as "shooting gun." ***Scene*** (S) channel handles the whole frame and is devised to capture the global context. It provides additional scene information and facilitates the recognition for particular action categories. For example, if a pool is observed, the probability that the action "diving" exists is high. For each channel, an individual LSTM is used to model the temporal dynamics and produces a score vector. We apply late fusion on the scores from different channels to obtain the final action score of a tubelet.

4 Implementations

Details of RTPR networks. For RTP networks, we adopt the pre-trained VGG-16 [30] as our basic CNN model, and build RTP at the top of its last convolutional layer. Following [22], we exploit a two-stream pipeline for utilizing multiple modalities, where the RGB frame and the stacked optical flow "image" are considered. To fuse their proposal results, we simply merge the proposal bounding boxes with non-maximum suppression (NMS). RPN [26] is exploited for proposal initialization, which is fine-tuned on the target datasets. During training, we randomly sample a set of two consecutive frames. For each pair, we use RPN to extract region proposals of the previous frame, and keep the top $N = 300$ proposals after NMS operation. These proposals plus the pair of frames are exploited for training. The output dimension d_c of CBP layer is $4,096$. λ in Equation (3) is set to 1. In testing, we initialize the proposals of the first frame with top $N_1 = 300$ ones generated from RPN. N_{min} is set to 150. When linking the frame-level proposals to form tubelets, γ and τ are set to 1 and 0.2, respectively. All the above parameters are determined via cross validation.

For RTR networks, we also exploit the pre-trained VGG-16 [30] for feature extraction. Similar to RTP, the two-stream pipeline is employed and late fusion strategy is utilized to fuse the two streams. For all channels, we employ the output of $fc6$ layer as the input of LSTM. Specifically, for channel I, we apply the RoI pooling on the feature maps of the last convolutional layer. The grid of RoI pooling layer is fixed to 7×7. The number of hidden states in LSTM is fixed to $1,024$. Both RTP and RTR networks are trained in an end-to-end manner.

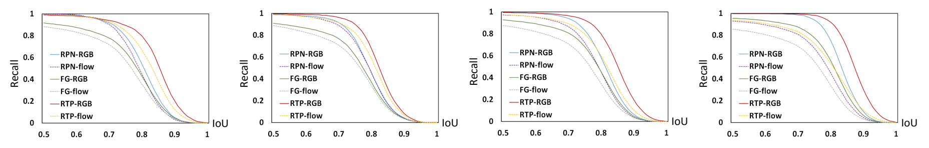
Training Strategy. Our method is implemented on Caffe [13]. We utilize mini-batch SGD algorithm to train the model. Following [4], we re-scale the frames, making their shorter sides 600 pixels. The batch size is 256 and 128 for RTP and RTR, respectively. The momentum and weight decay are set to 0.9 and 0.0005 for both networks. The initial learning rate is 0.001 and we decrease to its 10% after 8 epoches. The whole training procedure stops at 12 epoches.

5 Experiments

5.1 Datasets and Evaluation Metrics

We empirically evaluate our proposed framework on four datasets: UCF-Sports, J-HMDB, UCF-101 and AVA. **UCF-Sports** [27] consists of 150 short videos from 10 sports categories. Videos are truncated to actions and bounding box annotations are available for all frames. Following [17], 103 and 47 videos are used for training and testing, respectively. **J-HMDB** [12] contains 928 well trimmed video clips of 21 actions. The bounding box annotations are inferred from human silhouettes. It provides three training/test splits for evaluation. Following [29], we conduct analysis of different components on the first split, and report the average results over three splits when comparing to the state-of-the-arts.

UCF-101 [32] is an action recognition dataset of realistic videos. For detection, a subset of 24 classes with $3,207$ videos are provided with spatio-temporal ground truths. Unlike UCF-Sports and J-HMDB in which the whole videos are truncated to actions, videos in UCF-101 are untrimmed, and additional annotations of action temporal range are available. Following [22,28,39], all experiments



(a) UCF-Sports (b) J-HMDB (c) UCF-101 (d) AVA

Fig. 4. The frame-level Recall vs. IoU curves of different action proposal methods on UCF-Sports, J-HMDB (split 1), UCF-101, and AVA datasets.

are conducted on the first split. **AVA** [8] is a challenging dataset published very recently. It densely annotates 80 atomic visual actions in $57.6K$ video segments collected from 192 movies. The duration of each segment is 3 seconds. Different from the above datasets where annotations are provided for all frames, only the middle frame of each 3-second-long video segment is annotated in AVA. To take full advantage of the temporal information around the annotated key-frame, 15 consecutive frames (i.e., the key-frame, 7 frames before it, and 7 frames after it) are treated as a video clip to be processed in our framework. Note that each bounding box may be associated with multiple action categories, making the dataset more challenging. Experiments are performed on the standard splits [8].

Evaluation metrics. We adopt both *frame-mAP* and *video-mAP* as our evaluation metrics [7,22]. A frame or tubelet detection is treated as positive if its IoU with the ground-truth is greater than a threshold δ and the predicted label is correct. Specifically, for UCF-Sports, J-HMDB, and UCF-101, we follow [9,11,22] to exploit video-mAP as evaluation metric. And for AVA, we follow the standard evaluation scheme in [8] to measure frame-mAP. The reported metric is the mAP at IoU threshold $\delta = 0.5$ for spatial localization (UCF-Sports, J-HMDB and AVA) and $\delta = 0.2$ for spatio-temporal localization (UCF-101) by default.

5.2 Performance Evaluations and Experimental Analysis

Evaluation on Recurrent Tubelet Proposal. We first validate the RTP networks and compare with two other proposal generation methods: RPN [26] and Flow-Guided (FG) region tracking. RPN extracts region proposals of each frame or stacked optical flow "image" separately and then links the proposals in each frame or flow "image" to form tubelet. FG is initialized with proposals of the start frame or stacked optical flow "image" extracted from RPN. Then FG simply tracks each proposal according to the average optical flow over pixels within the proposal region. For fair comparison, we also utilize VGG-16 network for RPN and FG. RPN is fine-tuned on target datasets for both methods.

Figure 4 shows the frame-level Recall vs. IoU curve comparisons on the four datasets. RTP consistently outperforms the others significantly on both RGB and Flow streams, especially at high IoU area. The results demonstrate the effectiveness of incorporating long-term temporal coherence in proposal generation. It is not surprising that FG performs the worst among all. This somewhat reveals the weakness of the straightforward tracking scheme that only exploits

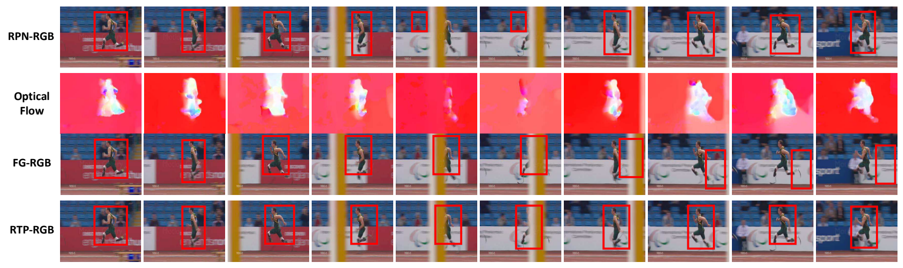


Fig. 5. An example of proposal generation results with three methods on action "Run."

Table 1. Evaluation on RTR networks with different channels on the four datasets.

Model	H	I	S	HI	IS	HS	HIS	H	I	S	HI	IS	HS	HIS
	UCF-Sports							**J-HMDB (split 1)**						
RGB	81.5	85.7	82.5	86.4	87.2	83.7	**87.6**	45.2	59.1	46.5	59.9	60.7	53.2	**61.7**
Flow	90.8	95.4	91.2	95.8	96.1	93.4	**96.5**	61.6	76.4	63.6	77.4	77.3	68.5	**78.1**
Fusion	92.3	96.8	93.4	97.4	97.1	95.1	**97.8**	66.3	78.5	70.8	79.8	80.0	74.3	**80.7**
	UCF-101							**AVA**						
RGB	54.8	59.2	56.3	59.4	59.8	58.0	**60.4**	14.4	18.2	13.7	18.9	18.7	17.2	**19.4**
Flow	65.6	71.5	68.1	72.0	72.7	70.5	**73.4**	11.2	13.7	9.4	15.2	14.9	12.5	**15.6**
Fusion	69.3	74.9	71.2	75.2	75.8	72.1	**76.3**	16.3	18.9	15.1	19.7	19.5	18.1	**20.1**

very few temporal clues, making FG sensitive to noise. Another observation is that RGB stream outperforms Flow stream. The reason is that Flow stream only focuses on salient motion regions (e.g., waving arm of a person), while the target proposals are human-centered bounding boxes which cover the entire person.

Figure 5 shows the comparison of three methods with RGB stream on an exemplar action "Run" in UCF-Sports. RPN generally works well but fails when the person is blocked by the obstacle. FG generates accurate proposals for former frames. Once the noise (occlusion) occurs, the error is aggregated and the model is then unable to localize the proposals well. RTP, in comparison, benefited from the temporal context and could predict precise proposals in current frame by leveraging those in previous frames.

Evaluation on Recurrent Tubelet Recognition. Next, we turn to measure the performance of the RTR networks on each of designed channels, i.e., *human only* (H) channel, *human-object interaction* (I) channel, and *scene* (S) channel, based on the tubelets generated by RTP. All possible combination schemes between three channels are considered. Table 1 summarizes the comparisons across different modalities. As indicated by our results, the channel I achieves the highest performance among the three channels, as the majority of action classes are related to objects (8 in 10 for UCF-Sports, 14 in 21 for J-HMDB, 20 in 24 for UCF-101, 49 in 80 for AVA). Moreover, the fusion on any two or all three channels could further improve the results, indicating that the three channels are complementary. In addition, combining RGB and Flow streams

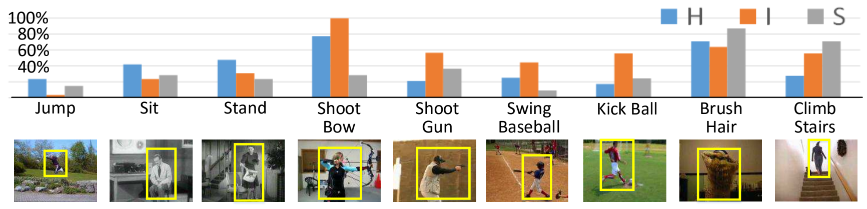


Fig. 6. Performance comparison of *human only* (H) channel, *human-object interaction* (I) channel, and *scene* (S) channel on nine action classes in J-HMDB (split 1).

also leads to considerable improvements. It is worth noting that the channel S performs better than the channel H on UCF-Sports, J-HMDB (split 1) and UCF-101, but worse on AVA. This observation is not surprising because most actions in AVA are collected from similar movie scenes and it is difficult to distinguish them with scene information. Similar in spirit, the performances of Flow stream are superior over RGB stream on UCF-Sports, J-HMDB (split 1) and UCF-101, but inferior on AVA, due to the fact that the actions in the first three datasets have more obvious movements and thus produce salient optical flows.

As described in section 3.3, our multi-channel RTR is devised to capture different semantic-level clues. To verify this, we examine the performance of these channels on several categories from J-HMDB and report the performances on RGB stream in Figure 6. We observe that the three channels indeed capture different semantic information. Specifically, for "body motion only" actions (e.g., "Jump," "Sit," and "Stand"), channel H performs the best. Similarly, on the object-related categories (e.g., "Shoot Bow," "Shoot Gun," "Swing Baseball," and "Kick Ball"), channel I achieves the best results and channel S outperforms other two on scene-related categories, e.g., "Brush Hair" and "Climb Stairs."

An Ablation Study. Here we study how each design in RTPR influences the overall performance. The basic way is to directly utilize Faster R-CNN on each RGB frame for both proposal generation and recognition, plus the tubelet generation between them. Average score over all proposals in the tubelet is exploited. **RTP** replaces the independent proposal generation with our recurrent scheme. **LSTM** leverages the temporal dynamics for recognition. **HIS** further employs the multi-channel architecture. **Flow** exploits optical flow stream additionally.

Table 2 details the improvement by considering one more factor at each stage. RTP is an effective method for action tubelet generation. It successfully boosts up the performance with 1.4%, 1.7%, 1.9%, and 1.2% on UCF-Sports, J-HMDB (split 1), UCF-101, and AVA, respectively. LSTM and HIS are two components of our RTR, which also lead to considerable improvement. More specifically, the performance gains range from 0.5% to 1.4% of LSTM, and 1.2% to 2.6% of HIS across the four datasets. In addition, we observe that the Flow stream exhibits significant improvements on UCF-Sports (10.2%), J-HMDB (19.0%) and UCF-101 (15.9%), but marginal gain on AVA (0.7%). As mentioned, this is because the actions in the first three datasets have more intensive motions.

Table 2. Performance contribution of each component in the proposed RTPR. U-S, J-H, and U-1 represent UCF-Sports, J-HMDB (split 1), and UCF-101 respectively.

Method	**RTP**	**LSTM**	**HIS**	**Flow**	**U-S**	**J-H**	**U-1**	**AVA**
Faster R-CNN					83.8	56.5	56.0	15.6
+RTP	√				85.2	58.2	57.9	16.8
+LSTM	√	√			85.7	59.1	59.2	18.2
+HIS	√	√	√		87.6	61.7	60.4	19.4
RTPR	√	√	√	√	**97.8**	**80.7**	**76.3**	**20.1**

Table 3. Video-mAP comparisons on UCF-Sports, J-HMDB, and UCF-101.

Method	**UCF-Sports**		**J-HMDB**					**UCF-101**			
	0.2	0.5	0.1	0.2	0.3	0.4	0.5	0.05	0.1	0.2	0.3
Gkioxari *et al.* [7]	-	75.8	-	-	-	-	53.3	-	-	-	-
Weinzaepfe *et al.* [38]	-	90.5	-	63.1	63.5	62.2	60.7	54.3	51.7	46.8	37.8
Saha *et al.* [28]	-	-	72.7	72.6	72.6	72.2	71.5	79.1	76.6	66.8	55.5
Peng *et al.* [22][1]	94.8	94.7	-	74.3	-	-	73.1	78.8	77.3	72.9	65.7
Singh *et al.* [31]	-	-	-	73.8	-	-	72.0	-	-	73.5	-
Kalogeiton *et al.* [14]	92.7	92.7	-	74.2	-	-	73.7	-	-	77.2	-
Hou *et al.* [11][2]	95.2	95.2	-	78.4	-	-	76.9	78.2	77.9	73.1	69.4
Yang *et al.* [39]	-	-	-	-	-	-	-	79.0	77.3	73.5	60.8
He *et al.* [9]	96.0	95.7	79.8	79.7	79.3	78.5	77.0	-	-	71.7	-
RTPR											
-w/ VGG-16	97.8	97.8	**83.0**	82.3	82.0	81.2	80.5	81.5	80.7	76.3	70.9
-w/ ResNet-101	**98.6**	**98.6**	**83.0**	**82.7**	**82.3**	**82.3**	**81.3**	**82.1**	**81.3**	**77.9**	**71.4**

Comparison with State-of-the-art. In addition to VGG-16, we also utilize ResNet-101 [10] as our backbone. In Table 3, we summarize the performance comparisons on UCF-Sports, J-HMDB (3 splits) and UCF-101 with different IoU thresholds δ. Early works mainly exploit R-CNN or Faster R-CNN to perform per-frame action detection [7,22,28,38]. T-CNN [11] improves them by utilizing 3D CNN to model short-term temporal information. Our RTPR achieves the best performance in most cases. Specifically, at the standard δ value (0.5 for UCF-Sports/J-HMDB, and 0.2 for UCF-101), our VGG-16 based model makes the improvements over VGG-16 based two-stream networks [9, 22] by 2.1%$\sim$7.4%. Ours also outperforms T-CNN [11] by 2.6%, 3.6%, and 3.2% on the three datasets respectively. This somewhat reveals the weakness of [11] when the detection data is insufficient to support 3D ConvNets fine-tuning or training from scratch. Compared to the competitor CPLA [39], our approach boosts up the performance from 73.5% to 76.3% on UCF-101 when $\delta = 0.2$. More importantly, from Table 1 we can see that only utilizing the single channel I in RTPR already exceeds most state-of-the-art approaches including CPLA. The results indicate the advantages of modeling the temporal correlations of proposals between consecutive frames

[1] Updated result from https://hal.inria.fr/hal-01349107/file/eccv16-pxj-v3.pdf
[2] Updated result from https://arxiv.org/pdf/1712.01111.pdf

Table 4. Frame-mAP comparisons on AVA.

Method	**RGB**	**Flow**	**Fusion**
[8] w/ ResNet-101	17.1	9.3	18.4
RTPR w/ VGG-16	19.4	15.6	20.1
RTPR w/ ResNet-101	**20.5**	**16.2**	**22.3**

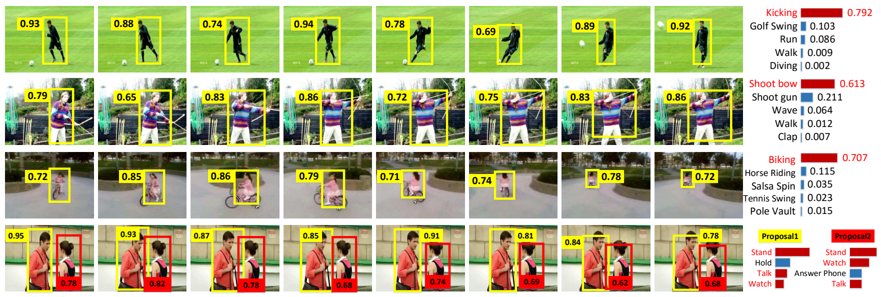


Fig. 7. Four detection examples of our method from UCF-Sports, J-HMDB, UCF-101, and AVA. The proposal score is given for each bounding box. Top predicted action classes for each tubelet are on the right. Red labels indicate ground-truth.

to predict movements in RTP rather than relying on RPN for each individual frame in CPLA. In addition, our ResNet-101 based model outperforms the best competitors [9, 14] by 2.9%, 4.3% and 0.7% on the three datasets respectively.

Table 4 shows the comparisons on AVA. Since AVA is a very recent dataset, there are very few studies on it and we only compare with [8], which implements Multi-Region Two-Stream CNN [22] with ResNet-101. We can observe that ours with both VGG-16 and ResNet-101 basic models outperform the baseline. Figure 7 showcases four detection examples from UCF-Sports, J-HMDB, UCF-101, and AVA. Even in complex cases, e.g., varying scales (third row) and multi-person plus multi-label (last row), our approach can still work very well.

6 Conclusion

We have presented Recurrent Tubelet Proposal and Recognition (RTPR) networks for video action detection, which is able to incorporate temporal contextual information. Particularly, we study the problem of utilizing the proposals in previous frames to facilitate the detection in current frame through building a recurrent neural network. To verify our claim, we have devised Recurrent Tubelet Proposal networks, which is to estimate the movements of proposals in the next frame in a recurrent manner. Furthermore, a multi-channel architecture is designed for proposal recognition, which leverages different semantic context to enhance recognition. Experiments conducted on four public datasets validate our model and analysis. More remarkably, we achieve the new state-of-the-art performances on all the four datasets.

References

1. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: CVPR (2015)
2. Feichtenhofer, C., Pinz, A., Zisserman, A.: Detect to track and track to detect. In: CVPR (2017)
3. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: CVPR (2016)
4. Girshick, R.: Fast r-cnn. In: ICCV (2015)
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
6. Gkioxari, G., Girshick, R., Malik, J.: Contextual action recognition with r* cnn. In: ICCV (2015)
7. Gkioxari, G., Malik, J.: Finding action tubes. In: CVPR (2015)
8. Gu, C., Sun, C., Vijayanarasimhan, S., Pantofaru, C., Ross, D.A., Toderici, G., Li, Y., Ricco, S., Sukthankar, R., Schmid, C., et al.: Ava: A video dataset of spatio-temporally localized atomic visual actions. arXiv preprint arXiv:1705.08421 (2017)
9. He, J., Ibrahim, M.S., Deng, Z., Mori, G.: Generic tubelet proposals for action localization. In: WACV (2018)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
11. Hou, R., Chen, C., Shah, M.: Tube convolutional neural network (t-cnn) for action detection in videos. In: ICCV (2017)
12. Jhuang, H., Gall, J., Zuffi, S., Schmid, C., Black, M.J.: Towards understanding action recognition. In: ICCV (2013)
13. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM MM (2014)
14. Kalogeiton, V., Weinzaepfel, P., Ferrari, V., Schmid, C.: Action tubelet detector for spatio-temporal action localization. In: ICCV (2017)
15. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR (2014)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
17. Lan, T., Wang, Y., Mori, G.: Discriminative figure-centric models for joint action localization and recognition. In: ICCV (2011)
18. Laptev, I., Pérez, P.: Retrieving actions in movies. In: ICCV (2007)
19. Li, Q., Qiu, Z., Yao, T., Mei, T., Rui, Y., Luo, J.: Action recognition by learning deep multi-granular spatio-temporal video representation. In: ICMR (2016)
20. Li, Q., Qiu, Z., Yao, T., Mei, T., Rui, Y., Luo, J.: Learning hierarchical video representation for action recognition. IJMIR **6**(1), 85–98 (2017)
21. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: ECCV (2016)
22. Peng, X., Schmid, C.: Multi-region two-stream r-cnn for action detection. In: ECCV (2016)
23. Qiu, Z., Yao, T., Mei, T.: Deep quantization: Encoding convolutional activations with deep generative model. In: CVPR (2017)
24. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: ICCV (2017)

25. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR (2016)
26. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NIPS (2015)
27. Rodriguez, M.D., Ahmed, J., Shah, M.: Action mach a spatio-temporal maximum average correlation height filter for action recognition. In: CVPR (2008)
28. Saha, S., Singh, G., Sapienza, M., Torr, P.H., Cuzzolin, F.: Deep learning for detecting multiple space-time action tubes in videos. In: BMVC (2016)
29. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: NIPS (2014)
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
31. Singh, G., Saha, S., Cuzzolin, F.: Online real time multiple spatiotemporal action localisation and prediction. In: ICCV (2017)
32. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
33. Srivastava, N., Mansimov, E., Salakhudinov, R.: Unsupervised learning of video representations using lstms. In: ICML (2015)
34. Tian, Y., Sukthankar, R., Shah, M.: Spatiotemporal deformable part models for action detection. In: CVPR (2013)
35. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV (2015)
36. Wang, L., Qiao, Y., Tang, X., Van Gool, L.: Actionness estimation using hybrid fully convolutional networks. In: CVPR (2016)
37. Wang, Y., Song, J., Wang, L., Van Gool, L., Hilliges, O.: Two-stream sr-cnns for action recognition in videos. In: BMVC (2016)
38. Weinzaepfel, P., Harchaoui, Z., Schmid, C.: Learning to track for spatio-temporal action localization. In: ICCV (2015)
39. Yang, Z., Gao, J., Nevatia, R.: Spatio-temporal action detection with cascade proposal and location anticipation. In: BMVC (2017)
40. Yuan, J., Liu, Z., Wu, Y.: Discriminative subvolume search for efficient action detection. In: CVPR (2009)
41. Yue-Hei Ng, J., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: Deep networks for video classification. In: CVPR (2015)