SEARCH VIDEO ACTION PROPOSAL WITH RECURRENT AND STATIC YOLO

Romain Vial[*], *Hongyuan Zhu*⋆, *Yonghong Tian*[†]∘, *Shijian Lu*⋆

∗ MINES ParisTech, PSL Research University, France
⋆ Institute for Infocomm Research, A*STAR, Singapore
∘ Peking University, China

ABSTRACT

In this paper, we propose a new approach for searching action proposals in unconstrained videos. Our method first produces snippet action proposals by combining state-of-the-art YOLO detector (Static YOLO) and our regression based RNN detector (Recurrent YOLO). Then, these short action proposals are integrated to form final action proposals by solving two-pass dynamic programming which maximizes actioness score and temporal smoothness concurrently. Our experimental comparison with other state-of-the-arts on challenging UCF101 dataset shows that our method advances state-of-the-art proposal generation performance while maintaining low computational cost.

Index Terms— action detection, action localization, action proposal, video object proposal, video object detection

1. INTRODUCTION

The explosion of video-streams has greatly boosted the development of related video analysis technologies. However, directly processing raw-video streams is computational expensive and may also hurt learning accuracy due to a large number of irrelevant background and motion changes, while meaningful human actions only take up a small portion of the videos. Hence, action proposal which produces a small number of sequences of bounding box has been attracting increasingly attentions in recent years.

This paper presents a new approach for video action proposal. Existing works have tackled the problem from different perspectives, e.g. voxel segmentation- and merging [1], dense motion feature tracking [2], human-centric models and object proposal [3, 4]. Though much progress has been achieved, video action proposal is still a challenging problem. Different from static proposal which relies on appearance cue only, action proposals need to consider not only spatial context in the current frame but also temporal context among adjacent frames. Due to the diversity and variations of human actions, background clutters and other dynamic motions, existing successful static proposal does not perform well for action recognition tasks.

To handle these challenges, we propose a novel framework for action proposal based on the latest development of deep learning, which combines the recent popular YOLO detector [5] with Recurrent Neural Network (RNN). Our framework complements existing frame level action proposal inference with temporal dynamic modeling using RNN. Hence the frame-level proposal can be generated by using strong CNN, meanwhile the objectness information from other frames can be propagated by RNN which provides rich temporal context information. These information are integrated together to form snippet-level action proposals and then we solve two dynamic programming problems to form final action proposal tubes. To evaluate the performance of action proposals, we test our method on challenging UCF-101 dataset. We notice that a small number of action proposals, e.g. 30 proposal tubes can already provide promising recall rate at a speed of 5FPS. Different from existing action proposal approaches, our action proposals do not rely on video segmentation hence are computationally efficient.

To summarize our achievements:

1. We propose the first recurrent video action proposal method by exploring the regression capability of RNN to exploit the spatial and temporal information simultaneously, to the best of our knowledge.

2. We formulate the action proposal generation as energy maximization problem which considers both actionness measure and temporal overlap.

3. Experiment on UCF-101 dataset proves that our method achieves the state-of-the-art performance.

2. METHODOLOGY

Our framework firstly regresses the snippet-level action proposals with Recurrent YOLO and static YOLO, as shown in Fig. 1. The snipped proposals are seamed into final action proposal by solving two-stage dynamic programming with trimming and some final proposals are shown in Fig. 3.

[*]The first author performed the work while he was an intern-student at I2R, Singapore

[†]This work is partially supported by grants from the National Natural Science Foundation of China under contract No. U1611461.

2.1. Recurrent YOLO and Static YOLO

Most proposal methods use hand-crafted features (e.g. EdgeBox [6] and SelectiveSearch [7]) or train ConvNets to perform detection on static images (e.g YOLO [5], FasterRCNN [8] and Actioness Estimation [9]), which ignores the temporal context. Hence, we explore to integrate spatial-temporal context among adjacent frames by using recent popular RNN. RNN has been applied to perform video classification [10] and fuse feature maps for static detection [11] and semantic segmentation [12, 13], however the application of RNN for object detections are less discussed. In this work, we explored the regression capability of RNN to directly regress the coordinates and confidence of the bounding boxes of potential action regions, which is inspired by the recent popular YOLO detector [5].

Static YOLO divides the image into $K \times K$ grid ($K = 7$ in our work). Each grid cell will predicts $B = 2$ bounding boxes with the confidence score that the box contains an object of interest and it will also predict an actioness score (s_{ac}, s_{bg}). Hence the output prediction is a $K \times K \times (B \times 5 + 2)$ tensor, where each bounding box is parametrized by (x, y, w, h, C) where (x, y) represent the center of the box relative to the bounds of the grid cell. The box's width and height are normalized with respect to the whole image's width and height. The confidence C predicts the objectness score of a bounding box. The parameters s_{ac} and s_{bg} predict the a box's likelihood belongs to 'action' or 'background'. The loss function is similarly defined as in [5]:

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

where 1_i^{obj} denotes if object appears in cell i and 1_{ij}^{obj} denotes that the jth bounding box predictor in cell i is "responsible" for the prediction. The loss function penalizes classification error if object appears in the grid cell. It also penalizes bounding box coordinate error if that predictor has the highest overlap with the ground-truth in that grid cell [5]

Recurrent YOLO uses the same loss function as Static YOLO, but replaces the last fully connected layer of Static version by a Long-Short Term Memory (LSTM), a recent popular variant of RNN, whose hidden cell has a dimension of 588. Hence, we use the powerful base ConvNet of Static YOLO as feature extractor, and train an LSTM on top of the CNN to directly regress the coordinates of the bounding box that contains actions. Hence, we can consider spatial and and temporal context with a single framework. The output activation function is the linear activation. We apply dropout with 0.5 to avoid over-fitting.

We use Adam [14] optimizer during training with default parameters. While training the Static YOLO, we use a batch size of 32 frames from different videos during 100 epochs with an initial learning rate of 10^{-3} decaying and 10^{-4} after the 20^{th} epoch. When training the Recurrent YOLO, we then use a batch size of 10 sequences of 10 frames from different videos during 50 epochs. The same learning rate planning is used.

After training the recurrent YOLO and static YOLO, we combine these two methods to produce high quality sequences of proposals.

2.2. Path Linking and Trimming

The output of Sec. 2.1 is a set of bounding boxes for each frame of the video $\mathbf{B} = \{\{b_i^{(j)}, j \in [1 \dots N_{b_i}]\}, i \in [1 \dots T]\}$ where T is the length of the video and N_{b_i} is the number of predicted boxes in frame i. For each box $b_i^{(j)}$ we have its confidence score C, action score s_{ac} and background score s_{bg}.

The next objective is to create a set of proposal paths $\mathbf{P} = \{p_i = \{b_{s_i}, b_{s_i+1} \dots b_{e_i}\}, i \in [1 \dots |\mathbf{P}|]\}$ where s_i and e_i are, respectively, the starting frame and ending frame of path p_i. The action proposal can be generated by solving two-stage energy maximization problems.

2.2.1. Action Path Linking

The first energy function scores a path p as follows:

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

The unary term helps to choose the best scored boxes, which are supposed to be the most well located, and the pairwise term increases coherence of the path by avoiding jumping from one position in the frame to another. λ_0 is a trade-off factor.

Given the set of boxes $\mathbf{B}$, we can solve the maximization problem and find the best path $\hat{p}$ using the Viterbi algorithm. After the optimal path is found, the boxes in $\hat{p}$ are removed and the algorithm is solved again. This is repeated until one of the frames contains no predicted boxes. We can also stop the algorithm earlier by setting a maximum number of paths.

2.2.2. Temporal Action Proposal Trimming

The action proposal generated at Sec. 2.2.1 spans the entire video duration, while human actions typically only occupy a

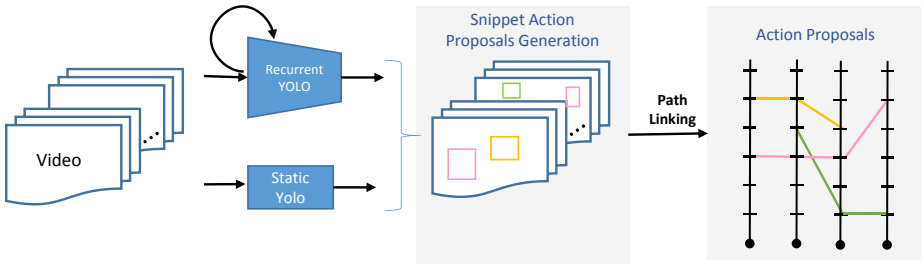

Fig. 1. Conceptual illustration of our method: we explore the regression capability of RNN and CNN to directly regress snippet-level action proposals. Then these proposals are seamed into longer action proposals with dynamic programming(to be described in Sec. 2.2).

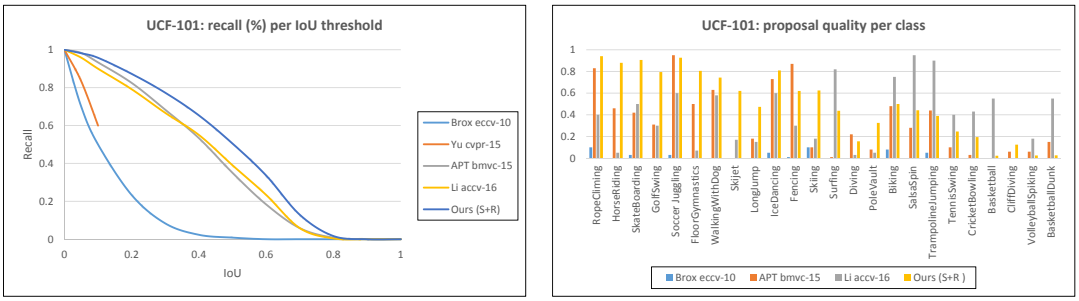

Fig. 2. . Recall vs IoU and Recall-per-Class on UCF-101 datasets.

fraction of it, hence it is necessary to perform temporal trimming. The first pass of dynamic programming aims at extracting connected paths by penalizing regions which do not overlap in time. As a result, however, not all detection boxes within a path exhibit strong actionness scores.

To perform temporal trimming, each box of a path p will be assigned a binary label $l_i \in \{bg, ac\}$, subject to the condition that the path's labeling is consistent with 1) the unary scores and 2) is smooth (no sudden jumps). Hence, we assign the following score to the path's labeling $L_p = [l_1 \dots l_T]$, which is adapted from [15]:

$$S(L_p) = \underbrace{\sum_{i=1}^{T} s_{l_i}(b_i)}_{unary} - \underbrace{\lambda_1 \times \sum_{i=2}^{T} \phi(l_i, l_{i-1})}_{pairwise}$$

where λ_1 is a trade-off factor and

$$\phi(l_i, l_{i-1}) = \begin{cases} 0 & \text{if } l_i = l_{i-1} \\ \alpha_{l_i} & \text{otherwise} \end{cases}$$

where α_{ac} and α_{bg} are parameters set by cross-validation.

The unary term of Eq. 2.2.2 tries to maximize the global score of the labeling by assigning the frame-label with the highest score whereas the pairwise term tend to avoid not smooth labeling. It can also be solved by the Viterbi algorithm as in Sec. 2.2.1. All consecutive frames assigned to the action label are extracted and constitute the final action paths.

3. EXPERIMENT

UCF-101 is a popular challenging dataset to date, and contains untrimmed sequences with large variation in camera motion, appearance, human pose, background clutter, scale and illumination. It contains 3207 video clips with 24 categories of actions with bounding box annotations. Although each video contains a single action category, it may contain multiple instances of the same action class. We evaluate our algorithm on split 1 as in [16] .

We first evaluate the performance of our action proposals based on the recall. We consider a hit of ground-truth **G** if the spatial-temporal intersection-over-union (IOU) $\phi(\mathbf{p}, \mathbf{G}) \geq \theta$ as in [4], and **p** is the predicted proposal, **G** is the ground-truth tube, and θ is the overlap threshold. Fig. 2 shows the recall of our action proposal algorithm for various IoU threshold and the recall for each class. One can observe that our method achieves the state-of-the-arts in proposal generation in a wide range of IoU threshold. Moreover, our method also perform well in a wider range of action classes, which means our method can generalize across classes. Fig. 4 shows the ablation study for each components. Recurrent YOLO performs slightly better than static YOLO. It is because the re-

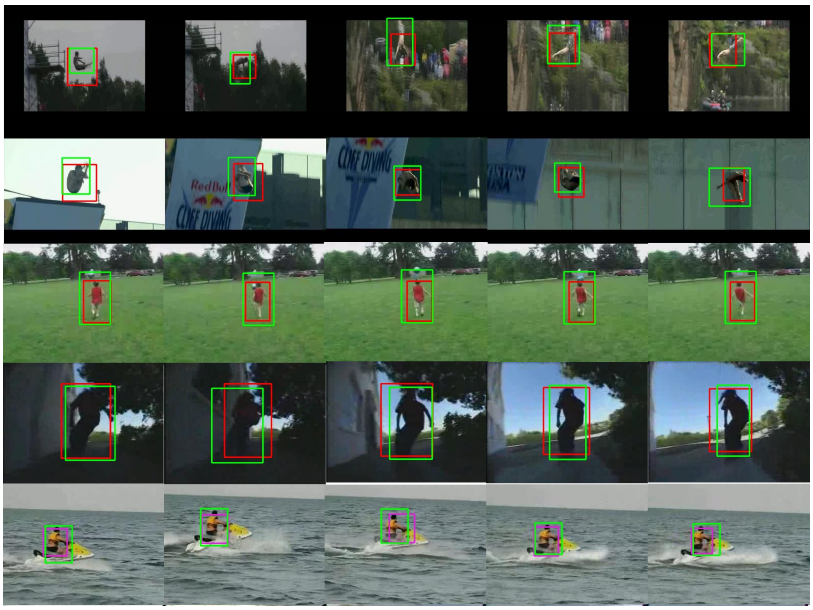

Fig. 3. . Qualitative results for randomly sampled 4 videos in UCF-101 datasets. The bounding boxes with green and red color are the ground-truth and the action proposal which has an IoU ≥ 0.5 with the ground-truth, respectively.

current YOLO captures the long-range temporal dynamics of human actions. While the ensemble of both detectors improve the recall by 1.5% and 2% over the recurrent YOLO and static YOLO respectively, which shows that static and recurrent YOLO are complementary to each other.

Finally, we report the overall performance in Table. 1 using several commonly used metrics, including ABO (Average Best Overlap), MABO (Mean ABO over all classes) and average number of proposals per video, which are also commonly used metric [4]. The result further confirms the superior performance of our method than the state-of-the-arts in terms of MABO, Recall using a small number of proposals.

In Fig. 3, a few action proposals are illustrated with red rectangles. We can see that even with serious pose and scale changes, our action proposal can correctly localize the same actors under various challenges with abrupt motion change, background clutters and illumination variation.

UCF-101	ABO	MABO	Recall	#Proposal
Brox *et al.* [1]	13.28	12.82	1.40	**3**
Yu *et al.*[16]	n.a	n.a	0.0	10,000
APT [4]	40.77	39.97	35.45	2299
Li *et al.* [3]	**63.76**	40.84	39.64	18
Ours	47.51	**47.72**	**50.46**	35

Table 1. Quantitative comparison with the state-of-the-arts with commonly used metrics.

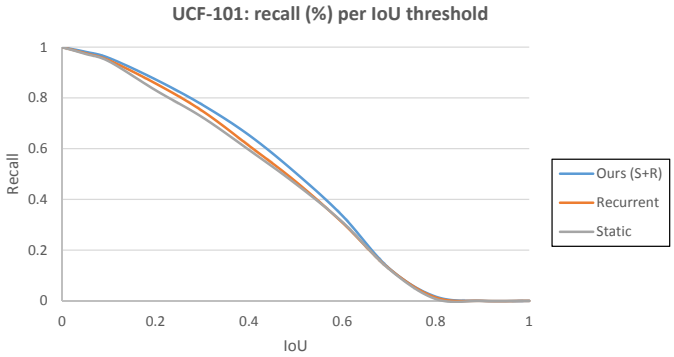

Fig. 4. . Recall vs IoU on UCF-101 for static and recurrent.

4. CONCLUSION

We propose a novel framework for searching action proposals in video clips. Given an unconstrained video as input, our method produces a relatively small number of spatially compact and temporally smooth action proposals. The proposed method combines the state-of-the-art YOLO detector with RNN to sequentially regress action proposal, hence help improve the localization accuracy and reduce the number of false positives. These snippet proposals are further integrated by maximizing two energy functions with dynamic programming. The experimental results on challenging UCF-101 dataset highlights the effectiveness of our modeling.

5. REFERENCES

[1] Thomas Brox and Jitendra Malik, "Object segmentation by long term analysis of point trajectories," in *ECCV*, 2010.

[2] Dan Oneata, Jérôme Revaud, Jakob J. Verbeek, and Cordelia Schmid, "Spatio-temporal object detection proposals," in *ECCV*, 2014.

[3] Nannan Li, Dan Xu, Zhenqiang Ying, and Ge Li Zhihao Li, "Search action proposals via spatial actionness estimation and temporal path inference and tracking," in *ACCV*, 2016.

[4] Jan C. van Gemert, Mihir Jain, Ella Gati, and Cees G. M. Snoek, "APT: action localization proposals from dense trajectories," in *BMVC*, 2015.

[5] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, "You only look once: Unified, real-time object detection," in *CVPR*, 2016.

[6] C. Lawrence Zitnick and Piotr Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014.

[7] Jasper R. R. Uijlings, Koen E. A. van de Sande, Theo Gevers, and Arnold W. M. Smeulders, "Selective search for object recognition," *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[8] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[9] Limin Wang, Yu Qiao, Xiaoou Tang, and Luc J. Van Gool, "Actionness estimation using hybrid fully convolutional networks," 2016.

[10] Bharat Singh, Tim K. Marks, Michael J. Jones, Oncel Tuzel, and Ming Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *CVPR*, 2016.

[11] Sean Bell, C. Lawrence Zitnick, Kavita Bala, and Ross B. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *CVPR*, 2016.

[12] Bing Shuai, Zhen Zuo, Bing Wang, and Gang Wang, "Dag-recurrent neural networks for scene labeling," in *CVPR*, 2016.

[13] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, and Y. Chen, "Learning contextual dependence with convolutional hierarchical recurrent neural networks," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 2983–2996, 2016.

[14] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[15] Suman Saha, Gurkirt Singh, Michael Sapienza, Philip H. S. Torr, and Fabio Cuzzolin, "Deep learning for detecting multiple space-time action tubes in videos," in *BMVC*, 2016.

[16] Gang Yu and Junsong Yuan, "Fast action proposals for human action detection and search," in *CVPR*, 2015.