Mining Spatial and Spatio-Temporal ROIs for Action Recognition

Xiaochen Lian[1] Zhuoyuan Chen[2] Yi Yang[2] Jiang Wang[2] Alan Yuille[1,3]
[1]University of California, Los Angeles [2]Baidu Research, USA [3]John Hopkins University
{lianxiaochen@,yuille@stat.}ucla.edu {chenzhuoyuan,yangyi05,wangjiang03}@baidu.com

Abstract

In this paper, we propose an approach to classify action sequences. We observe that in action sequences the critical features for discriminating between actions occur only within sub-regions of the image. Hence deep network approaches will address the entire image are at a disadvantage. This motivates our strategy which uses static and spatio-temporal visual cues to isolate static and spatio-temporal regions of interest (ROIs). We then use weakly supervised learning to train deep network classifiers using the ROIs as input. More specifically, we combine multiple instance learning (MIL) with convolutional neural networks (CNNs) to select discriminative action cues. This yields classifiers for static images, using the static ROIs, as well as classifiers for short image sequences (16 frames), using spatio-temporal ROIs. Extensive experiments performed on the UCF101 and HMDB51 benchmarks show that both these types of classifiers perform well individually and achieve state of the art performance when combined together.

1 Introduction

Recognition of human actions in realistic videos is a challenging, due to its complex content, cluttered background, and large intra-class variations. Humans appear to tackle this challenge using two abilities: (i) The ability to rapidly detect static and spatio-temporal regions of interest (ROIs), instead of processing the entire image (e.g., bottom-up attention). (ii) The ability to determine which ROIs are useful for detecting specific actions and to extract the relevant visual cues for action discrimination. These ROIs contain the key information about the action.

These considerations motivate us to propose a video action recognition method that attends to regions of the videos, instead of the entire video. The proposed method consists of two models: the Static Model and the Motion Model. Both models mine ROIs in the video to obtain discriminative action cues: The Static Model takes image frames as input and uses generic object proposal methods (*e.g.* [17]) to propose static ROIs. The Motion Model works on video clips (*i.e.* a short sequence of frames), and mines spatio-temporal ROIs, which we call *video tubes*.

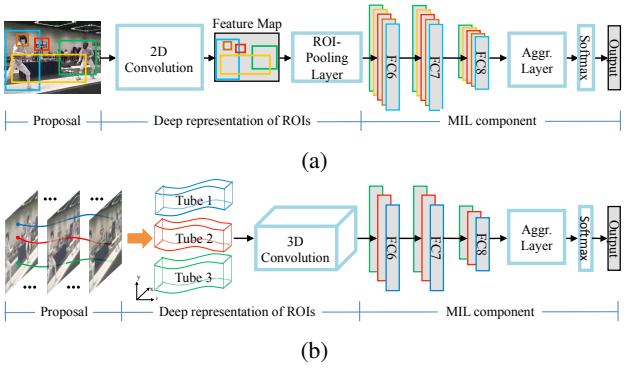

Figure 1: The network architectures of the proposed Static Model (a) and the Motion Model (b)

Mining the ROIs is challenging because we do not know which ROIs are helpful for discriminating the actions. It would be helpful if the ROIs were annotated by action class, but this has only been done for humans (*e.g.* UCF101 [8], J-HMDB [2]). This means we cannot use fully supervised methods for mining the ROIs and must instead use weak supervision. More specially, we use multiple instance learning (MIL), where a video frame or a video clip is a "bag" and the ROIs are its "instances". We combine ML with deep convolutional neural networks (CNNs) to mine deep features from the ROIs. This enables both the Static and the Motion Models to classify image frames and video clips respectively. Our final system combines these classifiers.

2 Approach

In this section, we describe in details the Static Model and the Motion Model. Both models have three components: ROI proposal generation, computation of deep features within ROIs, and training the deep network using MIL (after encoding and aggregation of the ROI deep features).

The ROI proposal algorithms are low-level and class-agnostic, since learning proposals would require annotated ROIs. We use an ROI ranking mechanism, so that our models only need to process a few, top scored, ROIs. This saves computation time and simplifies learning discriminative classifiers. Deep convolutional features are computed

Figure 2: An example of our region proposals for Static Model. The left is the original frame image, the middle is the edge map, the right shows top 10 bounding box ROIs.

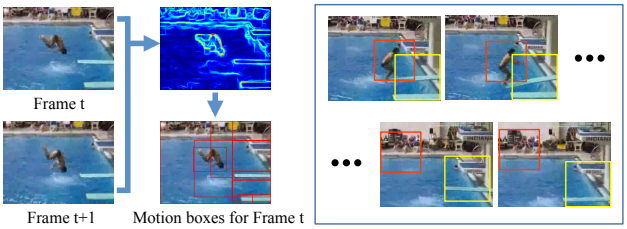

Figure 3: Left: Motion box generation on a single frame. Right: Two video tubes proposals on the first two and last two frames of a 16-frame video clips. The red tubes localizes the diver and the yellow one finds the diving board.

for ROIs, which are then encoded and aggregated using MIL.

2.1 Static Model

Fig. 1a shows the pipeline of the Static Model. Given an image frame I from a video, the first step is generating a set of candidate regions, which will be used as instances in the MIL framework. We use video labels as bag labels and the labels of instances (*i.e.* of the ROIs) are unknown and treated as latent variables. The deep convolutional features of the candidate regions are instance-level features. Next, the MIL component of the Static Model encodes the instance features, and learns the action classification model using the video class label.

Spatial ROI Proposals. To obtain a list of K regions of interest (ROIs) $R(I) = \{r_1, \ldots, r_K\}$ from frame I, we use the formulation of Edge Boxes [17], which estimates bounding boxes for objects based on the amount of contours wholly within the box, together with an "objectiveness" score. After obtaining ROIs from Edge Boxes, we remove small boxes (*i.e.* those whose shorter side are less than 50 pixels), and keep K boxes with highest "objectiveness" scores. We also include the whole frame region in case the full background context is needed. Fig. 2 illustrates the process.

Deep Instance Features. For each ROI r_k in $R(I)$, we compute the deep instance features $f(r_k, I; w_f)$ within it using CNN whose parameters are denoted by w_f. To compute the features efficiently, we perform convolutions at the frame level, and feed the convolutional feature map and $R(I)$ into the ROI Pooling layer [1]. This converts the features inside r_k into a feature map with a fixed spatial extent of H×W (*e.g.* 7×7 in our experiments).

Multiple Instance Learning. The instance features of the regions in $R(I)$ are then passed to the MIL component shown in Fig. 1a, which has three steps: First, the instance features are encoded, $s_k = e(f(r_k, I; w_f); w_e)$, where e represents the encoding which is a composition of three fully connected layers (*i.e.* $FC6$, $FC7$ and $FC8$) with parameters w_e. Second, the encoded features $\{s_k\}_{k=1}^K$ are mapped to one bag-level feature by the aggregation function $h = g(s_1, \ldots, s_k; w_g)$, where w_g is the parameters of the aggregation function $g(\cdot)$. Finaly, the bag feature h is transformed into the action scores of C classes p_c through the softmax function. The loss is the cross-entropy classification loss *i.e.* $-\log(p_{\hat{c}})$ where $\hat{c}$ is the ground truth class label of I.

2.2 Motion Model.

Fig. 1b shows the pipeline of our Motion Model, which is composed of low-level local motion proposals to obtain spatio-temporal ROIs (video tubes for short) followed by the multiple instance learning of action classification. The objective of the local motion proposal step is to generate a set of spatio-temporal ROIs of videos, which may contain cues for actions. Besides human motions, we also consider the movements of objects and even some backgrounds. These types of background movement are often very useful to help identify actions (*e.g.* the motion of a road as a biker cycles down it).

Video Tubes. Given a video clip of L frames $V = (I^1, \ldots, I^L)$, the goal of this step is to propose a set of K spatio-temporal ROIs, or video tubes $T = \{t_1, \ldots, t_K\}$, where each tube $t_k = (r_k^1, \ldots, r_k^L)$ is a temporal series of 2D bounding boxes that localize motions. We call these 2D bounding boxes "motion boxes". Our algorithm build up a video tube from a single image frame, by generating motion boxes on individual image frames and then linking the boxes across frames to form video tubes.

The left part of Fig. 3 illustrates motion box generation on a single frame I. Unlike the object boxes in the Static Model, motion boxes are intended to capture moving parts in the video. We apply Edge Boxes again, but use the motion boundaries [14] detected based on two consecutive image frames as edge map. In this case, the objectiveness score estimated by Edge Boxes actually reflects the amount of motion contours within in a motion box b, which we call the "motionness" score $m(b)$.

Once we have motion boxes on individual frames, we produce a set of video tubes by linking boxes across frames. A good video tube proposal t_k should have high motionness

score, *i.e.* $m(t_k) = \sum_{l=1}^{L} m(r_k^l)$ is large, and should satisfy, along the tube, spatio-temporal smoothness constraint, *i.e.* IOU$(r_k^l, r_k^{l+1}) \geq \sigma_o$ and appearance consistency constraint, *i.e.* $\parallel A(r_k^l) - A(r_k^{l+1}) \parallel_2 \leq \sigma_a$. σ_o and σ_a are thresholds and $A(\cdot)$ compute the color histogram within a box. In this paper, we use $\sigma_o = 0.5$, $\sigma_a = 0.2$ and divide R, G and B channels into 16 bins when computing color histogram.

Now for each motion box b_i^L in the last frame I^L of V, we compute the best tube ending at b_i^L, using dynamic programming, $f(b_i^l) = \max_{b_j^{l-1} \in I^{l-1}} f(b_j^{l-1}) + m(b_i^l) + d(b_i^l, b_j^{l-1})$, where $d(b_i^l, b_j^{l-1})$ is $-\infty$ if b_i^l and b_i^{l-1} do not satisfy the constraints aforementioned, and is equal to 0 otherwise. Then we can back-trace from every $b_i^L \in M(I^L)$ to recover a video tube. This yields a large amount of tubes. Finally, we apply non-maximum suppress to prune out highly overlapping video tubes, according to their motionness scores.

For each remaining video tube, say t_k, we first crop from l-th frame a square patch p_k^l with its center at the center of r_k^l and size $a = \max\big(\text{median}(h(r_k^1), \ldots, h(r_k^l)), \text{median}(w(r_k^1), \ldots, w(r_k^l))\big)$, where $h(\cdot)$ and $w(\cdot)$ returns the height and the width of a bounding box respectively. We then update t_k by replacing r_k^l with p_k^l and obtain the final video tube t_k. The right part of Fig. 3 shows two example video tubes.

Deep Instance Features We choose the 3D convolutional network (C3D) in [9] for computing the deep features of a video clip, due to its good performance and the convenience of joint end-to-end training. In C3D, traditional 2D convolution and 2D pooling operations are replaced with the 3D version, *i.e.* with an additional temporal dimension, to prevent the temporal information from being collapsed. We use the output of the last convolution layer as the instance feature.

3 Experiments

In this section, we first introduce the details of our experimental settings. Then we provide quantitative and qualitative results.

3.1 Datasets

The evaluation is performed on UCF101 [8] and HMDB51 [3] benchmarks. UCF101 contains $13,320$ videos of 101 action classes; HMDB51 includes $6,766$ videos of 51 actions. Both datasets provide three official splits into training and test data. The performance is measure by the average classification accuracy across the splits.

For comparison with the state of the art, we follow the standard evaluation protocol on both UCF101 and HMDB51.

3.2 Diagnostic Experiments

We have conducted a series of diagnostic experiments on the aggregation functions and the number of ROI, using the first split of UCF101 dataset (UCF split1). For the Static Model, max function and 20 ROIs per frame together yields the best performance (81.0%), which we denote by S-max-ROI(20). For the Motion model, max and 10 ROIs per video clip, *i.e.* M-max-ROI(10), is the best combination (84.4%) among those we tried. We will use these configurations for the following experiments.

3.3 Comparison with The State of The Art

Table 1 shows comparison between our models and the Two-Stream models on UCF101 and HMDB51. [11] used VGG-16 network to boost the performance of the original Two-Stream model [7]. Note that [11] did not report experiments on HMDB51. We fine-tune the Two-Stream model pre-trained on UCF101, and denote this model as "Two-Stream by us". Our Static Model outperforms the spatial net (*i.e.* the network operating on individual frames) of [11] in the static stream. In the motion stream, our Motion Model performs worse than the temporal net on UCF101. We argue that the temporal net uses 5 more convolution layers than us, we expect the Motion Model to get better results when fine-tuning from a deeper CNN. While on HMDB51, our Motion Model is better than the temporal net. The reason maybe HMDB51 has less training data; By attending to ROIs, our model suffers less from over fitting problem.

Table 2 presents action recognition accuracy of our method compared with current best methods. On UCF101, our method (Static Model + Motion Model) does not perform as well as [13, 11, 15]. However, when fused with the Two-Stream model [11], our method got a 2.3% performance gain and achieve the best performance. This shows that our models and the Two-Stream model are complementary to each other. On HMDB51, our method got the state of the art result on its own, and when combined with the Two-Stream model, the accuracy further increases 2.2%.

In Fig. 4 and Fig. 5, we visualize the top two scored spatial ROIs by S-ROI(20)-max and the top-two scored spatio-temporal ROIs by M-ROI(10)-max, from which we can see that our models are able to find action-related local regions.

4 Conclusion and Future Work

In this work, we introduce a novel deep action recognition method with ROIs. By exploiting video benchmarks, we find that critical representations occur with in sub-regions of videos. Based on this observation, we extract static and

Table 1: Comparison to the Two-Stream model [11]. on UCF101 (left) and the Two-Stream model fine-tuned by us ("Two-Stream by us") on HMDB51 (right).

		Two-Stream [11]	Ours	Two-Stream by us	Ours
Static	split 1	79.8%	**81.0%**	54.3%	**57.0%**
	split 2	77.3%	**78.4%**	50.3%	**52.6%**
	split 3	77.8%	**78.8%**	50.1%	**52.6%**
	average	78.4%	**79.4%**	51.6%	**53.9%**
Motion	split 1	**85.7%**	84.4%	65.6%	**66.8%**
	split 2	**88.2%**	87.7%	62.4%	**64.3%**
	split 3	**87.4%**	86.5%	62.0%	**64.0%**
	average	**87.0%**	86.2%	63.3%	**65.0%**
Fusion	split 1	**90.9%**	89.8%	70.1%	**72.0%**
	split 2	**91.6%**	91.3%	67.2%	**68.2%**
	split 3	**91.6%**	90.3%	66.8%	**68.4%**
	average	**91.4%**	90.5%	68.0%	**69.5%**

Table 2: Comparison with the state of the art results.

HMDB51		UCF101	
IDT+FV [10]	57.2%	IDT+FV [10]	85.9%
Two-Stream [7]	59.4%	Hybrid [5]	87.9%
H-VLAD [4]	59.8%	Two-Stream [7]	88.0%
Hybrid [5]	61.1%	LSTM+Two-Stream [16]	88.6%
TDD+FV [12]	63.2%	C3D+iDT+SVM [9]	90.4%
Two Stream Siamese [13]	63.4%	Hybrid LSTM [15]	91.3%
SFV [6]	66.8%	Two Stream [11]	91.4%
Two-Stream by us	68.4%	Two-Stream Siamese [13]	92.4%
Ours	**69.5%**	Ours	90.5%
Ours+Two-Stream by us	**71.7%**	Ours+Two-Stream [11]	**92.8%**

spatio-temporal regions of interest (ROI) to enhance the performance of deep network. Features from different instances are naturally integrated into our MIL framework to adaptively select the most discriminative ROIs to enable end-to-end learning. Extensive experiments on UCF 101 and HMDB51 benchmarks demonstrate that our algorithm not only outperform existing methods quantitatively, but also capture the most relevant part qualitatively.

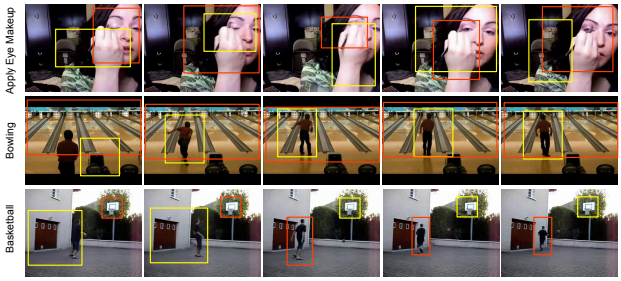

Figure 4: Visualization of the top two regions selected by S-ROI(20)-max. Each row corresponds to a video from the test partition of UCF101 split1. Red box corresponds to the top score one, and the yellow is the second best one. For each video we display five frames with equal temporal intervals.

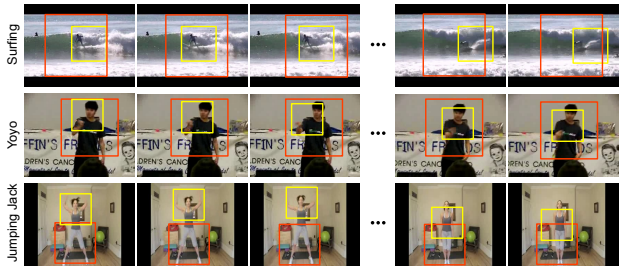

Figure 5: Visualization of the top two scored regions selected by M-ROI(10)-max. For each video clip we display first three and last two frames and omit the between. The red boxes correspond to the video tube with best action score, and the yellow is the one with second best score.

References

[1] Ross Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015.

[2] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, pages 3192–3199, December 2013.

[3] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, pages 2556–2563. IEEE, 2011.

[4] Xiaojiang Peng, Limin Wang, Yu Qiao, and Qiang Peng. Boosting vlad with supervised dictionary learning and high-order statistics. In *ECCV*, pages 660–674. Springer, 2014.

[5] Xiaojiang Peng, Limin Wang, Xingxing Wang, and Yu Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *arXiv preprint arXiv:1405.4506*, 2014.

[6] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. Action recognition with stacked fisher vectors. In *ECCV*, pages 581–595. Springer, 2014.

[7] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576, 2014.

[8] K. Soomro, A. Roshan Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. In *CRCV-TR-12-01*, 2012.

[9] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. *ICCV*, 2014.

[10] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *ICCV*, pages 3551–3558, 2013.

[11] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao. Towards Good Practices for Very Deep Two-Stream ConvNets. *ArXiv e-prints*, July 2015.

[12] Limin Wang, Yu Qiao, and Xiaoou Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, pages 4305–4314, 2015.

[13] Xiaolong Wang, Ali Farhadi, and Abhinav Gupta. Actions˜ transformations. *arXiv preprint arXiv:1512.00795*, 2015.

[14] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid. Learning to detect motion boundaries. In *CVPR*, pages 2578–2586, 2015.

[15] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference*, pages 461–470. ACM, 2015.

[16] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, pages 4694–4702, 2015.

[17] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, pages 391–405. Springer, 2014.