Improved Dense Trajectory with Cross Streams

Katsunori Ohnishi
Graduate School of Information Science and Technology
The University of Tokyo
ohnishi@mi.t.u-tokyo.ac.jp

Masatoshi Hidaka
Graduate School of Information Science and Technology
The University of Tokyo
hidaka@mi.t.u-tokyo.ac.jp

Tatsuya Harada
Graduate School of Information Science and Technology
The University of Tokyo
harada@mi.t.u-tokyo.ac.jp

ABSTRACT

Improved dense trajectories (iDT) have shown great performance in action recognition, and their combination with the two-stream approach has achieved state-of-the-art performance. It is, however, difficult for iDT to completely remove background trajectories from video with camera shaking. Trajectories in less discriminative regions should be given modest weights in order to create more discriminative local descriptors for action recognition. In addition, the two-stream approach, which learns appearance and motion information separately, cannot focus on motion in important regions when extracting features from spatial convolutional layers of the appearance network, and vice versa. In order to address the above mentioned problems, we propose a new local descriptor that pools a new convolutional layer obtained from crossing two networks along iDT. This new descriptor is calculated by applying discriminative weights learned from one network to a convolutional layer of the other network. Our method has achieved state-of-the-art performance on ordinal action recognition datasets, 92.3% on UCF101, and 66.2% on HMDB51.

Keywords

Action recognition; Video representation; Local descriptor

1. INTRODUCTION AND RELATED WORK

Video representation is becoming increasingly important in today's online environment in which a massive amount of videos are uploaded on a daily basis. Various approaches have been proposed to efficiently and accurately represent the videos.

Dense trajectories [18] and improved dense trajectories (iDT) [19] have dominated action recognition. Extracting hand-crafted features [1, 10, 2] along these trajectories can provide effective local descriptors, and encoding these local descriptors with a Fisher vector (FV) [12] or a vector of locally aggregated descriptors (VLAD) [6] can provide an effective video representation [7].

Figure 1: Illustration of visualized iDT and feature map from convolutional layer in temporal net. We can see that there are some noisy trajectories in background due to camera shaking.

Fueled by the recent success of convolutional neural networks (CNN) in image classification, video representations based on CNN have also been developed in action classification. The two-stream approach [13] is one of the most successful methods that learns appearance information and motion information separately using one network whose input is RGB and the other network whose input is optical flow. The idea of this separate learning has been widely used in later works [5, 20, 22, 23, 25, 26].

Aiming at fully end-to-end learning, three-dimensional CNN learning methods that can capture spatial and temporal information simultaneously and automatically [16, 17] have been developed recently. However, three-dimensional CNN learning is still a very difficult task, and these methods have not yet achieved comparable performance to the state-of-the-art approach.

Trajectory-pooled deep-convolutional descriptors (TDD) [20] have shown state-of-the-art performance in action recognition by pooling convolutional two-stream layers along iDT. Because the convolutional layer retains position information, it is possible to combine it with iDT. However, TDD, which is based on iDT and the two-stream approach, has two main shortcomings: (1) as shown in Figure 1, iDT cannot completely remove the background image for videos captured by a shaking camera. This can be solved by giving modest weights to background trajectories. (2) Although each network in the two-stream approach captures important infor-

MM '16, October 15-19, 2016, Amsterdam, Netherlands

DOI: http://dx.doi.org/10.1145/2964284.2967222

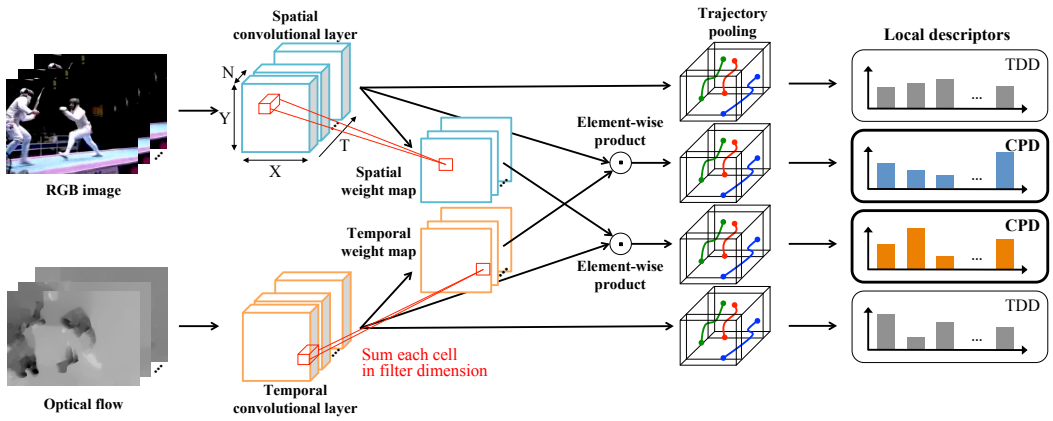

Figure 2: Illustration of the proposed local descriptors, named cross-stream pooled descriptors (CPD). After extraction, we encode these descriptors layer by layer and classify each of them. We then obtain final scores by simply summing all of their scores.

mation for action recognition, separate CNN learning sometimes lacks other important information that can be obtained only when spatial and temporal information are combined together.

For example, thinking about the action of pitching a ball, it is difficult for spatial CNN to focus on the region around the pitcher's hand from only a single RGB image. This shortcoming makes it difficult to discriminate between similar actions such as the difference between a soccer penalty kick and a field-hockey penalty shot, or between pitching a ball, a cricket shoot, a tennis serve, and a volleyball spike, especially when no background or context information is in the movie. Although iDT helps to solve this problem when recognizing action, iDT trajectories are hand-crafted so that they still contain both discriminative and non-discriminative trajectories. However, focusing on motion-important regions helps to extract more discriminative appearance features. When seeing a field hockey penalty, for example, the motion-important region is around the shooter's stick and the kipper. Extracting appearance features around these regions enables us to better recognize whether the player uses their own leg or the hockey stick or whether the kipper wears a protector or not. This can be also said in the case where spatial CNN and temporal CNN are reversed.

In order to address the above-mentioned problems, we utilize both networks in a two-stream approach by crossing two networks. Convolutional layers in spatial CNNs provide discriminative appearance features with position information while those in temporal CNNs provide discriminative motion features with position information. Thus, we propose a new descriptor that uses one network for the weights and gives these to the other network and pools along iDT, named cross-stream pooled descriptors (CPD). This is equivalent to pooling a convolutional layer of one network along iDT weighted by the convolutional layer of the other network, which leads to giving modest weights to iDTs in less discriminative regions. Our method has improved the performance of TDD on the ordinal action recognition datasets, UCF101 [15] and HMDB51 [8].

2. ACTION RECOGNITION REVISITED

In this section, we describe previous works on which our method is based.

2.1 Improved Dense Trajectories

Improved dense trajectories (iDT) [19] are the improved version of dense trajectories [18], which can remove dense trajectories in background images considering camera motion. A video whose size is (V_x, V_y, T) contains trajectories P^k $(k = 1 \dots K)$:

$$P^k = \{(x_1^k, y_1^k, t_1^k), (x_2^k, y_2^k, t_2^k), \cdots, (x_L^k, y_L^k, t_L^k)\}, \quad (1)$$

where K is the number of trajectories in a video, (x_l^k, y_l^k, t_l^k) is the position of the lth point in trajectory P^k, and L is the length of trajectory. Following other works [3, 9, 19, 20], we set $L = 15$ in this paper.

2.2 Two-Stream Approach

The two-stream approach [13] is a method that learns spatial information from RGB images and temporal information from optical flow images with each CNN separately. Since it is extremely difficult for a temporal net to learn motion only with a single flow image, a sequence of ten frames are used as input. In this paper, we call the network learned from RGB images a 'spatial network' and a network learned from optical flow images a 'temporal network.'

2.3 Trajectory-Pooled Deep-Convolutional Descriptors

Trajectory-pooled deep-convolutional descriptors (TDD) [20] combine iDT and the two-stream approach and achieves state-of-the-art performance on the UCF101 dataset. Given a ReLU applied convolutional layer $C \in \mathbb{R}^{X \times Y \times N \times T}$ from the two-stream approach, two normalization methods are applied to C, where X and Y are the width and height of the convolutional layer, N is the number of channels, T is the length of the video, and $C \geq 0$. Spatial normalization provides that $\widetilde{C}_{st}$ and channel normalization provides that $\widetilde{C}_{ch}$:

$$\widetilde{C}_{st}(x, y, n, t) = C(x, y, n, t) / \max_{x,y,t} C(x, y, n, t), \quad (2)$$

$$\widetilde{C}_{ch}(x, y, n, t) = C(x, y, n, t) / \max_{n} C(x, y, n, t), \quad (3)$$

where (x, y) is the position of the convolutional layer, n is the channel number of the convolutional layer, and t is the time in the video.

These $\widetilde{C}_{st}$ and $\widetilde{C}_{ch}$ are pooled along iDT instead of the originally pooled features (HOG [1], HOF [10], and MBH

[2]). Given a normalized convolutional layer $\widetilde{C}_b^a$, which is the convolutional layer after applying spatiotemporal normalization or channel normalization ($b \in \{st, ch\}$) from spatial or temporal nets ($a \in \{sp, tmp\}$), proposed descriptors $TDD(P^k, \widetilde{C}_b^a) \in \mathbb{R}^N$ are obtained as follows:

$$TDD(P^k, \widetilde{C}_b^a) = \sum_{l=1}^{L} \widetilde{C}_b^a(\overline{(r_x \times x_l^k)}, \overline{(r_y \times y_l^k)}, t_l^k), \quad (4)$$

where $\overline{(\cdot)}$ is the rounding operation and $(r_x, r_y) = (X/V_w, Y/V_h)$. These descriptors are encoded by FV. The final video representation is obtained by concatenating encoded vectors from both normalization methods.

3. IDT WITH THE CROSS STREAMS

As described to this point, separate CNN learning cannot always focus on truly important regions to capture an action's characteristics. Additionally, improved dense trajectories (iDT) cannot completely eliminate background trajectories from videos whose capturing camera experiences large motions. We address these problems to improve recognition performance using two equivalent methods. In this section, we describe both approaches in order to evaluate whether each problem can be improved by each calculation.

3.1 Cross-Stream Pooling Along iDT

In order to enhance motion-important regions in a spatial convolutional layer and appearance-important regions in a temporal convolutional layer, we propose a new convolutional layer for iDT pooling: the cross-stream layer. As shown in Figure 2, we produce spatial and temporal convolutional layers element-wise and pool the resulting four-dimensional matrix along iDT. We call this method cross-stream pooled descriptors (CPD). However, since each of the nth filters in C^{tmp} and C^{sp} do not have the same meaning, the simple element-wise product $C^{tmp}(x, y, n, t) \times C^{sp}(x, y, n, t)$ might not work well. A convolutional layer shows large activation for discriminative regions. Thus, we can obtain a discriminative weight map $W \in \mathbb{R}^{X \times Y \times T}$ by simply taking the sum in the n-direction:

$$W^{tmp}(x, y, t) = \sum_{n=1}^{N} \widetilde{C}^{tmp}(x, y, n, t), \quad (5)$$

where $\widetilde{C}^{tmp}$ is a normalized layer calculated from C^{tmp} as in equations (2) and (3). With this motion-based weight map, we can enhance the normalized spatial convolutional layer $\widetilde{C}^{sp}$, which contains appearance information:

$$D^{sp}(x, y, n, t) = \widetilde{C}^{sp}(x, y, n, t) \times W^{tmp}(x, y, t). \quad (6)$$

D^{sp} represents new appearance features enhanced by motion-important regions.

Similarly to motion-based weights, we can obtain appearance-based weights W^{sp} from C^{sp}, and D^{tmp} is calculated in the same way. The term ' cross stream ' originated from this cross utilization of two networks.

We then pool this D along iDT as in equation (4) to obtain $CPD(P^k, D_b^a) \in \mathbb{R}^N$ as follows:

$$CPD(P^k, D_b^a) = \sum_{l=1}^{L} D_b^a(\overline{(r_x \times x_l^k)}, \overline{(r_y \times y_l^k)}, t_l^k). \quad (7)$$

Table 1: Performance of each layer type on the UCF101 split1 dataset using parameters $(D, K) = (64, 128)$ for FV and $(D, K) = (128, 64)$ for VLAD.

Convolutional layer type	FV	VLAD
(a) Spatial	81.2%	81.8%
(b) Temporal	84.7%	85.5%
TDD: (a) + (b)	90.7%	91.5%
(c) Spatial weighted by temporal	81.3%	82.9%
(d) Temporal weighted by spatial	85.3%	85.9%
CPD (**ours**): (c) + (d)	90.4%	91.6%
TDD + CPD (**ours**)	90.8%	92.0%

Table 2: The combination of convolutional layers resulting in each network on the UCF101 split1 dataset when VLAD is applied using parameters $(D, K) = (128, 64)$. (a), (b), (c), and (d) represent spatial, temporal, spatial weighted by temporal, and temporal weighted by spatial cases.

	(a)	(b)	(c)	(d)
Conv3	71.9%	77.6%	74.1%	77.7%
Conv4	78.2%	82.2%	78.5%	82.0%
Conv5	76.3%	82.8%	75.7%	81.2%
Conv3 + Conv4	79.0%	82.5%	80.3%	83.2%
Conv4 + Conv5	81.3%	85.5%	81.5%	85.8%
Conv3 + Conv4 + Conv5	82.2%	85.8%	83.3%	86.5%

3.2 Two-Stream Pooling Along Weighted iDT

We next consider our method from a different point of view. Cross-stream pooled descriptors (CPD) can also be calculated as follows. In order to give modest weights to trajectories in the background region, we take advantage of the rest of the network. A convolutional layer C^{tmp} obtained from a temporal CNN in the two-stream approach, for example, has discriminative motion features without losing position information. Using this C^{tmp} as the weight and giving this weight to iDT, we can obtain new trajectories that are emphasized if they are in the region that contains motion-discriminative trajectories and are less emphasized if they are in regions that contain less motion-discriminative trajectories. As in equation (5), we obtain a discriminative weight map W^{tmp} by taking the sum in the n-direction. Each trajectory is weighted by this map W^{tmp}; then, we can obtain the weighted iDT. As for motion-based weights, an iDT weighted by an appearance-based map is calculated in the same way. We then pool the normalized convolutional layer $\widetilde{C}^a$ along the emphasized iDT whose weights are calculated from $W^{\overline{a}}$ and obtain the CPD as follows:

$$CPD(P^k, \widetilde{C}_b^a, W_b^{\overline{a}}) = \sum_{l=1}^{L} W_b^{\overline{a}}(x_l^k, y_l^k, t_l^k) \times \widetilde{C}_b^a(\overline{(r_x \times x_l^k)}, \overline{(r_y \times y_l^k)}, t_l^k). \quad (8)$$

This is equivalent to $CPD(P^k, D_b^a)$.

4. EXPERIMENTS

4.1 Datasets and Settings

We conducted experiments on widely used action recognition datasets, UCF101 [15] and HMDB51 [8]. We chose VGG16 [14] as our CNN and utilized publicly available models [21] that had been already trained on UCF101. Because UCF101 has more variety of actions and videos, we used a model learned on UCF101 split 1 as the initial model for HMDB51 training.The learning rate and other training set-

Table 3: Mean accuracy of CPD and other baseline methods on HMDB51 and UCF101. The score[*1] of two-stream (VGG16) on HMDB51 in our calculation.

Algorithm		HMDB51	UCF101
iDT & FV [19]		57.2%	85.9%
Two stream [13]		59.4%	88.0%
TDD & FV [20]		63.2%	90.3%
Two stream (VGG16)		61.9%[*1]	91.4% [21]
Spatial net	(VGG16 w/o flip&crop)	39.7%	75.5%
Temporal net	(VGG16 w/o flip&crop)	53.6%	81.0%
Two stream	(VGG16 w/o flip&crop)	59.3%	87.6%
TDD (VGG16) & FV		63.2%	91.3%
TDD (VGG16) & VLAD		65.0%	92.0%
CPD & VLAD (**ours**)		65.2%	91.8%
TDD (VGG16) & VLAD	+ CPD (**ours**) & VLAD	**66.2%**	**92.3%**

tings were the same as the training settings for UCF101[21]. We chose the models that showed the best validation scores during training.

As the convolutional layer for pooling, we chose conv3_3, conv4_3, and conv5_3 from VGG16. We call these conv3, conv4, and conv5 in this paper, respectively. A final video representation of each layer was obtained by concatenating st-normed and ch-normed Fisher vectors following TDD[20]. We fused SVM scores from each layer by taking the sum. Note that, in consideration of the calculation cost, we did not use multi-scale CNN, unlike TDD, and did not apply flipping or cropping to input images, unlike the original two-stream approach.

4.2 Analysis

Parameters and Coding Methods: We found the best coding method and parameters for TDD and CPD with UCF101 split1. Some previous works [7, 24] showed that VLAD encoding is also effective for action recognition. Thus, we tried both FV and VLAD for encoding. Through numerous experiments, we found that the best parameters for FV coding were $(D, K) = (64, 128)$, and those for VLAD coding were $(D, K) = (128, 64)$, where D is the dimension after compression by PCA and K is the number of clusters. Details are given in the supplemental material owing to limited space here.

Convolutional Type Combination: Table 1 shows that weighting the convolutional layer heightens accuracy for every layer and method, and combining our method with TDD improves the recognition accuracy of TDD. It is also shown that VLAD is more effective for all convolutional layer types than FV.

Layer Combination on Each Network: Table 2 presents the combination patterns of convolutional layers in each network. In all network types, we can see that using all layers showed the best performance. Thus, we simply employed all of them.

4.3 Evaluation of CPD

Table 3 represents the action accuracy of CPD and related methods on UCF101 [15] and HMDB51 [8], which are widely used action recognition datasets. Note that we did not flip and crop input images when predicting, unlike the original TDD. Although the two-stream approach of VGG16 without flipping and cropping shows worse performance than that of the original two-stream approach, as denoted in Table 3, the performance of TDD with FV is improved by replacing the CNN with VGG16. Encoding VLAD instead of FV also improves recognition accuracy. We then combine the scores

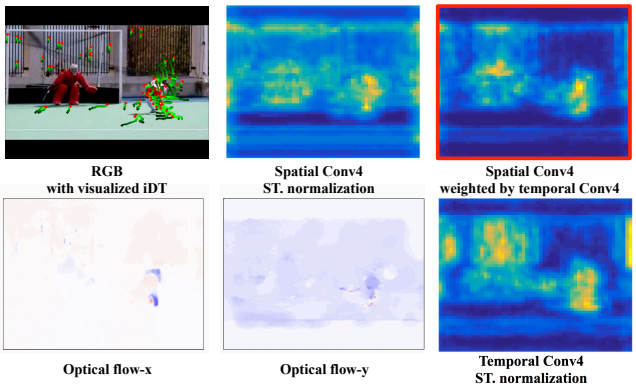

Figure 3: Sum of filter activations. We only show spatial layer weighted by temporal, because it will appear as the same image as temporal layer weighted by spatial with this visualization method.

Table 4: Comparison with the state-of-the-art methods. The scores written inside of () is the accuracy when combined with iDT & FV [19].

HMDB51		UCF101	
iDT & FV [19]	57.2%	iDT & FV [19]	85.9%
iDT & Stacked FV [11]	56.2%	C3D [17]	85.2%
+ iDT & FV	*(66.8%)*	+ iDT & FV	*(90.4%)*
F_{ST}CN [16]	59.1%	F_{ST}CN [16]	88.1%
LATE [3]	62.2%	MIFS [9]	89.1%
TDD & FV [20]	63.2%	TDD & FV [20]	90.3%
+ iDT & FV	*(65.9%)*	+ iDT & FV	*(91.5%)*
Video darwin [4]	63.7%	Hybrid LSTM [23]	91.3%
MIFS [9]	65.1%	Two stream (VGG16) [21]	91.4%
CPD (**ours**)	65.2%	CPD (**ours**)	91.8%
TDD + CPD (**ours**)	**66.2%**	TDD + CPD (**ours**)	**92.3%**

of this TDD using VLAD with those of CPD, which increases the performance of TDD both on UCF101 and HMDB51.

Fig. 3 shows an example of the visualized iDTs and convolutional layer activation. We can see that the spatial convolutional layer shows activation on many other objects whilst the convolutional layer weighted by the temporal convolutional layer shows activation mainly of the players. It can also be seen that some background iDTs still remain in the image due to camera shaking. However, the spatial layer weighted by the temporal layer activates mainly over the shooter and the kipper, ignoring their backgrounds. Thus, we can confirm that our method extracts appearance information mainly from motion-important regions and that these features capture different characteristics from those of TDD, which augments recognition performance.

4.4 Comparison with state-of-the-art

Table 4 shows the comparison of our method with other methods of action recognition on the UCF101 and HMDB51 datasets. On UCF101, the proposed method achieved state-of-the-art performance: 0.8% improvement over the combination of TDD [20] and iDT [19]. On HDMB51, our method achieved comparable performance to state-of-the-art methods. Considering the scores without adding iDT & FV [19], our method shows the best performance.

5. CONCLUSION

This study proposed a new type of local descriptors for action recognition, termed cross-stream pooled descriptors (CPD), that pools crossed convolutional layers along iDT. Our method achieved state-of-the-art performance on the widely used action recognition datasets UCF101 and HMDB51.

Acknowledgments

This work was supported by CREST, JST.

6. REFERENCES

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[2] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006.

[3] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Dynamically encoded actions based on spacetime saliency. In *CVPR*, 2015.

[4] B. Fernando, E. Gavves, J. M. Oramas, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, 2015.

[5] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann. Devnet: A deep event network for multimedia event detection and evidence recounting. In *CVPR*, 2015.

[6] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010.

[7] V. Kantorov and I. Laptev. Efficient feature extraction, encoding and classification for action recognition. In *CVPR*, 2014.

[8] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011.

[9] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, and B. Raj. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *CVPR*, 2015.

[10] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. *IJCAI*, 81:674–679, 1981.

[11] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *ECCV*. 2014.

[12] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007.

[13] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[15] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv:1212.0402*, 2012.

[16] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*, 2015.

[17] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[18] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *CVPR*, 2011.

[19] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.

[20] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015.

[21] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao. Towards good practices for very deep two-stream convnets. *arXiv:1507.02159*, 2015.

[22] X. Wang, A. Farhadi, and A. Gupta. Actions˜ transformations. In *CVPR*, 2016.

[23] 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 *ACMMM*, 2015.

[24] Z. Xu, Y. Yang, and A. G. Hauptmann. A discriminative CNN video representation for event detection. In *CVPR*, 2015.

[25] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015.

[26] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov. Exploiting image-trained cnn architectures for unconstrained video classification. In *BMVC*, 2015.