Exploring Fisher Vector and Deep Networks for Action Spotting

Zhe Wang[1] , Limin Wang[1,2] , Wenbin Du[1] , Yu Qiao[1]

[1]Shenzhen Institutes of Advanced Technology, CAS, China
[2]The Chinese University of Hong Kong, Hong Kong

June 12, 2015

Outline

Outline

Introduction

Figure: Examples of video clips in current action recognition dataset.

- Action recognition from a short video clip is widely studied in computer vision research (e.g. dataset of HMDB51, UCF101).
- Action spotting is more challenging as it needs to recognize and temporally localize the ongoing action in the long video sequence simultaneously.

LAP Action Recognition Challenge

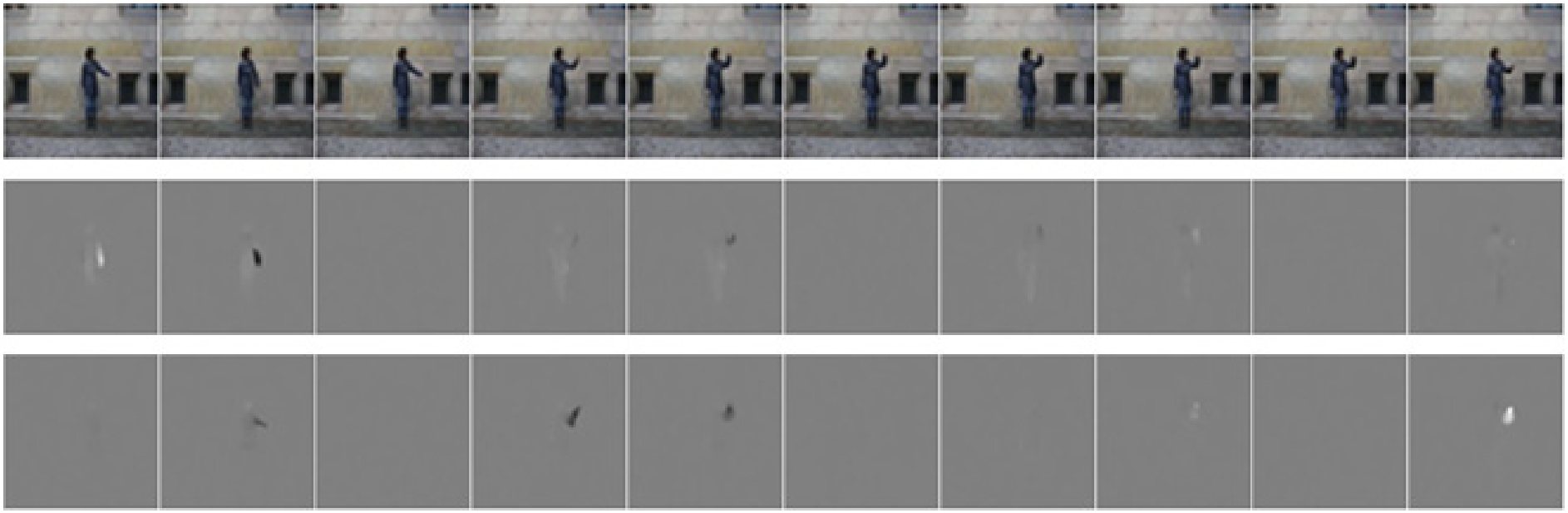


Figure: Examples of video images and their corresponding optical flow fields in Track 2.

- The track of action recognition challenge requires us segment the action instances in continuous video streams.
- This year also asks us to label actors for each action instance.

Outline

Overview of Method

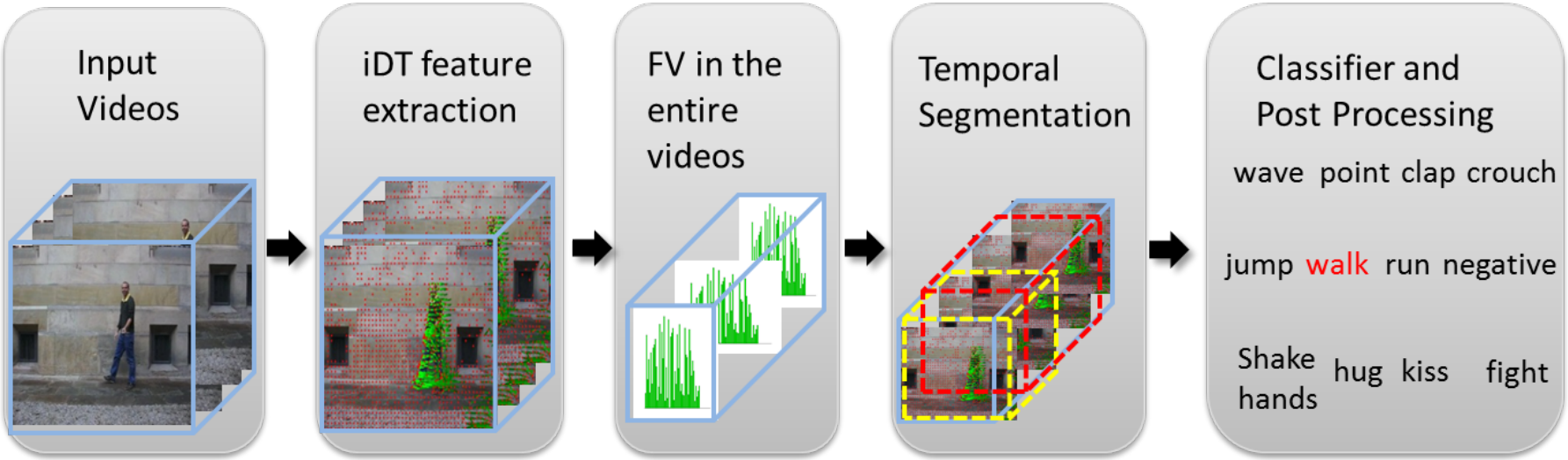


Figure: The architecture of exploring Fisher vector for action spotting

- The method follows our winner method of last ChaLearn LAP challenge.
- We make several modifications of our method to make it perform better

X. Peng , L. Wang , Z. Cai , Y. Qiao. *Action and Gesture Temporal Spotting with Super Vector Representation*, in ECCVW, 2014.

Step 1: iDT Feature Extraction

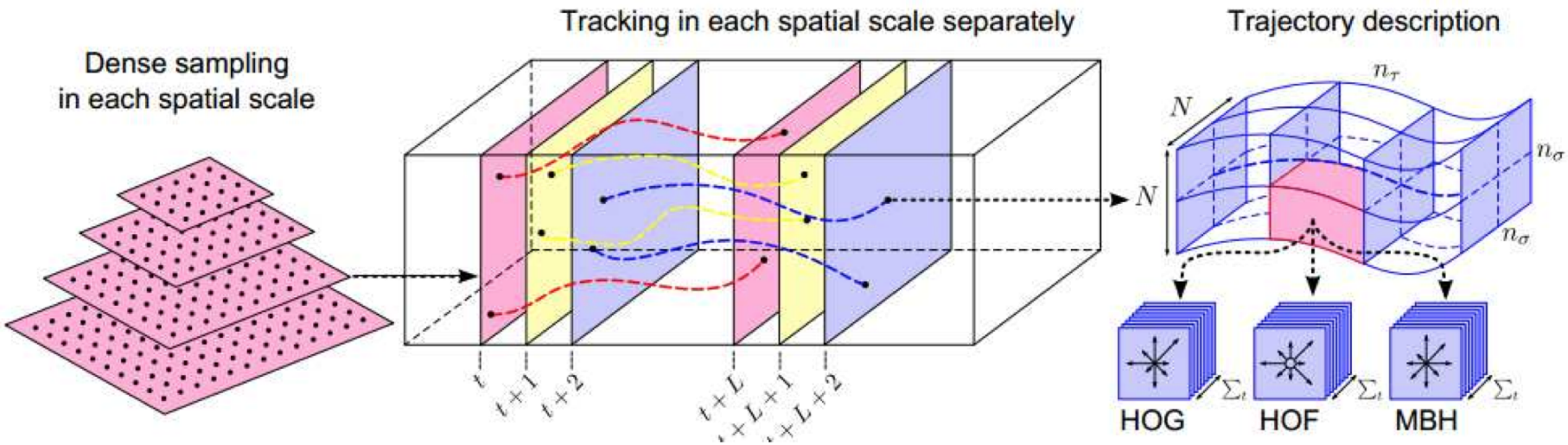
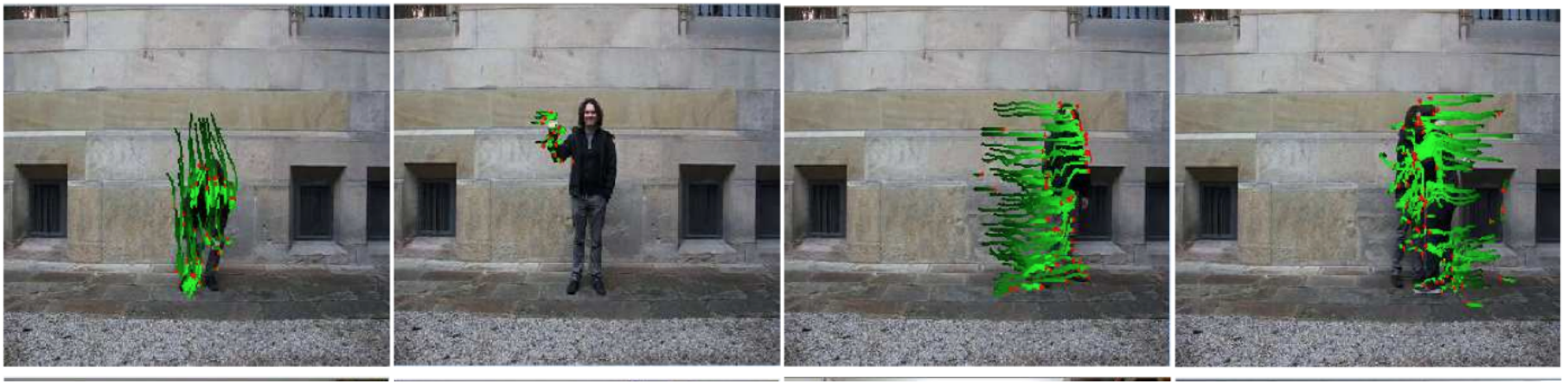


Figure: iDT feature extraction process.

We extract dense trajectory features and use four kinds of descriptors: HOG, HOF, MBHx, and MBHy.

Heng Wang and Cordelia Schmid *Action Recognition With Improved Trajecotires*, in ICCV, 2013.

Examples of iDT Features



Step 2: Fisher Vector Encoding

- We first perform PCA to de-correlate the dimensions of local descriptors and reduce dimensionality by a factor of 2.
- We then learn a generative GMM models:
  $p(\mathbf{x};\theta) = \sum_{i=1}^{K} \pi_i \mathcal{N}(\mathbf{x};\mu_k,\Sigma_k)$
- Then, the Fisher vector for each descriptor $\mathbf{x}$ is derived as follows:

$$\mathcal{G}_{\mu_i}^{\mathbf{x}} = \frac{1}{\sqrt{\pi_i}}\gamma_i(\mathbf{x})\frac{\mathbf{x}-\mu_i}{\sigma_i}$$

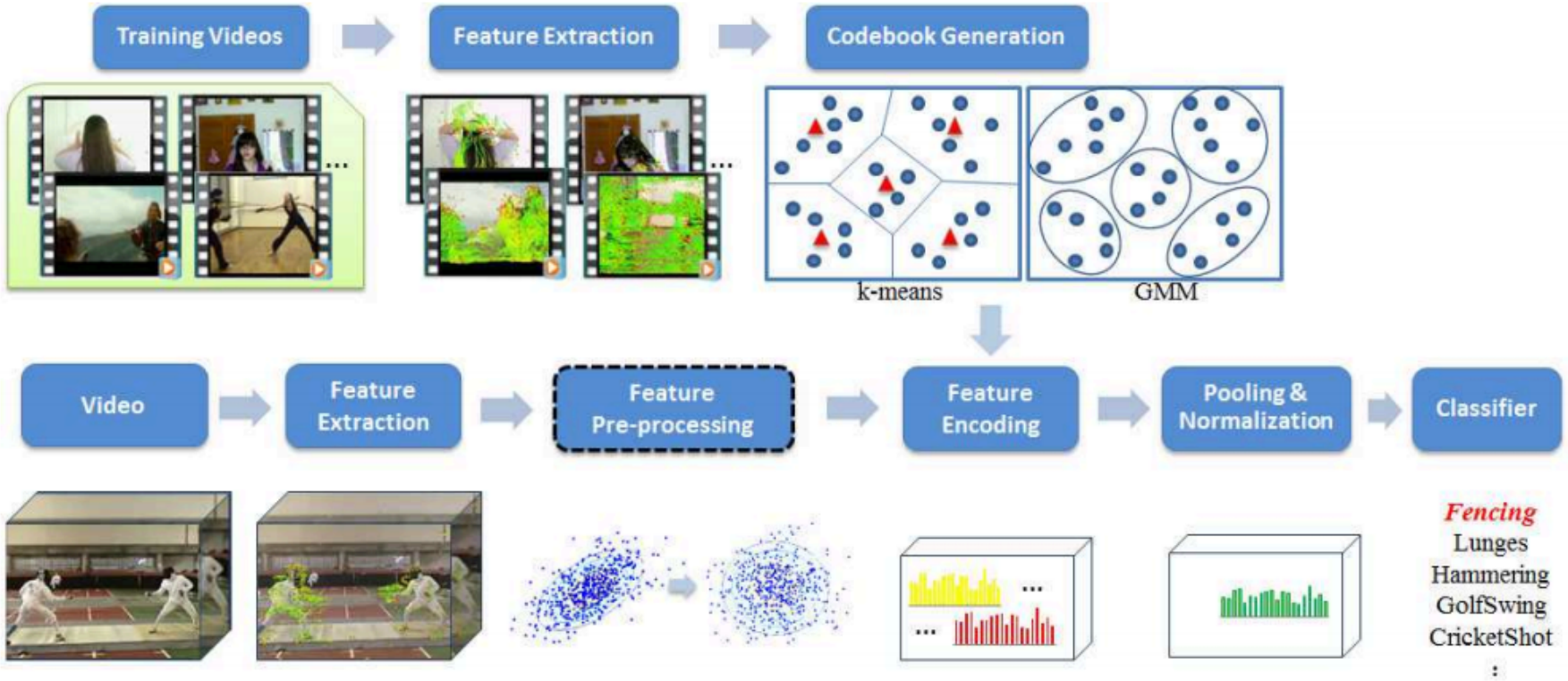
$$\mathcal{G}_{\sigma_i}^{\mathbf{x}} = \frac{1}{\sqrt{2\pi_i}}\gamma_i(\mathbf{x})\left[\frac{(\mathbf{x}-\mu_i)^2}{\sigma_i^2}-1\right]$$

J. Sanchez et. al. *Image Classification with the Fisher Vector: Theory and Practice*, in IJCV, 2013.

Step 3: Video Segmentation

- We resort to a temporal sliding window method to divide continuous video stream into short clips.
- According to the observation on training dataset, we set the window duration as 15-frames and the scanning step as 5-frames.
- We first extract the low-level features and encode them for the whole video stream.
- We then design a temporal integration histogram to efficiently calculate the feature histogram for the sub-window of any location and any duration.

Step 4: Clip Representation and Classification



- Bag of Visual Word (BoVW) has many choices for each step.
- Super vector encoding obtains higher performance.

Xiaojiang Peng, Limin Wang, Xingxing Wang, Yu Qiao, *Bag of Visual Words and Fusion Methods for Action Recognition: Comprehensive Study and Good Practice*. CoRR abs/1405.4506, 2014.

Step 4: Clip Representation and Classification

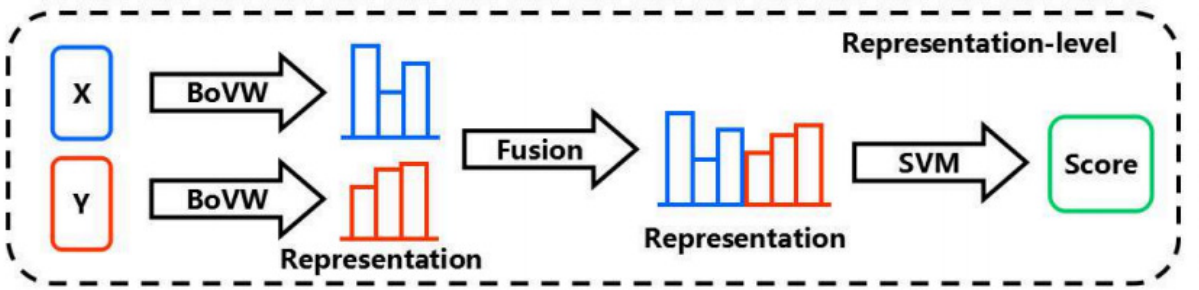


Figure: Feature fusion and classifier training.

- We separately construct Fisher vector representation for each kind of descriptor.
- We use the representation-level fusion method to combine these different descriptors.
- We adopt one-vs-all training scheme and train a linear SVM for each action class.

Step 5: Post-processing

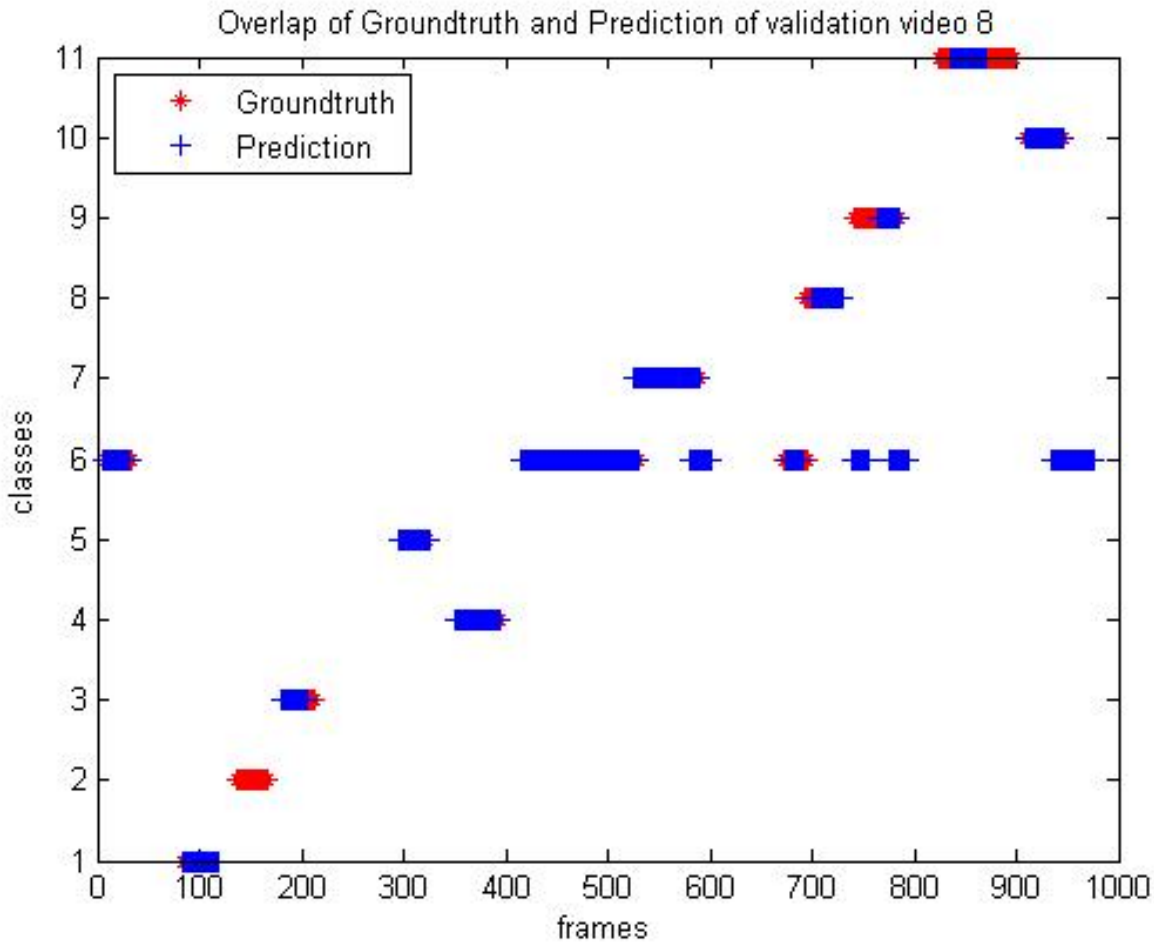
- In order to avoid the false detections, we design a post processing step.
- During training phase, we mine some instances of static background or noisy motions, and then train a classifier corresponding to background class.
- If a video clip is predicted as background class, we will remove this detection.
- We also use thresholds to eliminate those detections with low confidence score.

Modification of This Year

- Denser sampling of iDT features contributes to better classification result. We set iDT stride as 3 pixels and track length as 9 frames.
- The number of actors is related to action class: some action includes one actor (e.g. walking), some action includes two actors (e.g. hand shaking).
- We use a simple method to determine the user of action class. We firstly label all action with user 1.
- Then for those actions belonging to interactions such as hand shaking and hugging, we label them with both user 1 and user 2.

Modification of This Year (cont'd)

We introduce multiple thresholds for different action class. We set threshold as 0.40 for classifiers except action class 2 and set threshold as 0.55 for action class 2.



Modification of This Year (cont'd)

We make an attempt on applying deep networks on action spotting.

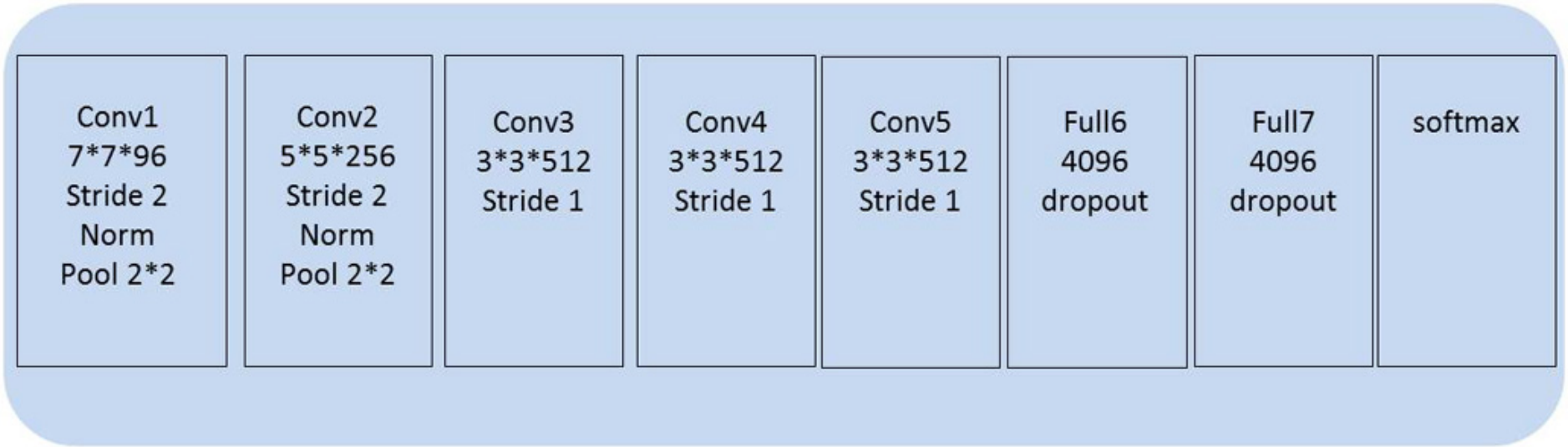


Figure: The architecture of deep networks (both spatial and temporal) for action spotting.

Karen Simonyan, and Andrew Zisserman, *Two-Stream Convolutional Networks for Action Recognition in Videos*, in NIPS, 2014.

Deep Networks for Action Spotting

- Spatial Net: its input is a single RGB image.
- Temporal Net: its input is the stacking optical flow fields (10-frames).
- Two-stream ConvNets: we fuse output of spatial net and temporal net.
- Training: we choose to pre-train them on the recently released two-stream CovNets model and then fine tune network weights on the challenge dataset.

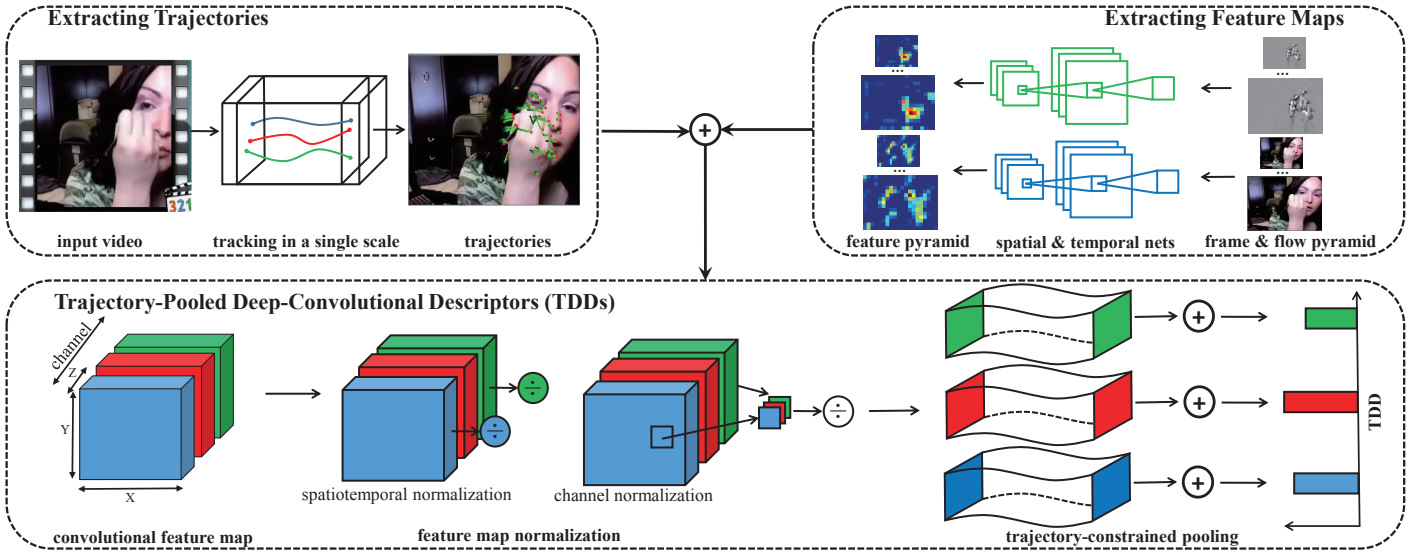
Limin Wang, Yu Qiao, and Xiaoou Tang *Action Recognition with Trajectory-Pooled Deep-Convolutional Descriptores*, in CVPR, 2015.

Trajectory-Pooled Deep-Convolutional Descriptors (TDD)

- Share the benefits of Improved Trajectories (hand-crafted features) and Two-Stream ConvNets (deep-learned features).
- Replace HOG, HOF, MBH descriptors with convolutional activations of CNNs.
- Fisher vector meeting CNN features.

Limin Wang, Yu Qiao, and Xiaoou Tang, *Action Recognition with Trajectory-Pooled Deep-Convolutional Descriptores*, in CVPR, 2015.

Trajectory-Pooled Deep-Convolutional Descriptors (TDD)



Algorithm	HMDB51	UCF101	Algorithm	HMDB51	UCF101
HOG [1]	40.2%	72.4%	Spatial conv4	48.5%	81.9%
HOF [1]	48.9%	76.0%	Spatial conv5	47.2%	80.9%
MBH [1]	52.1%	80.8%	Spatial conv4 and conv5	**50.0%**	**82.8%**
HOF+MBH [1]	54.7%	82.2%	Temporal conv3	54.5%	81.7%
iDT [1]	**57.2%**	**84.7%**	Temporal conv4	51.2%	80.1%
Spatial net [2]	40.5%	73.0%	Temporal conv3 and conv4	**54.9%**	**82.2%**
Temporal net [2]	54.6%	83.7%	TDD	63.2%	90.3%
Two-stream ConvNets [2]	**59.4%**	**88.0%**	TDD and iDT	**65.9%**	**91.5%**

Model and code is available at http://wanglimin.github.io/tdd/index.html

Outline

Experimental Setup

- Action/Interaction recognition task provides a dataset composed of 11 action classes and 9 untrimmed videos.
- The dataset is divided into three parts: development data(5 videos), validation data(2 videos), and evaluation data(2 videos).
- During develop phase, we train our model on the development data and verify the performance of different setting for method on the validation data.
- For final evaluation, we use both the development data and the validation data to re-train our model. The settings of our method are the same with the one during develop phase.

Implementation Details (Deep Networks)

- We pre-train the networks on the recently released model by CVPR paper.
- During develop phase, all images are resized to 256×256 and a 227×227 sub-image is randomly cropped from the image.
- The learning rate is initially set to 10^{-2} and decreases to 10^{-3} after 8k iterations, to 10^{-4} after 16k iterations.
- During evaluation phase, we use a multi-view voting method to classify each image, where we obtain 10 inputs by cropping and flipping four corners and the center of images, and average the recognition scores of these 10 inputs.
- **However, our current CNN based method does not obtain good results.**

Challenge Result

- Based on our Fisher vector method, we get a Jaccard Index of 53.85.
- This also verify the effectiveness of our visual analysis of multiple thresholds in post-processing, denser sampling of iDT features, and negative motion mining.

Rank	**Team**	**Score**
1	MMLAB(**Ours**)	0.5385
2	FKIE	0.5239

Table: Challenge results

Outline

Conclusions

- We explore action spotting from two aspects: Fisher vector and Deep networks.
- From our experimental results, Fisher vector based method shows its effectiveness with adjustment of denser iDT features sampling, negative motion mining, multi-threshold post processing and relationship between action and user.
- Our current CNN based method does not obtain good results. This might be ascribed to two facts. Firstly, We do not have sufficient data to train the temporal and spatial nets. Secondly, the supervised information or CNN training can be noisy, which harms the final performance.

What's Next

- Make more efforts on applying deep networks on action spotting and hope it is able to get better performance.
- Apply our proposed TDD features on top of current system.

Limin Wang, Yu Qiao, and Xiaoou Tang, *Action Recognition with Trajectory-Pooled Deep-Convolutional Descriptores*, in CVPR, 2015.

Model and code is available at http://wanglimin.github.io/tdd/index.html

Thank you!