Encoding Feature Maps of CNNs for Action Recognition

Xiaojiang Peng, Cordelia Schmid

Encoding Feature Maps of CNNs for Action Recognition

Xiaojiang Peng, Cordelia Schmid
Inria*
{xiaojiang.peng, cordelia.schmid}@inria.fr

Abstract

We describe our approach for action classification in the THUMOS Challenge 2015. Our approach is based on two types of features, improved dense trajectories and CNN features. For trajectory features, we extract HOG, HOF, MBHx, and MBHy descriptors and apply Fisher vector encoding. For CNN features, we utilize a recent deep CNN model, VGG19, to capture appearance features and use VLAD encoding to encode/pool convolutional feature maps which shows better performance than average pooling of feature maps and full-connected activation features. After concatenating them, we train a linear SVM classifier for each class in a one-vs-all scheme.

1. Introduction

Human action recognition in videos is a challenging problem due to difficulties like background clutter, viewpoint change, and various action styles. The THUMOS Challenge 2015 represents such challenging conditions. It consists of 5613 untrimmed test videos, where the action may be short compared to the video length, and multiple (including zero) instances can be present in each video. See [1] for more details.

The performance of an action recognition system strongly depends on the video representation. In this paper, we resort to two types of video representations, based on improved dense trajectory(IDT) [6] features and CNN feature maps. In the following, we describe our system for action classification in more detail.

2. System pipeline

Our system pipeline is shown in Figure 1. It consists of two type of video representations, one based on IDT and the other based on CNN feature maps. After normalization, they are concatenated and action classifiers are learned with linear SVMs in a one-vs-all scheme.

*LEAR team, Inria Grenoble Rhone-Alpes, Laboratoire Jean Kuntzmann, CNRS, Univ. Grenoble Alpes, France.

2.1. IDT based representation

Improved dense trajectories (IDT) based video representations have shown excellent performance on many action datasets [7]. IDT includes local appearance (HOG) and motion (HOF/MBH) descriptors. We rescale the videos to be at most 320 pixels wide, and skip every second frame to extract IDT features.We use a vocabulary of size 256 for GMM, and apply Fisher vector encoding separately for HOG, HOF, MBHx, and MBHy descriptors as [3][4]. We, then, normalize the resulting supervectors by power and intra normalization as suggested in [4], i.e., performing $\ell 2$ normalization for each FV block independently after power normalization.

2.2. CNN feature maps based representation

CNN features have become increasingly popular in action recognition [2] [9]. In [2], a video representation is obtained by average pooling of fc6 activation extracted for static frames every 10 frames. In [9], VLAD and Fisher vector are applied to fc6 activations and pool5 feature maps for event detection on TRECVID MED dataset. Following [9], we leverage VLAD and Conv5 feature maps for action recognition. We illustrate this idea in Figure 2. Considering a frame f_i and the Conv5 layer of CNNs, we can view the filters of Conv5 layer as feature extractors, and the pixels of Conv5 feature maps as local features of corresponding patches in f_i (see the pink squares in Figure 2). With these local features, which are called Latent Concept Descriptors in [9], we can apply any bag-of-words pipeline to obtain a video representation over all the frames in a video. In particular, we choose VLAD encoding to encode these local features considering its efficiency, and also apply power and intra normalization to the resulting VLAD vectors.

Similar in spiriting is the Trajectory-Pooled Deep-Convolutional Descriptor (TDD) [8]. Our approach presents several differences. First, instead of sampling all the pixels in the feature maps, they use trajectories to select pixels. Second, they extract local features with sum-pooling in 3D volumes aligned to the trajectories over consecutive feature maps. Third, they leverage an image pyramid to augment data. Finally, they use Fisher vectors for encoding

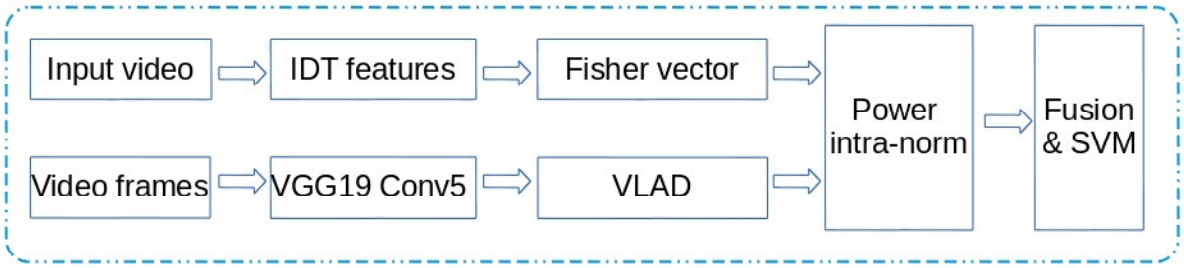

Figure 1. Overview of our system.

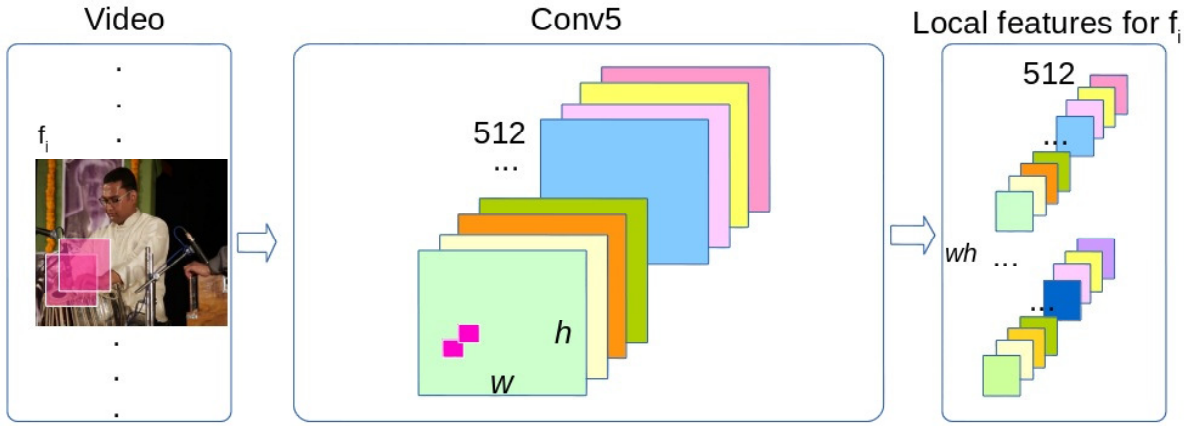

Figure 2. Local features from convolutional feature maps. Each pixel (pink square in the middle image) in the Conv5 feature map is actually a feature for the corresponding patch in original frame. We obtain w*h 512-D features for frame f_i.

and also conduct experiments on optical flow CNN feature maps. Our approach can be seen as a simplified scheme of TDD, which is more practicable on very large dataset such as the THUMOS Challenge 2015.

3. Experiments

In this section we present experimental results obtained on the validation set.

Settings. For the IDT based representation, we use the same setting as [2] except for the normalization (power and intra-norm). For the feature maps based representation, we utilize the VGG19 CNN model [5] with or without fine-tuning. VGG19 consists of 3 full-connected layers and 16 convolutional layers with fixed kernel size of 3x3. We fine tune the model on the UCF101 dataset and validation set. Note that fine-tuning on the validation set slightly bias our preliminary results and was done to speed up experimentation. Obviously, we do not fine tune on the test set. For fine-tuning, the learning rate is initially set to 10^{-3} and changed to 10^{-4} after 70K iterations, and training stopped after 20K iterations. We extract CNN features frame by frame on UCF101 dataset but every 5 frames on THUMOS validation and test set. All the sampled frames are rescaled to a fixed size of 224x224x3. We randomly select 256000 Conv5 local features and use k-means to construct a codebook of size 256 for VLAD. We split the validation set into 10 train/test folds. For each train/test fold we samples from each class with proportion 7/3. We report the mean and the standard deviation of the mAP score across these 10 folds. We also report the mAP for training on UCF101 and testing on the entire validation set.

3.1. Evaluation of CNN features

Here we use the Conv5_4 and fc6 layer of the VGG19 CNN model. We found that the performance decreased by using either later layers (e.g., fc7 and fc8) or earlier layers (e.g., pool4). Previous works mainly pool CNN features by average pooling or max pooling, here we provide a comparison of different pooling schemes as well as results for our final approach.

Table 1 and Table 2 present the results of different pooling methods and various CNN features without and with fine-tuning, respectively. The results in Table 1 are obtained by training on UCF101 dataset and testing on the validation set except for the last column. The last column in Table 1 and all the results in Table 2 are from 10-fold validation on the validation set. We did not perform VLAD on the fc6 features because it has shown worse results than on the Conv5 feature maps in [9]. Several findings can be concluded from these results. First, without fine-tuning, the

Table 1. Evaluation of Conv5_4 and fc6 layers without fine-tuning on the validation set.

	avg	max	VLAD	VLAD-folds
Conv5	46.02%	34.3%	56.95%	68.7 ± 1.1%
fc6	39.38%	28.38%	-	-

Table 2. Evaluation of Conv5_4 and fc6 layers with fine-tuning (10-fold on the validation set).

	avg	VLAD
Conv5	69.32 ± 1.1%	74.36 ± 1.3%
fc6	72.32 ± 1.1%	-

Conv5 based representation significantly outperforms the fc6 based one for action recognition. The main reason may be that fc6 of the VGG19 CNN model is trained to abstract concepts for object classification and is more related to object classes than action classes. Second, pooling Conv5 by VLAD encoding is much better than the other pooling methods. Finally, fine-tuning boosts the performance by 5.66% (74.36% vs. 68.7%).

3.2. Evaluation of feature combinations

We explore several combinations of IDT and CNN features in this section, see Table 3. Fusion is conducted by concatenating video representations. The performance of the IDT representation serves as our baseline. Combining IDT with our Conv5-VLAD representation improves mAP by 12.88% and 7.02% given the "Tr1" and "Tr2" train/test setting, respectively. The performance of CNN features is better than IDT in both settings except for fc6. Conv5-avg and fc6 based representation are complementary to IDT+Conv5-VLAD. We think that both Conv5-avg and fc6 based video representations contain appearance layout information which may not be captured by either IDT or Conv5-VLAD. That is the main reason why Conv5-VLAD outperforms Conv5-avg and fc6 when using "Tr1" train/test setting given different appearance in these train and test sets. When combining all representations, we obtain 79.52% mAP which is 10.33% better than the baseline by "Tr2" train/test setting.

In addition, we also extend the training set by adding the UCF101 dataset when conducting "Tr2" train/test experiments, which brings 3–7% gain depending on the representation. To suppress a response for videos not containing any action class, we threshold the response by τ. This threshold τ is determined on the training set as the minimum correct score.

Table 3. Fusion results with fine-tuning. "Tr1" denotes training on UCF101 dataset and testing on validation set, and "Tr2" is the 10-fold train/test scheme on validation set.

Fusion	Tr2	Tr1
IDT (HOG+HOF+MBH)	69.19±0.8%	52.23%
Conv5-VLAD	74.36±0.8%	63.87%
Conv5-avg	69.32±1.1%	57.47%
fc6	72.32±1.1%	47.07%
IDT+ Conv5-VLAD	76.21±1.0%	65.11%
IDT+ Conv5-avg	75.38±0.8%	62.95%
IDT+ fc6	76.1±1.0%	58.59%
IDT+Conv5-VLAD+Conv5-avg	77.69±0.9%	66.17%
IDT+ Conv5-VLAD+fc6	79.36±1.0%	64.84%
IDT+ Conv5-VLAD+Conv5-avg+fc6	**79.52**±1.1%	**66.64**%

Acknowledgements

This work was partly supported by the ERC advanced grant ALLEGRO, a MSR/INRIA joint project, and a Google Research Award.

References

[1] A. Gorban, H. Idrees, Y.-G. Jiang, A. Roshan Zamir, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://www.thumos.info/`, 2015.

[2] D. Oneata, J. Verbeek, and C. Schmid. The LEAR submission at Thumos 2014. In *ECCVW*, 2014.

[3] X. Peng, L. Wang, Z. Cai, Y. Qiao, and Q. Peng. Hybrid super-vector with improved dense trajectories for action recognition. In *ICCVW*, 2013.

[4] X. Peng, L. Wang, X. Wang, and Y. Qiao. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. *arXiv preprint arXiv:1405.4506*, 2014.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[6] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 103(1):60–79, 2013.

[7] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *arXiv preprint arXiv:1504.05524*, 2015.

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

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