Efficient feature extraction, encoding and classification for action recognition

Anonymous CVPR submission

Paper ID 1835

Abstract

Local video features provide state-of-the-art performance for action recognition. While the accuracy of action recognition has been continuously improved over the recent years, the low speed of feature extraction and subsequent recognition prevents current methods from scaling up to real-size problems. We address this issue and first develop highly efficient video features using motion information in video compression. We next explore feature encoding by Fisher vectors and demonstrate accurate action recognition using fast linear classifiers. Our method improves the speed of video feature extraction, feature encoding and action classification by two orders of magnitude at the cost of minor reduction in recognition accuracy. We validate our approach and compare it to the state of the art on four recent action recognition datasets.

1. Introduction

The amount of video has increased dramatically over recent yearsand continues to grow. Striking examples of this development include 6000 years of video uploaded to YouTube yearly[1] and millions of surveillance cameras installed only in the UK. According to Cisco[2], video is expected to dominate Internet traffic by 91% in 2014.

The access to information in such gigantic quantities of video data requires accurate and efficient methods for automatic video analysis. Much work has been recently devoted to automatic videounderstanding and recognition of human actions in particular [14, 16, 22, 31, 33, 36]. While the recognition accuracy has been continuously improved, current methods remain limited to relatively small datasets due to the low speed of video processing, often ranging in the order of 1-2 frames per second. This stands in a sharp contrast with the needs of large-scale video indexing and retrieval in modern video archives. Fast video recognition is also required by client applications, e.g., for automatic on-the-fly video moderation and video editing. Efficient video representations enabling fast event recognition will also foster solutions to new problems such as automatic clustering of very large video collections.

The main goal of this work is efficient action recognition. We follow the common bag-of-features action recognition pipeline [14, 33, 36] and explore the speed and memory trade-offs of its main steps, namely, feature extraction, feature encoding and classification. Given their success for action recognition, we represent video using motion-based HOF [14] and MBH [36] local descriptors. Motion estimation at dense grid, however, is a time-consuming process that limits the speed of feature extraction. In this work we avoid motion estimation and design fast descriptors using motion information from video compression. In contrast to the dense optical flow (OF), video compression provides sparse motion vectors only (we call it MPEG flow). As one contribution of this paper, we show that the use of sparse MPEG flow instead of the dense OF improves the speed of feature extraction by *two orders of magnitude* and implies only minor reduction in classification performance.

Feature encoding typically involves assignment of local descriptors to one or several nearest elements in a visual vocabulary. Given the large number of video descriptors, the speed of this step is a major bottleneck. We evaluate using kd-forest approximate nearest neighbor search [25] and analyze the associated trade-off between the speed and recognition accuracy. We next investigate Fisher vector (FV) encoding [23] and show improved action recognition while using fast linear classifiers. We evaluate the speed and accuracy of our approach on Hollywood-2 [18], UCF50 [27], HMDB51 [12] and UT-Interaction [30] benchmarks.

The rest of the paper is organized as follows. After reviewing related work in Section 2 we address efficient extraction of local video features in Section 3. Section 4 describes our fast and compact video encoding. Section 5 presents experimental results.

2. Related work

Recent methods show significant progress towards action recognition in realistic and challenging videos from YouTube, movies and TV [14, 15, 16, 22, 29, 31, 36]. Among other approaches, bag-of-features (BOF) meth-

[1] http://youtube.com/t/press_statistics
[2] http://newsroom.cisco.com/dlls/2010/prod_060210.html

ods [5, 13, 33] have gained popularity due to their simplicity, wide range of application and high recognition accuracy. BOF methods represent actions by collections of local space-timedescriptors aggregated over the video. Several alternative local descriptors have been proposed including histograms of flow orientations (HOF) [14], histograms of 3D gradients (HOG3D) [10, 34], motion boundary histograms (MBH) [4, 36], shapes of point trajectories [19, 26, 36], local trinary patterns [11, 37] and others. Mid-level features such as action attributes [16] and action bank [31] have also been explored. Recent comprehensive evaluation [36] suggests that MBH, HOF and HOG descriptors sampled along dense point trajectories result in great performance improving accuracy of other methods on a number of challenging datasets [36]. We follow [36] and design a new motion-based local descriptor that drastically improves the speed of previous methods at the cost of minor decrease in the recognition accuracy.

Efficient action recognition has been addressed by several methods. The work [1, 38, 39] is particularly related to ours as it makes use of motion information from video compression for fast action recognition. This previous work, however, designs action-specific descriptors and, hence, its speed scales linearly with the number of action classes. In contrast, we design a generic action representation and evaluate its accuracy and efficiency on many action classes in challenging settings. Yeffet and Wolf [37] extend the fast LBP image descriptor to a Local Trinary Pattern (LTP) descriptor in video and evaluate its accuracy on action recognition. While LTP was claimed to run in real time, no quantitative evaluation of its speed was reported in [37]. Differently to LTP, we use flow-based MBH and HOF descriptors which have recently shown excellent results for action recognition [36]. Yu et al. [40] have proposed another pixel-based local descriptor for efficient action recognition. We quantitatively compare our method with [40] and show improvements in both the speed and accuracy. Closely related to our work, [35] has recently improved the speed of local feature extraction in video by random feature sampling. We experimentally compare our method to [35] and show one order of magnitude improvement in speed while also demontrating improved accuracy.

Alternative schemes for feature encoding representations, have been recently evaluated for image classification in [3]. Fisher vector (FV) encoding [24] has been shown to provide best accuracy using efficient linear kernels for classification. FV encoding has been successfully applied for event detection [28] and we are confirming its improved performance and efficiency compared to the histogram encoding typically used in action recognition. We also investigate efficient computation of FV using approximate nearest neighbor methods for descriptor assignment.

FV encoding enables high recognition accuracy using fast linear classifiers which is a big advantage for large-scale video recognition.

Contributions

The contributions of this work are the following. First, we design and thoroughly evaluate an efficient motion descriptor based on the video compression which is 100x faster to compute at a minor degradation of recognition performance compared to the state-of-the-art [36]. Second, to the best of our knowledge, we are the first to evaluate performance of FV and VLAD for action recognition and find it improving the recognition rates without loss in speed compared to the histogram encoding.

3. Efficient video features

Dense Trajectory (DT) features together with MBH and HOF descriptors achieve state-of-the-art accuracy in action recognition [36] at the cost of high computational requirements. The analysis in [36] indicates that most of the running time (52%) is spent on the computation of optical flow, while the second most expensive operation (26%) is aggregation of dense flow measurements into histogram descriptors. In this paper we alleviate the expense of both of these steps by (i) re-using motion estimates available from video compression and (ii) constructing descriptors from very sparse motion measurements. In this section we first analyze the quality of motion vectors available in compressed video representations and then describe our efficient video descriptor.

3.1. Motion field from video compression

Consequent video frames are highly redundant with most of the changes between frames typically originating from the object or camera motion. As the storage of sparse motion vectors is much more efficient compared the storage of pixel values, video compression schemes heavily rely on motion estimation and encode coarse motion information in compressed video representations such as MPEG. This motion field can be accessed at the time of video decompression without additional cost[3].

Motion estimation by video encoders is designed to optimize video compression size and may not necessarily correspond to the true motion in the video. To verify the quality of the motion field obtained from video compression (here called MPEG flow), we compare it with the ground truth flow as well as with the output of optical flow methods by Lucas and Kanade [17] and Farnebäck [6]. We choose these two methods since they are deployed in existing implementations of local motion descriptors [14] and [36] which we use in this work for comparison. Figure 1 illustrates the

[3]Full video decompression may not be required.

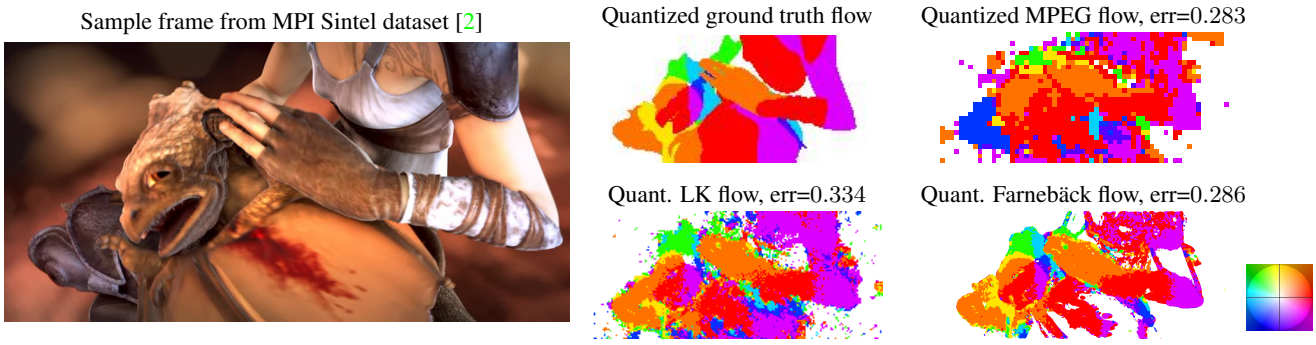

Figure 1. Comparison of optical flow estimation for a synthetic video sequence "bandage_1" from MPI Sintel dataset [2]. On the right we compare the ground truth flow with the flow obtained from DivX video compression (MPEG flow) as well as optical flow estimated with Lukas-Kanade [17] and Farnebäck [6] methods. The direction of flow vectors is quantized into eight equally distributed orientations and is color-coded according to the color-wheel on the bottom-right. White color represents no motion. The error of the flow is obtained by the ratio of incorrectly quantized flow values when compared to the ground truth and evaluated over the whole video. While the resolution of MPEG flow is lower compared to other methods, the accuracy of all three methods is compared both quantitatively and qualitatively. We submitted the MPEG flow maps to the Sintel website and obtained competitive error rates EPE all=11.148 and EPE matched=6.706 for the Clean subset that outperform methods [14],[15] reported at http://sintel.is.tue.mpg.de/results.

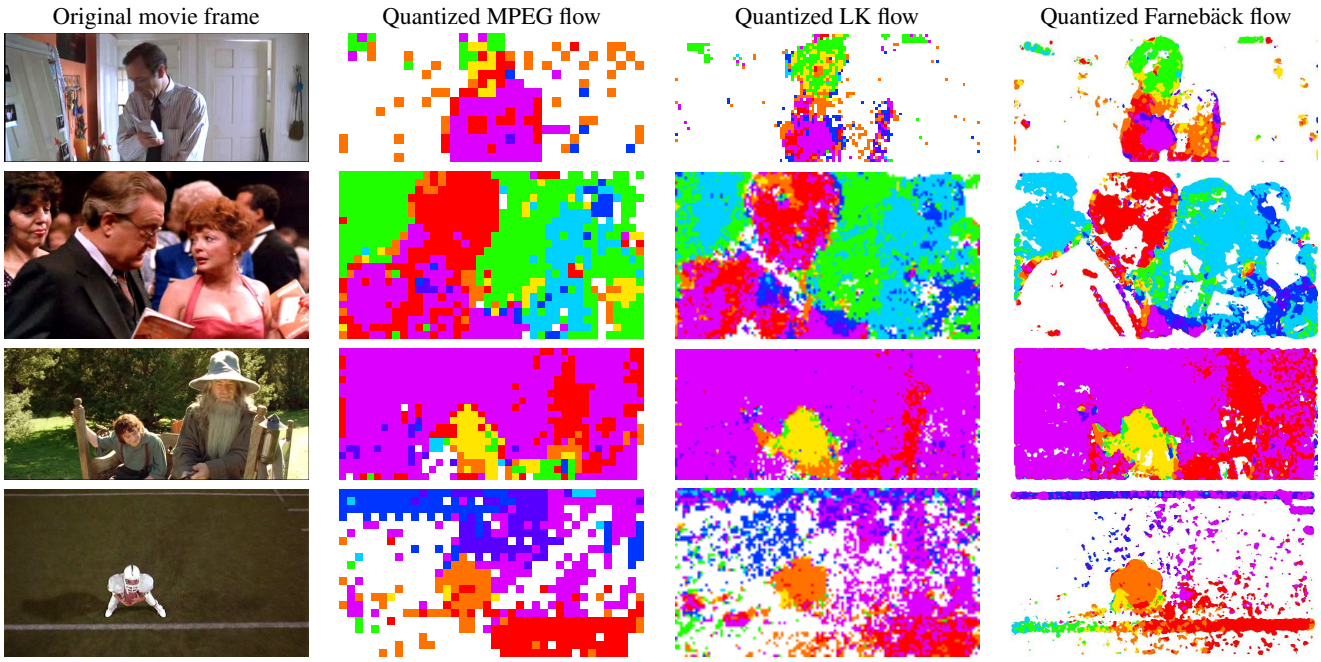

Figure 2. Qualitative comparison of optical flow in real movie frames using flow estimation methods in Figure 1. All methods produce noisy flow estimates. Quantized values of MPEG flow are consistent with the quantized output of the two optical flow algorithms. We compare quantized flow values since these are used in descriptor computation by our and other methods.

ground truth flow and automatic flow estimates obtained for a synthetic video sequence from the recent MPI Sintel dataset [2]. For this experiment we quantize flow fields into eight orientation values and one "no-motion" bin since this representation of flow is used in this paper for the construction of our fast video descriptor. The details and measurement results are in Figure 1.

Motion field obtained from MPEG flow has lower resolution compared to other methods. Meanwhile, the quantitative error of MPEG flow is comparable to the errors of two other OF methods. Qualitatively, the level of noise in MPEG flow and optical flow estimates is comparable both for synthetic and real video examples in Figure 2. This indicates that the substitute of the slow optical flow used in video descriptors [14, 36] by the "virtually free" MPEG flow may not have large implications on the performance

of subsequent recognition steps.

3.2. Compressed domain video descriptor

We follow the design of previously proposed space-time descriptors [14, 36] and define our descriptor by histograms of MPEG flow vectors accumulated in a video patch. Each patch is divided into cells as illustrated in Figure 3 and the normalized histograms from patch cells are concatenated into a descriptor vector. Constrained by the coarse 16×16 pixel spatial resolution of MPEG flow, we align descriptor grid cells with positions of motion vectors (red points in Figure 3). We also use bilinear interpolation of flow and increase the spatial resolution of the flow field by the factor of two (yellow points in Figure 3)[4].

Following [36], we compute HOF descriptors as histograms of MPEG flow discretized into eight orientation bins and a no-motion bin. For MBHx and MBHy descriptors the spatial gradients of the v_x and v_y components of the flow are similarly discretized into nine orientation bins. The final descriptor is obtained by concatenating histograms from each cell of the $2 \times 2 \times 3$ descriptor grid followed by l_2-normalization of every temporal slice. HOG descriptors are computed at the same sparse set of points.

The above scheme defines a $32 \times 32 \times 15$ pixel descriptor which we compute at every location of the video defined by the spatial stride of 16 pixels and temporal stride of 5 frames. To sample multiple spatial scales, we similarly define a $48 \times 48 \times 15$ pixel descriptor sampled with 24 pixels spatial stride. For a video of 640×480 pixels spatial resolution we obtain around 300 descriptors per frame. This is comparable to ≈ 350 dense trajectory features produced by the method in [36] for the same video resolution.

Main computational advantages of our descriptor originate from the elimination of the optical flow computation and from the coarse sampling of flow vectors. As we demonstrate experimentally in Section 5, these modifications imply drastic improvements of computational requirements at the cost of minor reduction in the classification performance.

4. Descriptor encoding

The purpose of descriptor encoding is to transform collections of local image or video descriptors $\{\mathbf{x}_i \dots \mathbf{x}_N\}$ into fixed-size vector representations. In this paper we investigate two descriptor encoding schemes and propose their computational improvements as described below.

4.1. Histogram encoding

Histogram encoding is a common method for representing video by local descriptors. First, for each descriptor type (HOG, HOF, MBHx, MBHy) a vocabulary of K visual words is constructed using the K-means algorithm. Then each descriptor is assigned to one of the visual words. Then the visual word indices are accumulated in a histogram. We follow [14] and use "space-time pyramid" which allows to capture the rough spatial and temporal layout of action elements. We split videos into six spatio-temporal grids ($x \times y \times t$: 1x1x1, 2x2x1, 1x3x1, 1x1x2, 2x2x2, 1x3x2x, a total of 24 cells). In total we have $C = 96$ channels $=$ 4 descriptor types $\times$ 24 cells, that is one channel per a pair of descriptor type and grid cell. Each cell accumulates its own histogram which is then l_1-normalized, thus the total histogram representation is of size $C \cdot K$. We also employ space-time pyramids for the Fisher vector and VLAD encoding methods described in sections 4.2 and 4.3 respectively.

For assignment we experimented with using brute-force nearest neighbors and with using the kd-trees from the FLANN library. The motivation for using kd-trees is the significant improvement in processing speed over the brute-force algorithm (see Subsection 5.2). The main parameters for the kd-trees method are the number of trees and the number of tests made during the descent over the trees.

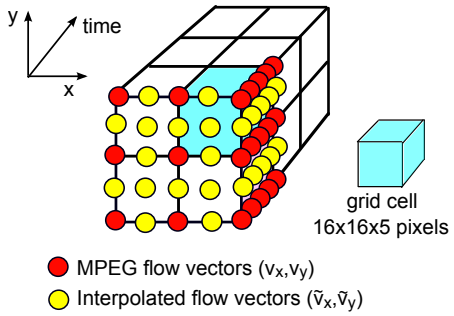

Figure 3. Compressed Domain (CD) video descriptor defined at positions of MPEG flow vectors. Each $2 \times 2 \times 3$ descriptor cell is accumulated from quantized values of original and interpolated flow vectors indicated by the red and yellow circles respectively.

4.2. Fisher vector

Fisher vector has been reported to consistently improve the performance for the image classification and image retrieval tasks [9]. Another advantage over the histogram encoding is that Fisher vectors allow for linear kernel classification. Fisher vector encoding assumes that descriptors are generated by a GMM model with diagonal covariance matrices. Similarly to histogram encoding, the GMM model of K Gaussians is first learned on the training set. Once the model $(\boldsymbol{\mu}_k, \boldsymbol{\sigma}_k)$ is learned, the Fisher vector representation of the descriptors $\{\mathbf{x}_1 \dots \mathbf{x}_N\}$ is given by the two parts [24]:

$$\mathbf{u}_k = \frac{1}{N\sqrt{\pi_k}} \sum_{i=1}^{N} q_{ki} \left(\frac{\mathbf{x}_i - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right)$$

[4] We found interpolation of the flow to be important in our experiments

Figure 4. Sample frames from action recognition datasets. Top: Hollywood-2 [18], Middle: UCF50 [27], Bottom: UT-Interaction [30].

$$\mathbf{v}_k = \frac{1}{N\sqrt{2\pi_k}} \sum_{i=1}^{N} q_{ki} \left[\left(\frac{\mathbf{x}_i - \boldsymbol{\mu}_k}{\boldsymbol{\sigma}_k} \right)^2 - \mathbf{1} \right] \qquad (1)$$

where q_{ki} is the Gaussian soft assignment of the descriptor $\mathbf{x}_i$ to the k-th Gaussian, π_k are the mixture weights, division and square are term-by-term operations. The $\mathbf{u}$ part captures the first-order differences, the $\mathbf{v}$ part captures the second-order differences. The final representation of size $C \cdot 2DK$ is given by concatenation of the two parts. As suggested by [24], at the end we take signed square root and l_2-normalize the result. We apply the same normalization scheme to VLAD, described next.

4.3. VLAD

The VLAD encoding is the simplified version of Fisher vector which considers only the first-order differences and assigns descriptors to a single mixture component. It was also shown to out-perform pure histogram encoding [9]. In our variant we keep the soft-assignment factor q_{ki}. The VLAD representation of size $C \cdot DK$ is then given by concatenating:

$$\mathbf{u}_k = \sum_{i:NN(\mathbf{x}_i)=\boldsymbol{\mu}_k} q_{ki} \left(\mathbf{x}_i - \boldsymbol{\mu}_k \right)$$

5. Experimental evaluation

In this section we evaluate the proposed descriptor and encoding schemes on Hollywood2 [18], UCF50 [27], HMDB51 [12] and UT-Interaction [30] action recognition benchmarks (see Figure 4) and compare the speed and accuracy of action recognition to the state of the art. We report classification results as mean average precision (mAP) for Hollywood2 and mean accuracy (acc) for UCF50, HMDB51 and UT-Interaction datasets. The processing speed is reported in frames-per-second (fps), run at a single-core Intel Xeon E5345 (2.33 GHz). We first evaluate the accuracy and the computational cost of the proposed descriptor and compare results with [36]. We then evaluate the speed, accuracy and compression of descriptor encoding.

To recognize actions we use SVM with multi-channel exponential χ^2-kernel [41] together with histogram-based action representations. For Fisher vector and VLAD encodings we use linear SVMs.

5.1. Descriptor evaluation

Table 1(Left) presents results of action recognition using the proposed compressed domain (CD) features and compares performance to Dense Trajectories (DT) baseline [36]. For the full combination of four descriptors the performance of DT (60.0%) is approximately four per-cent higher compared to CD features (56.2%). When comparing the speed of feature extraction for both methods measured on videos with 640×480 pixels spatial resolution, our method achieves 168fps which is about 7 times faster than real-time and 140 times faster compared to [36]. The speed measurements in this experiment include the time of feature computation as well as the time for video reading and decompression. Most of the time for our method is spent on the computation of integral histograms and aggregation of histogram-based video descriptors. In order to study time saving in detail, we contrast the execution times of our method run with parameters similar to [36] and with parameters used for all other experiments in this subsection. We save 66% by avoiding OF computation, 4% by sparser descriptor sampling and 29% by the aggregation of sparse motion vectors. The run-time of our descriptor is, hence, $< 1\%$ compared to [36]. Reducing the number of descriptor types increases the speed of our method to 347fps (14 times real time) at the cost of 9% drop in mAP.

DT features [36] are computed along point tracks which is different to our features computed inside the fixed-size

	Classification (mAP)		Speed (fps)	
	CD (our)	DT [36]	CD (our)	DT [36]
HOF	47.2%	52.9%	346.8	
MBHx	49.0%	52.0%	330.3	
MBHy	50.4%	56.1%	330.3	
HOF+MBHx+MBHy	53.9%	58.9%	218.7	
HOF+MBHx+MBHy+HOG	56.2%	60.0%	168.4	1.2

	DivX	x264
1024 kbit/s	54.3%	54.4%
512 kbit/s	54.5%	54.9%
256 kbit/s	54.2%	55.3%
128 kbit/s	53.4%	54.8%

Table 1. **Left:** Evaluation of action classification accuracy and the speed of feature computation for Hollywood2 action recognition benchmark. The speed of feature computation is reported for video of spatial resolution 640×480 pixels. **Right:** Evaluation of action recognition accuracy in Hollywood2 under the variation of video compression in terms of video codecs and compression bit-rates. The results are reported for CD descriptors in combination with histogram encoding.

	mAP
HOF+MBHx+MBHy+HOG (V0)	58.0%
HOF+MBHx+MBHy+HOG (V*)	58.9%
HOF+MBHx+MBHy+HOG [36]	60.0%
HOF+MBHX+MBHY+HOG+TRAJ [36]	60.3%

Table 2. Action classification accuracy for different versions of the dense trajectory features [36].

video patches. To identify the reason for 4% mAP drop of our method compared to [36] we simplify DT features by first approximating free-shape trajectories in DT features by straight lines (DT V*). In the second simplification we remove trajectory information from DT descriptor and compute DT features in fixed axes-parallel cuboids (DT V0). The results of these modifications in the first two lines of Table 2 indicate 1% drop due to DT V* and 1% further drop in mAP due to DT V0. Explicitly including the shape of trajectories into the descriptor (last line of Table 2) does not improve performance significantly. From this experiment we conclude that the shape of trajectories does not contribute to other descriptors significantly. We believe that the difference in accuracy of DT features and our method should be in the denser and more accurate flow estimation deployed by DT. Per-class action classification comparison of CD and DT features is illustrated in Figure 5.

MPEG flow might be influenced by the types and parameters of video codecs. To investigate this aspect, we evaluate the performance of action recognition for two common video codecs (x264 and DivX 5.2.1) under varying video compression bit-rates. Results in Table 1(Right) indicate the stable performance of action recognition for different types of video codecs and decreasing video quality corresponding to low values of bit-rates.

5.2. Descriptor encoding evaluation

While our method achieves high speed descriptor computation, the feature quantization becomes a bottle-neck with nearest-neighbor (NN) quantization performing only at the rate of 10fps when using efficient l_2-distance implementation. To improve quantization speed, we experiment with tree-based approximate nearest neighbor (ANN) quantization schemes implemented in FLANN [21]. The trade-off between computational time and the accuracy of quantization measured on the final recognition task is illustrated in Figure 6. Notably, the quantization using four trees and 32 tests in ANN search does not degrade recognition performance and achieves factor ≈ 5 speed-up compared to NN. Further increase of the speed implies approximately linear degradation of classification accuracy.

We next investigated Fisher Vector (FV) [23] and VLAD encodings [8] described in Section 4. To the best of our knowledge, Fisher vector and VLAD encodings have not been applied to action classification earlier. We train a GMM model with $K = 256$ Gaussians [9]. Results in Figure 6 and Table 3 indicate improvements in both the speed and accuracy when using FV and VLAD encodings compared to the histogram encoding. As for the histogram encoding, FLANN provides considerable speed-up for FV and VLAD encodings.

Table 4 presents action recognition results for the UCF50 dataset. Similar to the Hollywood-2 dataset, the accuracy of our methods is only a few percent below the state-of-the-art results in [36] with FV and VLAD encodings providing improvements over the histogram encoding both in speed and accuracy. The overall speed improves the speed of [36] by two orders of magnitude. Table 5 presents results for the UT-Interaction dataset and compares our method with the efficient action recognition method [40]. Our method demonstrates better classification accuracy compared to [40] and improves the speed of [40] by the factor of 10.

In Table 6 we compare our results on the HMDB dataset to [35] that investigates the effect of feature sampling on the trade-off between computational efficiency and recognition performance. For the MBH descriptor our method obtains significant improvement both in speed and accuracy, despite the fact that the [35]'s system was evaluated on Intel i7-3770K (3.50 GHz) which runs 1GHz faster than our Intel

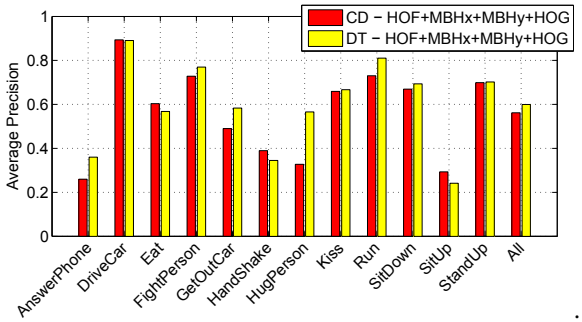

Figure 5. Per-class action classification results for DT [36] and our CD features using HOF+MBHx+MBHy+HOG descriptor combination and histogram encoding on the Hollywood2 benchmark.

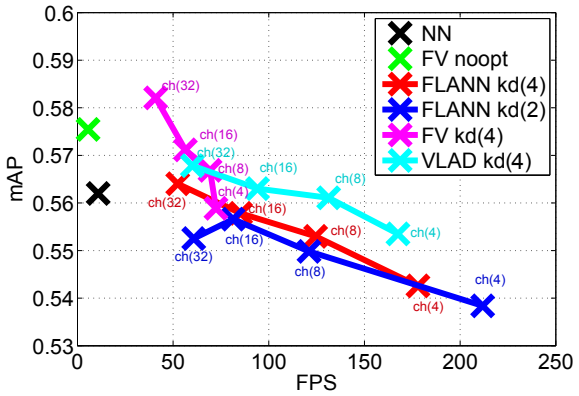

Figure 6. Performance of action recognition and the speed of descriptor encodings in Hollywood2 dataset for different types and parameters of encoding schemes. kd(x) indicates the number of trees x in the kd-forest used for ANN descriptor assignment. ch(y) indicates the number of tests y when descending the forest.

	Acc.	Feat. (fps)	Quant. (fps)	Total (fps)
CD FLANN(4-32)	55.8%		52.4	40.0
CD VLAD(4)	56.7%	168.4	167.5	84.0
CD FV(32)	58.2%		40.9	32.9
DT [36]	59.9%	1.2	5.1	1.0

Table 3. Comparison of our method to the state of the art on Hollywood2 dataset. The speed of [36] and of our method is reported for videos with spatial resolution 640 × 480 pixels.

	Acc.	Feat. (fps)	Quant. (fps)	Total (fps)
CD FLANN(4-32)	81.6%		52.4	48.1
CD VLAD(4)	80.6%	591.8	671.4	314.6
CD FV(32)	82.2%		171.3	132.8
DT [36]	85.6%	2.8	5.1	1.8

Table 4. Comparison of our method to the state of the art on the UCF50 dataset [27]. The speed is reported for videos with the spatial resolution 320 × 240 pixels.

Xeon E5345 (2.33 GHz). Both systems were evaluated on single-core machines.

	Acc.	Total FPS
FV(32)	87.6%	99.2
PSRM+BOST[40]	83.3%	10.0

Table 5. Accuracy and speed of action recognition in the UT-interaction dataset [30].

	Acc.	Feat. (fps)	Quant. (fps)	Total (fps)
CD ALL FV(32)	46.7%	455.6	129.7	101.0
CD MBH FV(32)	45.4%	683.3	268.0	192.5
CD ALL VLAD(32)	46.3%	455.6	455.6	227.8
MBH [35]	41.1%	33.9	267.1	30.8
HOG3D [35]	33.3%	49.6	290.8	42.2
DT [36]	48.3%	3.1		

Table 6. Comparison of our method to [35] on the HMDB dataset [12]. The speed is reported for videos with the spatial resolution 360 × 240 pixels

6. Conclusions

We present an efficient method for extracting and encoding local descriptors in video. We show that sparse motion vectors from video compression enable accurate action recognition at reduced computational cost. We next apply and evaluate Fisher vector encoding for action recognition and demonstrate the improved speed and accuracy. We also address the problem of high dimensionality of Fisher vector signatures, and show drastic improvements in the size of FV using PCA. Out method is fast and enables accurate action recognition using small signatures and linear classifiers.

References

[1] R. V. Babu and K. R. Ramakrishnan. Recognition of human actions using motion history information extracted from the compressed video. *Image and Vision Computing*, 22:597–607, 2004. 2

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, 2012. 3

[3] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011. 2

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

[5] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, pages 65–72, 2005. 2

[6] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In *SCIA*, 2003. 2, 3

[7] K. Grauman and R. Fergus. Learning binary hash codes for large-scale image search. In R. Cipolla, S. Battiato, and G. M. Farinella, editors, *Machine Learning for Computer Vi-*

sion, volume 411 of *Studies in Computational Intelligence*, pages 49–87. Springer Berlin Heidelberg, 2013.

[8] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, pages 3304–3311, 2010. 6

[9] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *PAMI*, 2012. 4, 5, 6

[10] A. Klaser, M. Marszałek, and C. Schmid. A spatio-temporal descriptor based on 3D-gradients. In *BMVC*, 2008. 2

[11] O. Kliper-Gross, Y. Gurovich, T. Hassner, and L. Wolf. Motion interchange patterns for action recognition in unconstrained videos. In *ECCV*, pages 256–269. 2012. 2

[12] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011. 1, 5, 7

[13] I. Laptev. On space-time interest points. *IJCV*, 64(2/3):107–123, 2005. 2

[14] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008. 1, 2, 3, 4

[15] I. Laptev and P. Pérez. Retrieving actions in movies. In *ICCV*, 2007. 1

[16] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, 2011. 1, 2

[17] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop*, pages 121–130, 1981. 2, 3

[18] M. Marszałek, I. Laptev, and C. Schmid. Actions in context. In *CVPR*, 2009. 1, 5

[19] P. Matikainen, M. Hebert, and R. Sukthankar. Trajectons: Action recognition through the motion analysis of tracked features. In *Workshop on Video-Oriented Object and Event Classification, ICCV 2009*, September 2009. 2

[20] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-Zero Cost. In *ECCV*, 2012.

[21] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISSAPP*, pages 331–340, 2009. 6

[22] J. Niebles, C.-W. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010. 1

[23] F. Perronnin and J. Sanchez. High-dimensional signature compression for large-scale image classification. In *CVPR*, 2012. 1, 6

[24] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, pages 143–156. 2010. 2, 4, 5

[25] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, pages 1–8, 2007. 1

[26] C. P. R. Messing and H. Kautz. Activity recognition using the velocity histories of tracked keypoints. In *ICCV*, 2009. 2

[27] K. Reddy and M. Shah. Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, pages 1–11, 2012. 1, 5, 7

[28] J. Revaud, M. Douze, C. Schmid, and H. Jégou. Event retrieval in large video collections with circulant temporal encoding. In *CVPR*, Portland, United States, 2013. IEEE. 2

[29] M. Rodriguez, A. Javed, and M. Shah. ction mach: a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008. 1

[30] M. S. Ryoo and J. K. Aggarwal. UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA), 2010. 1, 5, 7

[31] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. In *CVPR*, pages 1234–1241, 2012. 1, 2

[32] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek. Compressed Fisher Vectors for Large-Scale Image Classification. Research Report RR-8209, INRIA, 2013.

[33] C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, pages III:32–36, 2004. 1, 2

[34] P. Scovanner, S. Ali, and M. Shah. A 3-dimensional sift descriptor and its application to action recognition. In *ACM Conference on Multimedia*, 2007. 2

[35] F. Shi, E. Petriu, and R. Laganiere. Sampling strategies for real-time action recognition. In *CVPR*, pages 2595–2602, 2013. 2, 6, 7

[36] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Dense trajectories and motion boundary descriptors for action recognition. *IJCV*, 2013. 1, 2, 3, 4, 5, 6, 7

[37] L. Yeffet and L. Wolf. Local Trinary Patterns for human action recognition. In *ICCV*, pages 492–497, 2009. 2

[38] C. Yeo, P. Ahammad, K. Ramachandran, and S. S. Sastry. Compressed domain real-time action recognition. In *Multimedia Signal Processing*, 2006. 2

[39] C. Yeo, P. Ahammad, K. Ramchandran, and S. S. Sastry. High-speed action recognition and localization in compressed domain videos. *IEEE Transactions on Circuits and Systems*, 18:1006–1015, 2008. 2

[40] T.-H. Yu, T.-K. Kim, and R. Cipolla. Real-time action recognition by spatiotemporal semantic and structural forests. In *BMVC*, 2010. 2, 6, 7

[41] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 73(2):213–238, 2007. 5