Int J Comput Vis (2016) 119:239–253
DOI 10.1007/s11263-016-0905-6

EXMOVES: Mid-level Features for Efficient Action Recognition and Video Analysis

Du Tran[1] · **Lorenzo Torresani**[1]

Received: 15 May 2014 / Accepted: 4 April 2016 / Published online: 26 April 2016

Abstract In this paper we present EXMOVES—learned exemplar-based features for efficient recognition and analysis of actions in videos. The entries in our descriptor are produced by evaluating a set of movement classifiers over spatial-temporal volumes of the input video sequences. Each movement classifier is a simple exemplar-SVM trained on low-level features, i.e., an SVM learned using a single annotated positive space-time volume and a large number of unannotated videos. Our representation offers several advantages. First, since our mid-level features are learned from individual video exemplars, they require minimal amount of supervision. Second, we show that simple *linear* classification models trained on our global video descriptor yield action recognition accuracy approaching the state-of-the-art but at orders of magnitude lower cost, since at test-time no sliding window is necessary and linear models are efficient to train and test. This enables scalable action recognition, i.e., efficient classification of a large number of actions even in massive video databases. Third, we show the generality of our approach by training our mid-level descriptors from different low-level features and testing them on two distinct video analysis tasks: human activity recognition as well as action similarity labeling. Experiments on large-scale benchmarks demonstrate the accuracy and efficiency of our proposed method on both these tasks.

Communicated by Ivan Laptev, Josef Sivic, Deva Ramanan.

✉ Lorenzo Torresani
lorenzo@cs.dartmouth.edu

Du Tran
dutran@cs.dartmouth.edu

[1] Computer Science Department, Dartmouth College, 6211 Sudikoff Lab, Hanover, NH 03755, USA

Keywords Action recognition · Action similarity labeling · Video representation · Mid-level features

1 Introduction

Human action recognition and matching are important but still largely-unsolved computer vision problems motivated by many useful applications, including content-based video retrieval, automatic surveillance, and human-computer interaction. The difficulty of the task stems from the large intra-class variations in terms of subject and scene appearance, motion, viewing positions, as well as action duration.

We argue that most of the existing action recognition methods are not designed to handle such heterogeneity. Typically, these approaches are evaluated only on simple datasets involving a small number of action classes and videos recorded in lab-controlled environments (Blank et al. 2005; Veeraraghavan et al. 2006). Furthermore, in the design of the action recognizer very little consideration is usually given to the computational cost which, as a result, is often very high.

We believe that modern applications of action recognition demand scalable systems that can operate efficiently on large databases of unconstrained image sequences, such as YouTube videos. For this purpose, we identify three key-requirements to address (1) the action recognition system must be able to handle the substantial variations of motion and appearance exhibited by realistic videos; (2) the training of each action classifier must have low-computational complexity and require little human intervention in order to be able to learn models for a large number of human actions; (3) the testing of the action classifier must be efficient so as to enable recognition in large repositories, such as video-sharing websites.

This work addresses these requirements by proposing a global video descriptor that (at the time of this paper submission) yields state-of-the-art action recognition accuracy even with simple *linear* classification models. The feature entries of our descriptor are obtained by evaluating a set of movement classifiers over the video. Each of these classifiers is an exemplar-SVM (Malisiewicz et al. 2011) trained on low-level features (Laptev 2005; Wang et al. 2013) and optimized to separate a single positive video exemplar from an army of "background" negative videos. Because only one annotated video is needed to train an exemplar-SVM, our features can be learned with very little human supervision. The intuition behind our proposed descriptor is that it provides a semantically-rich description of a video by measuring the presence/absence of movements similar to those in the exemplars. Thus, a linear classifier trained on this representation will express a new action-class as a linear combination of the exemplar movements (which we abbreviate as EXMOVES). We demonstrate that these simple linear classification models produce surprisingly good results on challenging action datasets. In addition to yielding high-accuracy, these linear models are obviously very efficient to train and test, thus enabling *scalable* action recognition, i.e., efficient recognition of many actions in large databases.

Our approach can be viewed as extending to videos the idea of classifier-based image descriptors (Wang et al. 2009; Torresani et al. 2010; Li et al. 2010; Deng et al. 2011) which describe a photo in terms of its relation to a set of predefined object classes. To represent videos, instead of using object classes, we adopt a set of movement exemplars. In the domain of action recognition, our approach is most closely related to the work of Sadanand and Corso (2012), who have been the first to describe videos in terms of a set of actions, which they call the Action Bank. The individual features in Action Bank are computed by convolving the video with a set of predefined action templates. This representation achieves high accuracy on several benchmarks. However, the template-matching step to extract these mid-level features is very computationally expensive. As reported in Sadanand and Corso (2012), extracting mid-level features from a single video of UCF50 (Soomro et al. 2013) takes a minimum of 0.4 h up to a maximum of 34 h. This computational bottleneck effectively limits the number of basis templates that can be used for the representation and constrains the applicability of the approach to small datasets.

Our first contribution is to replace this prohibitively expensive procedure with a technique that is almost two orders of magnitude faster. This makes our descriptor applicable to action recognition in large video databases, where the Action Bank framework is simply too costly to be used. The second advantage of our approach is that our mid-level representation can be built on top of any arbitrary spatial-temporal low-level features, such as appearance-based descriptors computed at interest points or over temporal trajectories. This allows us to leverage the recent advances in design of low-level features: for example, we show that when we use dense trajectories Wang et al. (2013) as low-level features, a simple linear classifier trained on the HMDB51 dataset using our mid-level representation yields a 41.6 % relative improvement in accuracy over the Action Bank built from the same set of video exemplars. Furthermore, we demonstrate that our representation is general in the sense that it can be applied to different low-level features and it can be used for several video analysis tasks, such as action recognition and action similarity labeling. Finally, the experiments reported in this article show that a linear classifier applied to our mid-level representation produces consistently much higher accuracy than the same linear model directly trained on the low-level features used by our descriptor.

Our EXMOVES are also related to Discriminative Patches (Jain et al. 2013), which are spatial-temporal volumes selected from a large collection of random video patches by optimizing a discriminative criterion. The selected patches are then used as a mid-level vocabulary for action recognition. Our approach differs from this prior work in several ways. As discussed in 4.4, each EXMOVE feature can be computed from simple summations over individual voxels. This model enables the use of *Integral Videos* (Ke et al. 2010), which reduce dramatically the time needed to extract our features. Discriminative Patches cannot take advantage of the Integral Video speedup and thus they are much more computationally expensive to compute. This prevents their application in large-scale scenarios. On the other hand, Discriminative Patches offer the advantage that they are automatically mined, without any human intervention. EXMOVES require some amount of human supervision, although minimal (just one hand-selected volume per exemplar). In practice such annotations are inexpensive to obtain. In our experiments we show that EXMOVES learned from only 188 volumes greatly outperform Discriminative Patches using 1040 volumes.

2 Related Work

Human action recognition and analysis have a long history in the computer vision literature. The previous approaches can be roughly classified into low-level feature-based, mid-level feature-based, and top-level action modeling approaches.

2.1 Low-Level Feature-Based Approaches

Low-level feature-based approaches represent videos by low-level feature primitives. These features can be either sparsely or densely sampled from the videos. Spatio-temporal interest points can also be applied for sparse features. Efros et al. used optical flows to represent and classify actions (Efros et al. 2003). Laptev and Lindeberg extended the Harris corner detector to 3D in order to detect spatio-temporal interest points (STIPs) (Laptev and Lindeberg 2003; Laptev 2005). Dollár et al. used a 1D Gabor filter and a 2D Gaussian smoothing kernel to detect Cuboids for behavior recognition (Dollar et al. 2005). The Cuboids interest point detector is denser compared to STIPs and allows the users to adjust the desired level of sparsity. Gorelick et al. proposed Space-Time Shapes for modeling actions Blank et al. (2005) by solving a Poisson equation. Derpanis et al. used 3D Gabor filters to extract "space-time orientation" for action recognition Derpanis et al. (2010). Motivated by the success of image-based features such as HOG (Dalal and Triggs 2005) and SIFT (Lowe 2004), corresponding 3D descriptors were proposed for video, e.g., HOG3D (Scovanner et al. 2007) and SIFT3D (Klaser et al. 2008). Ke et al. used boosting to learn volumetric features for event detection (Ke et al. 2005). Quoc et al. demonstrated that spatio-temporal features can be learned under unsupervised setting using stacked ISA with strong performance (Le et al. 2011). Recently, Wang et al. proposed dense trajectories (Wang et al. 2011 and its improved version, namely improved Dense Trajectories Wang et al. (2013) which is widely considered the current state-of-the-art in video features for human action recognition, achieving the top performance on various benchmarks.

2.2 Mid-Level Feature-Based Approaches

Mid-level feature-based approaches represent videos using a set of mid-level features, which are usually classifiers trained on low-level representations. Fathi and Mori used Adaboost to train a set of mid-level weak classifiers for human action recognition (Fathi and Mori 2008) using optical flow as low-level features. Similarly, Ke et al. used a Boosting method to learn volumetric features for action detection (Ke et al. 2010), but directly on raw video voxels. Along the line of visual attributes for static images (Ferrari and Zisserman 2007; Lampert et al. 2009; Farhadi et al. 2009), Liu et al. proposed to represent human actions by data-driven attributes and used them for action recognition (Liu et al. 2011). Inspired by the success of ObjectBank (Li et al. 2010), Sadanand and Corso proposed to represent videos as a set of video templates called Action-Bank (Sadanand and Corso 2012). Despite its promising discriminative power, the high computational cost prevents this method to be applicable in large-scale scenarios. Jian et al. used Discriminative Patches to represent videos for action classification (Jain et al. 2013). The main benefit of this method is being trained in an unsupervised manner. However, due to the unsupervised nature of the training, the method does need to have a large number of mid-level features in order to attain a reasonable discriminative capacity (see results in our experimental section). Our EXMOVES are closely related to Action Bank and Discriminative Patches as these are all forms of mid-level representation. Compared to Action Bank, our mid-level classifiers are linear SVMs while Action Bank builds on template matching, which is much more computationally expensive. On the other hand, while Discriminative Patches are trained without supervision, our EXMOVES are weakly supervised. In our experiments we show that EXMOVES achieve better discriminative power while requiring a minimal amount of annotation, i.e., one annotated bounding box per mid-level classifier.

2.3 Top-Level Action Modeling Approaches

The top-level action modeling approaches use low-level feature representations but focus on top-level action modeling to improve the classification accuracy. Wang and Suter proposed the use of silhouettes to describe human activities (Wang and Suter 2007). Niebles and Fei-Fei used bag-of-word representation to model videos for action recognition (Niebles and Fei-Fei 2007). Tran et al. showed metric learning (Weinberger et al. 2006) can improve action recognition (Tran and Sorokin 2008). Laptev et al. used Boosting method to classify human action in realistic movies (Laptev and Prez 2007; Laptev et al. (2008)). Yuan et al. used mutual information maximization to detect and recognize actions in videos (Yuan et al. 2009; Yuan et al. (2011)). Yu et al. used random forest for indexing and retrieving actions in videos (Yu et al. 2011). Hu et al. used multiple-instance learning to detect human actions (Hu et al. 2009). Hough transform was also used to recognize actions (Yu et al. 2012).

Although many of these approaches have been shown to yield good accuracy on standard human action benchmarks, they are difficult to scale to recognition in large repositories as they involve complex feature representations or learning models, which are too costly to compute on vast datasets.

3 Approach Overview

We explain the approach at a high level using the schematic illustration in Fig. 1. During an offline stage, our method learns N_a exemplar-movement SVMs (EXMOVES), shown on the left side of the figure. Each EXMOVE is a binary classifier optimized to recognize a specific action exemplar (e.g., an instance of "biking") and it uses histograms of quantized space-time low-level features for the classification. Note that in order to capture different forms of each activity, we use multiple exemplars per activity (e.g., multiple instances of "biking"), each contributing a separate EXMOVE. The set of learned EXMOVES are then used as mid-level feature extractors to produce an intermediate representation for any new input video: we evaluate each EXMOVE on subvolumes of the input video in order to

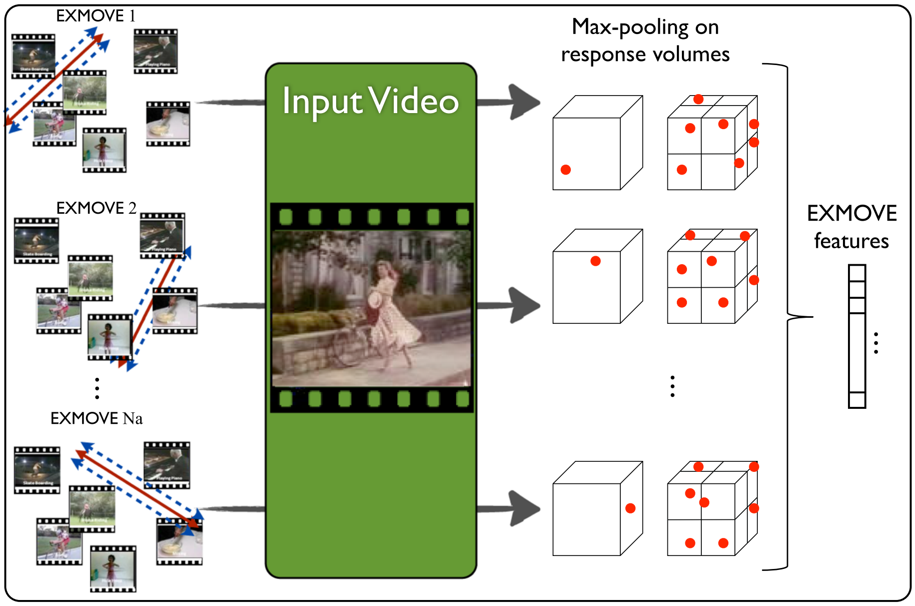

Fig. 1 Overview of our approach. During an offline stage, a collection of exemplar-movement SVMs (EXMOVES) is learned. Each EXMOVE is trained using a single positive video exemplar and a large number of negative sequences. These classifiers are then used as mid-level feature extractors to produce a semantically-rich representation of videos

compute the probability of the action at different space-time positions in the sequence. Specifically, we slide the subvolume of each EXMOVE exemplar at N_s different scales over the input video. As discussed in Sect. 4.4, this evaluation can be performed efficiently by using *Integral Videos* (Ke et al. 2010). Finally, for each EXMOVE, we perform max-pooling of the classifier scores within N_p spatial-temporal pyramid volumes. Thus, for any input video this procedure produces a feature vector with $N_a \times N_s \times N_p$ dimensions. Because the EXMOVE features provide a semantically-rich representation of the video, even simple linear classification models trained on our descriptor achieve good action categorization accuracy.

4 Exemplar-Movement SVMs (EXMOVES)

Our EXMOVE classifiers are linear SVMs applied to histograms of quantized space-time low-level features calculated from subvolumes of the video. In section 4.1 we describe the two space-time low-level descriptors used in our experiments, but any quantize-able appearance or motion features can be employed in our approach.

In principle, to train each SVM classifier we need a reasonable number of both positive and negative examples so as to produce good generalization. Unfortunately, we do not have many positive examples due to the high human cost of annotating videos. Thus, we resort to training each SVM using only one positive example, by extending to videos the exemplar-SVM model first introduced by Malisiewicz et al. for the case of still images (Malisiewicz et al. 2011). Specifically, for each positive exemplar, we manually specify a space-time volume enclosing the action of interest and excluding the irrelevant portions of the video. The histogram of quantized low-level space-time features contained in this volume becomes the representation used to describe the positive exemplar. Then, our objective is to learn a linear SVM that separates the positive exemplar from the histograms computed from all possible subvolumes of the same size in negative videos.

It may appear that training a movement classifier from a single example will lead to severe overfitting. However, as already noted in Malisiewicz et al. (2011), exemplar-SVMs actually have good generalization as their decision boundary is tightly constrained by the millions of negative examples that the classifier must distinguish from the positive one. In a sense, the classifier is given access to an incredible amount of training examples to learn what the positive class is *not*. Furthermore, we use the exemplar-SVMs simply as mid-level feature extractors to find movements similar to the positive exemplar. Thus, their individual categorization accuracy is secondary. In other words, rather than applying the individual exemplar-SVMs as action recognizers, we use them collectively as building blocks to define our action categorization model, in a role similar to the weak-learners of boosting techniques (Viola and Jones 2001).

4.1 Low-Level Features Used in EXMOVES

Although any arbitrary low-level description of space-time points or trajectories can be used in our framework, here we experiment with the two following representations:

- **HOG-HOF-STIPs**. Given the input video, we first extract spatial-temporal interest points (STIPs) (Laptev 2005). At each STIP we compute a histogram of oriented gradients (HOG) and a histogram of flows (HOF) (Dalal et al. 2006) using the implementation in Laptev (2005). We concatenate the HOG and the HOF descriptor to form a 162-dimensional vector representing the STIP. Finally, we run k-means on these vectors to learn a codebook of $D = 5000$ cluster centroids. Given the codebook, any space-time volume in a video is represented in terms of the histogram of codewords occurring within that volume. We normalize the final histogram using the L1 norm.

- **Dense Trajectories**. These are the low-level motion and appearance descriptors obtained from dense trajectories according to the algorithm described in Wang et al. (2013). The trajectories are computed for non-stationary points using a median-filtered optical flow method and are truncated every 15 frames. Each trajectory is then described in terms of its shape (point coordinate features, 30 dimensions), appearance (HOG features, 96 dimensions), optical flow (HOF features, 108 dimensions) and boundary motion (MBHx and MBHy features, 96 dimensions each). As in Wang et al. (2013), we learn a separate dictionary for each of these 5 descriptors. We use a codebook of $d = 5000$ cluster centroids for each descriptor. Thus, each space-time volume in a video is then represented as a vector of $D = 25{,}000$ dimensions obtained by concatenating the 5 histograms of trajectories occurring within that volume. We L1-normalize the final histogram.

4.2 Learning EXMOVES

The input for learning an EXMOVE consists of a positive video $\mathcal{V}^+$ containing a manually-annotated space-time 3D box bounding the action of interest $\mathbf{x}_E$, and thousands of negative videos $\mathcal{V}^-_{1\ldots N}$ *without* action volume annotations. The only requirement on the negative videos is that they must represent action classes different from the category of the positive exemplar (e.g., if the exemplar contains the action *dancing*, we exclude dancing videos from the negative set). But this constraint can be simply enforced given action class labels for the videos, without the need to know the space-time volumes of these negative actions. For example, tagged Internet videos (e.g., YouTube sequences) could be used as negative videos, by choosing action tags different from the activity of the positive exemplar.

It is worth noting that different movement exemplars will have different 3D box shapes. For example, we expect a walking action to require a tall volume while swimming may have a volume more horizontally elongated. As further discussed below, we maintain the original shape-ratio of the exemplar volume in both training and testing. This means that we look for only tall volumes when detecting walking, and short-and-wide volumes when searching for the swimming action.

Let $\mathbf{x}_E$ be the manually-specified volume in the positive sequence $\mathcal{V}^+$.

Let us denote with $\boldsymbol{\phi}(\mathbf{x})$ the L1-normalized histogram of codewords (computed from either HOG-HOF-STIPs or Dense Trajectories) within a video volume $\mathbf{x}$, i.e., $\boldsymbol{\phi}(\mathbf{x}) = \frac{1}{c(\mathbf{x})}\,[c_1(\mathbf{x}), \ldots, c_D(\mathbf{x})]^T$, where $c_i(\mathbf{x})$ is the number of codeword i occurring in volume $\mathbf{x}$, and $c(\mathbf{x})$ is the total number of codewords in $\mathbf{x}$, and D is the size of the dictionary. Note that in the case of Dense Trajectories, each trajectory contributes 5 codewords into the histogram since it is quantized according to the 5 separate dictionaries.

Adopting the exemplar-SVM method in Malisiewicz et al. (2011), our exemplar-SVM training procedure learns a linear classifier $f(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) + b$, by minimizing the following objective function:

$$\min_{\mathbf{w},b} \|\mathbf{w}\|^2 + C_1 \sum_{\mathbf{x}\in\mathcal{V}^+ \ \text{s}.t.\ \frac{|\mathbf{x}\cap\mathbf{x}_E|}{|\mathbf{x}_E|}\geq 0.5} h\big(\mathbf{w}^T \phi(\mathbf{x}) + b\big)$$

$$+C_2 \sum_{i=1}^{N} \sum_{\mathbf{x}\in\mathcal{V}_i^-} h\big(-\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) - b\big) \qquad (1)$$

where $h(s) = \max(0, 1 - s)$ is the hinge loss, while C_1 and C_2 are pre-defined parameters that we set so as to equalize the unbalanced proportion of positive and negative examples. Note that the first summation in the objective involves subvolumes whose spatial overlap with $\mathbf{x}_E$ is greater than 50 % and thus are expected to yield a positive score, while the second summation is over all negative subvolumes. Unfortunately, direct minimization of the objective in Eq. 1 is not feasible since it requires optimizing the SVM parameters on a gigantic number of subvolumes. Thus, we resort to an alternation scheme similar to that used in Malisiewicz et al. (2011) and Felzenszwalb et al. (2010): we iterate between 1) learning the parameters $(\mathbf{w}, b)$ given an active set S of negative volumes and 2) mining new negative volumes with the current SVM parameters.

We first initialize the parameters of the classifier by traditional SVM training using the manually-selected volume $\mathbf{x}_E$ as positive example and a randomly selected subvolumes from each of the other videos as negative example. At each iteration the current SVM is evaluated exhaustively on every negative video to find violating subvolumes, i.e., subvolumes yielding an SVM score below exceeding -1. These subvolumes are added as negative examples to the active set S to be used in the successive iterations of SVM learning. Furthermore, our training procedure adds as positive examples the subvolumes of $\mathcal{V}^+$ that have spatial overlap with $\mathbf{x}_E$ greater than 50 % and SVM score below 1. We stop the iterative alternation between these two steps when either no new subvolumes are added to the active set or a maximum number of iterations M is reached. In our implementation we use $M = 10$, but we find that in more than 85 % of the cases, the

Algorithm 1 EXMOVE training

```
Input: A set of negative videos $\{\mathcal{V}_1^-, \ldots, \mathcal{V}_N^-\}$ and a manually-selected
    volume $\mathbf{x}_E$ in exemplar video $\mathcal{V}^+$.
Output: Parameters $(\mathbf{w}, b)$ of exemplar-SVM.
1: $S \leftarrow \{(\mathbf{x}_E, +1)\}$
2: for $i = 1$ to $N$ do
3:     $S \leftarrow S \cup \{(\mathbf{x}_i, -1)\}$ with $\mathbf{x}_i$ randomly chosen from $\mathcal{V}_i^-$
4: for $iter = 1$ to $M$ do
5:     $(\mathbf{w}, b) \leftarrow$ svm_training($S$)
6:     $S_{old} \leftarrow S$
7:     for all $\mathbf{x}$ in $\mathcal{V}^+$ s.t. $\mathbf{w}^T\mathbf{x} + b < 1$ & $\frac{|\mathbf{x} \cap \mathbf{x}_E|}{|\mathbf{x}_E|} > 0.5$ do
8:         $S \leftarrow S \cup \{(\mathbf{x}, +1)\}$ //false negative
9:     for $i = 1$ to $N$ do
10:        for all $\mathbf{x}$ in $\mathcal{V}_i^-$ s.t. $\mathbf{w}^T\mathbf{x} + b > -1$ do
11:            $S \leftarrow S \cup \{(\mathbf{x}, -1)\}$ //false positive
12:    if $S_{old} = S$ then
13:        break
```

learning procedure converges before reaching this maximum number of iterations.

The pseudocode of our learning procedure is given in Algorithm 1. Lines 1–3 initialize the active set. The function `svm_training` in line 5 learns a traditional binary linear SVM using the labeled examples in the active set. Note that we found that at each iteration we typically have millions of subvolumes violating the constraints (lines 7–11). In order to maintain the learning of the SVM feasible, in practice we add to the active set only the volumes that yield the largest violations in each video, for a maximum of $k^- = 3$ per negative video and $k^+ = 10$ for the positive video.

4.3 Calibrating the Ensemble of EXMOVES

The learning procedure described above is applied to each positive exemplar independently to produce a collection of EXMOVES. However, because the exemplar classifiers are trained dis-jointly, their score ranges and distributions may vary considerably. A standard solution to this problem is to calibrate the outputs by learning for each classifier a function that converts the raw SVM score into a proper posterior probability compatible across different classes. To achieve this goal we use the procedure proposed by Platt in 1999: for each exemplar-SVM $(\mathbf{w}_E, b_E)$ we learn parameters (α_E, β_E) to produce calibrated probabilities through the sigmoid function $g(\mathbf{x}; \mathbf{w}_E, b_E, \alpha_E, \beta_E) = 1/[1+\exp(\alpha_E(\mathbf{w}_E^T\mathbf{x}+b_E)+\beta_E)]$. The fitting of parameters (α_E, β_E) is performed according to the iterative optimization described in Platt (1999) using as labeled examples the positive/negative volumes that are in the active set at the completion of the EXMOVE training procedure. As already noted in Malisiewicz et al. (2011), we also found that this calibration procedure yields a significant improvement in accuracy since it makes the range of scores more homogeneous and diminishes the effect of outlier values.

4.4 Efficient Computation of EXMOVE Scores

Although replacing the template matching procedure of Action Bank with linear SVMs applied to histograms of space-time features yields a good computational saving, this by itself is still not fast enough to be used in large-scale datasets due to the exhaustive sliding volume scheme. In fact, the sliding volume scheme is used in both training and testing. In training, we need to slide the current SVM over negative videos to find volumes violating the classification constraint. In testing, we need to slide the entire set of EXMOVE classifiers over the input video in order to extract the mid-level features for the subsequent recognition. Below, we describe a solution to speed up the sliding volume evaluation of the SVMs.

Let $\mathcal{V}$ be an input video of size $R \times C \times T$. Given an EXMOVE with parameters $(\mathbf{w}_E, b_E)$, we need to efficiently evaluate it over all subvolumes of $\mathcal{V}$ having size equal to the positive exemplar subvolume $\mathbf{x}_E$ (in practice, we slide the subvolume at N_s different scales but for simplicity we illustrate the procedure assuming we use the original scale). It is worth noting that the branch-and-bound method of Lampert et al. (2009) cannot be applied to our problem because it can only find the subwindow maximizing the classification score while we need the scores of all subvolumes; moreover it requires unnormalized histograms.

Instead, we use integral videos (Ke et al. 2010) to efficiently compute the EXMOVE score for each subvolume. An integral video is a volumetric data-structure having size equal to the input sequence (in this case $R \times C \times T$). It is useful to speed up the computation of functions defined over subvolumes and expressed as cumulative sums over voxels, i.e, functions of the form $H(\mathbf{x}) = \sum_{(r,c,t)\in\mathbf{x}} h(r, c, t)$, where (r, c, t) denotes a space-time point in volume $\mathbf{x}$ and h is a function over individual space-time voxels. The integral video for h at point (r, c, t) is simply an accumulation buffer B storing the sum of h over all voxels at locations less than or equal to (r, c, t), i.e., $B(r, c, t) = \sum_{r' \le r} \sum_{c' \le c} \sum_{t' \le t} h(r', c', t')$. This buffer can be built with complexity linear in the video size. Once built, it can be used to compute $H(\mathbf{x})$ for any subvolume $\mathbf{x}$ via a handful of additions and subtractions of the values in B.

In our case, the use of integral video is enabled by the fact that the classifier score can be expressed in terms of cumulative sums of individual point contributions, as we illustrate next. For simplicity we describe the procedure assuming that $\boldsymbol{\phi}(\mathbf{x})$ consists of a single histogram (as is the case for HOG-HOF-STIPs) but the method is straightforward to adapt for the scenario where $\boldsymbol{\phi}(\mathbf{x})$ is the concatenation of multiple histograms (e.g., the 5 histograms of Dense Trajectories). Let us indicate with $P(\mathbf{x})$ the set of quantized low-level features (either STIPs or Dense Trajectories) included in subvolume $\mathbf{x}$ of video $\mathcal{V}$ and let i_p be the codeword index of a point

$p \in P(\mathbf{x})$. Then we can rewrite the classification score of exemplar-SVM $(\mathbf{w}, b)$ on a subvolume $\mathbf{x}$ as follows (we omit the constant bias term b for brevity):

$$\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = \frac{1}{c(\mathbf{x})} \sum_{i=1}^{D} w_i c_i(\mathbf{x}) = \frac{\sum_{p \in P(\mathbf{x})} w_{i_p}}{\sum_{p \in P(\mathbf{x})} 1} . \qquad (2)$$

Equation 2 shows that the classifier score is expressed as a ratio where both the numerator and the denominator are computed as sums over individual voxels. Thus, the classifier score for any $\mathbf{x}$ can be efficiently calculated using two integral videos (one for the numerator, one for the denominator), without ever explicitly computing the histogram $\boldsymbol{\phi}(\mathbf{x})$ or the inner product between $\mathbf{w}$ and $\boldsymbol{\phi}(\mathbf{x})$. In the case where $\boldsymbol{\phi}(\mathbf{x})$ contains the concatenation of multiple histograms, then we would need an integral video for each of the histograms (thus 5 for Dense Trajectories), in addition to the common integral video for the denominator.

5 Implementation Details

Training data for EXMOVES. Since our approach shares many similarities with Action Bank, we adopt training and design settings similar to those used in Sadanand and Corso (2012) so as to facilitate the comparison between these two methods. Specifically, our EXMOVES are learned from the same set of UCF50 (Soomro et al. 2013) videos used to build the Action Bank templates. This set consists of 188 sequences spanning a total of 50 actions. Since the Action Bank volume annotations are not publicly available, we manually selected the action volume $\mathbf{x}_E$ on each of these exemplar sequences to obtain $N_a = 188$ exemplars. As negative set of videos we use the remaining 6492 sequences in the UCF50 dataset: for these videos no manual labeling of the action volume is available nor it is needed by our method. Action Bank also includes 6 templates taken from other sources but these videos have not been made publicly available; it also uses 10 templates taken from the KTH dataset. However, as the KTH videos are lower-resolution and contain much simpler actions compared to those in UCF50, we have not used them to build our EXMOVES. In the experiments we show that, while our descriptor is defined by a smaller number of movement classifiers (188 instead of 205), the recognition performance obtained with our mid-level features is consistently on par with or better than Action Bank.

Parameters of EXMOVE features. In order to compute the EXMOVE features from a new video, we perform max-pooling of the EXMOVE scores using a space-time pyramid based on the same settings as those of Action Bank, i.e., $N_s = 3$ scaled versions of the exemplar volume $\mathbf{x}_E$ (the scales are 1, 0.75, 0.5), and $N_p = 73$ space-time volumes obtained by recursive octree subdivision of the entire video using 3 levels (this yields 1 volume at level 1, 8 subvolumes at level 2, 64 subvolumes at level 3). Thus, the final dimensionality of our EXMOVE descriptor is $N_a \times N_s \times N_p = 41{,}172$.

6 Experiments

6.1 Action Recognition

Action classification model. All our action recognition experiments are performed by training a one-vs-the-rest linear SVM on the EXMOVES extracted from a set of training videos. We opted for this classifier as it is very efficient to train and test, and thus it is a suitable choice for the scenario of large-scale action recognition that we are interested in addressing. The hyperparameter C of the SVM is tuned via cross-validation for all baselines, Action Bank, and our EXMOVES.

Test datasets. We test our approach on the following large-scale action recognition datasets:

1. HMDB51 (Kuehne et al. 2011): It consists of 6849 image sequences collected from movies as well as YouTube and Google videos. They represent 51 action categories. The results for this dataset are presented using 3-fold cross validation on the 3 publicly available training/testing splits.
2. Hollywood-2 (Marszalek et al. 2009): This dataset includes over 20 hours of video, subdivided in 3669 sequences, spanning 12 action classes. We use the publicly available split of training and testing examples.
3. UCF50: This dataset contains 6676 videos taken from YouTube for a total of 50 action categories. This dataset was used in Sadanand and Corso (2012) and Jain et al. (2013) to train and evaluate Action Bank and Discriminative Patches.
4. UCF101 (Soomro et al. 2013) (part 2): UCF101 is a superset of UCF50. For this test we only use videos from action classes 51 to 101 (from now on denoted as part 2), thus omitting the above-mentioned classes and videos of UCF50. This leaves a total of 6851 videos and 51 action classes. We report the accuracy of 25-fold cross validation using the publicly available training/testing splits.

Comparison of recognition accuracies. We now present the classification performance obtained with our features on the four benchmarks described above. We consider in our comparison three other mid-level video descriptors that can be used for action recognition with linear SVMs: Action Bank (Sadanand and Corso 2012), Discriminative Patches (Jain et al. 2013) as well as histograms of visual words (BOW) built for the two types of low-level features that we use in EXMOVES, i.e., HOG-HOF-STIPs and Dense

Table 1 Comparison of recognition accuracies on four datasets

Low-level features	Mid-level descriptor	Descriptor dimensionality	Datasets			
			HMDB51	Hollywood-2	UCF50	UCF101 (part 2)
3D Gaussians	Action Bank	44, 895	26.9	n/a	57.9	n/a
HOG3D	Discriminative Patches	9, 360	n/a	n/a	61.2	n/a
HOG-HOF-STIPs	BOW	5, 000	20.0	32.6	52.8	49.1
	EXMOVES	41, 172	27.7	44.7	63.4	57.2
Dense Trajectories	BOW	25, 000	34.4	43.7	81.8	60.9
	EXMOVES	41, 172	**41.9**	**56.6**	**82.8**	**71.6**

Bold values indicate the best performing methods
The classification model is an efficient linear SVM applied to 4 distinct global mid-level descriptors: Action Bank (Sadanand and Corso 2012), Discriminative Patches (Jain et al. 2013), histogram of space-time visual words (BOW) and our EXMOVES. We consider two different low-level features to build BOW and EXMOVES: HOG-HOF-STIPs and Dense Trajectories. Our EXMOVES achieve the best recognition accuracy on all four datasets using Dense Trajectories, and greatly outperform the BOW descriptor for both our choices of low-level features, HOG-HOF-STIPs and Dense Trajectories

Trajectories. As in Wang et al. (2013), we use a dictionary of 25,000 visual words for Dense Trajectories and 5000 visual words for HOG-HOF-STIPs. Due to the high computational complexity of the extraction of Action Bank features, however, we were unable to test this descriptor on the large-scale datasets of Hollywood-2 and UCF101. For Discriminative Patches, we can only report accuracy on UCF50 since this is the only large-scale dataset on which they were tested in Jain et al. (2013) and no software to compute these features is available.

The accuracies achieved by the different descriptors are summarized in Table 1. From these results we see that our EXMOVE descriptor built from Dense Trajectories yields consistently the best results across all four datasets. Furthermore, EXMOVES give always higher accuracy than BOW built from the same low-level features, for both HOG-HOF-STIPs and Dense Trajectories. The gap is particularly large on challenging datasets such as Hollywood-2 and HMDB51. This underscores the advantageous effect of the movement exemplars to which we compare the input video in order to produce the EXMOVE features.

Table 2 lists the individual action recognition accuracies for the same subset of 13 UCF50 classes analyzed in Jain et al. (2013). We see that EXMOVES give the highest accuracy on 10 out of these 13 action categories.

In Table 3 we present the recognition accuracy for the individual classes of HMDB51 using a linear SVM trained on our EXMOVES with Dense Trajectories. The best recognition performance is achieved for “golfing” (accuracy is 96.7 %), while the worst prediction is for the class “waving” (accuracy is 5.6 %). Note that random chance would yield an accuracy of 1.96 %.

Computational cost of mid-level feature extraction. We want to emphasize that although our EXMOVES are based on a subset of the exemplars used to build Action Bank, they always generate equal or higher accuracy. Furthermore, our approach does so with a speedup of almost two-orders of magnitude in feature extraction: Table 4 reports the statistics of the runtime needed to extract EXMOVES and Action Bank. We used the software provided by the authors of Sadanand and Corso (2012) to extract Action Bank features from input videos. Due to large cost of Action Bank extraction, we collected our runtime statistics on the smaller-scale UT-I (Ryoo and Aggarwal 2010) dataset, involving only 120 videos. Runtimes were measured on a single-core Linux machine with a CPU @ 2.66 GHz. The table reports the complete time from the input of the video to the output of the descriptor, inclusive of the time needed to compute low-level features. The extraction of EXMOVES is on average over 70 times faster than for Action Bank when using HOG-HOF-STIPs and 11 times faster when using Dense Trajectories. We can process the entire UT-Interaction dataset with HOG-HOF-STIPs using a single CPU in 14 h; extracting the Action Bank features on the same dataset would take 41 days.

We were unable to collect runtime statistics for Discriminative Patches due to the unavailability of the software. However, we want to point out that this descriptor uses many more patches than EXMOVES (1040 instead of 188) and it cannot use the Integral Video speed-up.

Computational cost of action recognition. Finally, we would like to point out that as shown in Table 1, the accuracies achieved by an *efficient linear* SVM trained on EXMOVES are very close to the best published results of Wang et al. (2013), which instead were obtained with a much more computationally expensive model, not suitable for scalable action recognition: they report a top-performance of 46.6 and 58.2 % on HMDB51 and Hollywood-2, respectively, using an expensive non-linear SVM with an RBF-χ^2 kernel applied to BOW of Dense Trajectories. In our experiments we found that training a linear SVM on EXMOVES for one of the HMDB51

Table 2 Recognition accuracies of our EXMOVES (applied to Dense Trajectories) compared with those of Action Bank and Discriminative Patches using the same subset of 13 action classes from UCF50 considered in Jain et al. (2013)

Action Class	Action Bank	Discriminative Patches	EXMOVES
Basketball	53.84	50.00	**56.93**
Clean and jerk	85.00	**95.65**	91.07
Diving	78.79	61.29	**96.08**
Golf swing	**90.32**	75.86	90.14
High jump	38.46	55.56	**81.30**
Javeline throw	45.83	50.00	**73.50**
Mixing	42.85	55.56	**97.16**
Pole vault	60.60	84.37	**94.38**
Pull up	91.67	75.00	**96.00**
Push ups	85.00	86.36	**91.18**
Tennis swing	44.12	48.48	**85.03**
Throw discus	75.00	87.10	**93.13**
Volleyball spiking	43.48	**90.90**	89.66
Mean classification	64.23	70.47	**87.35**

Bold values indicate the best performing methods

Table 3 Recognition accuracy on the individual classes of HMDB51 using linear SVMs trained on EXMOVES based on Dense Trajectories

Golf	96.7	Laugh	48.9	Smoke	31.1
Pullup	87.8	Ride bike	47.8	Stand	31.1
Pushup	76.7	Turn	47.8	Kick	27.8
Brush hair	75.6	Shoot bow	46.7	Kick ball	26.7
Situp	71.1	Sit	46.7	Walk	26.7
Kiss	68.9	Drink	45.6	Sword	25.6
Catch	65.6	Hit	44.4	Cartwheel	20.0
Shake hands	65.6	Push	43.3	Run	20.0
Hug	62.2	Fall floor	42.2	Sword exercise	18.9
Dribble	61.1	Somersault	41.1	Dive	17.8
Pour	61.1	Shoot ball	38.9	Eat	17.8
Climb	58.9	Talk	37.8	Shoot gun	16.7
Ride horse	56.7	Jump	36.7	Pick	14.4
Flic flac	53.3	Climb stairs	35.6	Punch	11.1
Chew	48.9	Draw sword	35.6	Throw	7.8
Clap	48.9	Smile	33.3	Swing baseball	5.6
Fencing	48.9	Handstand	32.2	Wave	5.6

Note that random chance would yield a recognition rate of 1.96 %

classes takes only 6.2 s but learning a kernel-SVM on BOW of Dense Trajectories requires 25 minutes (thus overhead is 250X); the testing of our linear SVM on a video takes only 7 ms, while the nonlinear SVM is on average more than two orders of magnitude slower. Its cost depends on the on the number of support vectors, which varies from a few hundreds to several thousands. Nonlinear SVMs also need more memory to store the support vectors.

Varying the number of exemplars. In this experiment we study how the accuracy of our method changes as a function of the number of EXMOVES used in the descriptor. Starting from our complete feature vector defined by $N_a = 188$ exemplars and having dimensionality $N_a \times N_s \times N_p = 41{,}172$, we recursively apply a feature selection procedure that eliminates at each iteration one of the EXMOVE exemplars and removes its associated $N_s \times N_p$ features from the descriptor. We apply a variant of multi-class Recursive Feature Elimination (Chapelle and Keerthi 2008) to determine the EXMOVE to eliminate at each iteration. This procedure operates as follows: given a labeled training set of video examples for K classes, at each iteration we retrain the one-vs-the-rest linear SVMs for all K classes using the current version of our feature vector and then we remove from the descriptor the EXMOVE that is overall "least used" by the K linear classifiers by looking at the average magnitude of the SVM parameter vector $\mathbf{w}$ for the different EXMOVE sub-blocks.

We perform this analysis on the HDMB51 dataset using both HOG-HOF-STIPs and Dense Trajectories as low-level features for EXMOVES. Figure 2 reports the 3-fold cross-validation error as a function of the number of EXMOVES used in our descriptor. Interestingly, we see that the accuracy remains close to the top-performance even when we reduce the number of exemplars to only 100. This suggests a certain redundancy in the set of movement exemplars. The accuracy begins to drop much more rapidly when fewer than 50 exemplars are used.

The effects of multiple scales and spatio-temporal pyramid levels. We study the effects of varying the number of scales and the number of spatio-temporal pyramid levels on EXMOVES. The version of EXMOVES having highest dimensionality involves three different scales (1, 0.75, 5) and three different spatio-temporal pyramid levels ($1 \times 1 \times 1$, $2 \times 2 \times 2$, and $3 \times 3 \times 3$), as already discussed in the previous section. Here we vary the number of scales from only

Table 4 Statistics of time needed to extract the mid-level descriptors Action Bank and EXMOVES

Descriptor	Extraction time per video (minutes)			# frames per second
	Mean	Max	Min	mean
Action Bank	495	1199	132	0.012
EXMOVES w/ HOG-HOF-STIPs	7	16	3	0.82
EXMOVES w/ Dense Trajectory	43	70	29	0.13

The time needed to extract EXMOVES features for the entire UT-I dataset using a single CPU is only 14 h; instead, it would take more than 41 days to compute Action Bank descriptors for this dataset

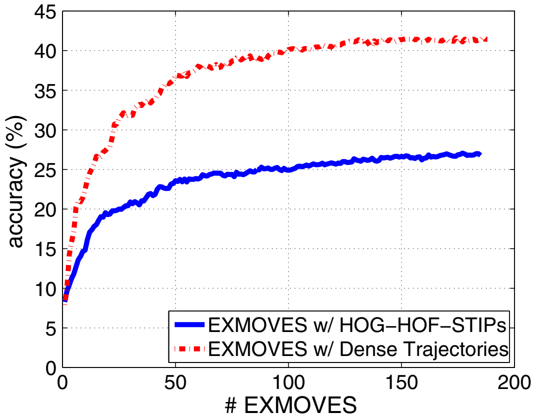

Fig. 2 Accuracy on HMDB51 as a function of the number of EXMOVES. We use Recursive Feature Elimination to reduce the number of EXMOVES. The accuracy remains near the state-of-the-art even when using only 100 exemplars

1 scale (1), 2 scales (1, 0.75), or all 3 scales. We also vary the number of pyramid levels: only 1 level (level 1), 2 levels (1 and 2), or all 3 levels. At the lowest dimensionality, we only use 1 scale and 1 level of pyramid, which gives rise to a 188-dimensional feature vector. At the highest dimensionality, with 3 scales and 3 pyramid levels, EXMOVES become 41,172-dimensional feature vectors. Table 5 presents the action recognition accuracy of EXMOVES varying the number of scales and pyramid levels on UCF50 and UCF101-part 2. The empirical results show that EXMOVES do not benefit much from multiple scales, but their performance is significantly boosted by the use of multiple spatio-temporal pyramid levels. Reducing from 3 scales to 2 scales causes only a 1–2 % in accuracy, while going from 2 scales to 1 scale causes a degradation in accuracy of 2.5–3.5 % on both datasets. Instead, moving from 3 to 2 pyramid levels the accuracy drops by 4–5 %, and reducing the number of pyramid levels from 2 to 1 degrades the accuracy by 10–12 %. Interestingly, EXMOVES built with 1-scale and 1-pyramid-level are 188-dimensional and they achieve an accuracy of 68.3 % on UCF50. This is considerably higher than the accuracies of Action Bank (Sadanand and Corso 2012) and Discriminative Patches (Jain et al. 2013) (57.9 and 61.2 %, respectively), which have much higher dimensionality.

The effects of bounding box annotations. We study the effects of annotations on our EXMOVES. In this experiment, we train our EXMOVES without using any bounding box annotations. We call these features WS-EXMOVES (weakly-supervised EXMOVES). We note that, as before, we are still using one positive example and many negative examples to train our mid-level descriptor, except for not using bounding box annotations. To train each WS-EXMOVE, we randomly generate $k^+ = 10$ subvolumes from the positive video and $k^- = 3$ subvolumes from each negative video. These subvolumes are used as positive and negative training examples to train a linear SVM. Each linear SVM is then calibrated by the same algorithm as before (Platt 1999). Table 6 compares the accuracies of WS-EXMOVES and EXMOVES on four different datasets. On Hollywood-2, the difference is small – only 0.6 %, due to the small dataset and the reduced number of classes to discriminate (12-categories). On UCF50 and UCF101-part 2, the difference is about 2.3–4.8 %, while on the more challenging HMDB51 dataset the gap is 6.7 %. As the accuracy drop for not having bounding box annotations is small, one can even afford to increase the number of exemplars to improve the discriminative power of the descriptor at very little annotation cost.

6.2 Qualitative Results on Action Retrieval

We also qualitatively evaluate our EXMOVES on the task of action retrieval. In this experiment, given a query video, we perform simple Top-K retrieval using the Euclidean distance. Figure 3 shows the 15-nearest neighbors for 4 different queries using EXMOVES on UCF50. We intentionally chose queries from classes that can be recognized reliably (“Pommel Horse” and “Punch”), but also from the hardest category to recognize (“Basketball” which is often confused with “Volleyball Spiking” and “’Pizza Tossing”). “Clean and Jerk” ranks roughly in the middle among all categories in terms of recognition accuracy with EXMOVES. As shown in Fig. 3, for the two queries in the categories “Pommel Horse” and “Punch”, all 15-nearest belong to the same class as the query. The top-15 results retrieved for the query “Clean and Jerk” include one incorrect example (belonging to the class “Bench Press”). The retrieval results for the query belonging to the

Table 5 Effects of multiple scales and spatio-temporal pyramid levels on EXMOVES. The recognition accuracy on UCF50 and UCF101-part2 are reported for different numbers of scales and levels of spatio-temporal pyramid. The EXMOVE dimensionalities for the different settings are shown in brackets

Datset	# of scales	1	2	3
UCF50	1 pyr. level	68.3 (188)	68.4 (376)	68.4 (564)
	2 pyr. levels	74.8 (1,692)	77.1 (3,384)	78.0 (5,076)
	3 pyr. levels	77.3 (13,724)	80.1 (27,448)	**82.8** (41,172)
UCF101-part2	1 pyr. level	51.9 (188)	54.3 (376)	53.7 (564)
	2 pyr. levels	62.0 (1,692)	64.7 (3,384)	66.2 (5,076)
	3 pyr. levels	65.4 (13,724)	69.1 (27,448)	**71.6** (41,172)

Bold values indicate the best performing methods

Table 6 **The effects of bounding box annotations on EXMOVES**. WS-EXMOVES are trained without manual bounding-box annotations, while EXMOVES are learned using a single manually-annotated bounding box in each video. The action recognition accuracy of WS-EXMOVES is 1–6 % lower than that of EXMOVES on the HMDB51, Hollywood-2, UCF50, and UCF101 datasets

Datset	HMDB51	Hollywood-2	UCF50	UCF101-part2
WS-EXMOVES	35.2	56.0	78.0	69.3
EXMOVES	**41.9**	**56.6**	**82.8**	**71.6**

Bold values indicate the best performing methods

most challenging class (“Basketball”) include many more mistakes (10 in the top-15).

We also qualitatively evaluate our EXMOVES on the task of action retrieval across different datasets. Figure 4 shows the top-5 retrieval results when the query videos are from HMDB51 and the retrieval database is UCF50. We randomly selected queries from the two classes having the highest recognition accuracy (the first and the second query), the two classes having the worst recognition accuracy (the fifth and the sixth query), and from two classes with intermediate accuracy (the third and the fourth query). For the “ride horse” query, the top-5 results are all correct. The “pull up” retrieval results include an error (a “Swing” video, which exhibits appearance and motions similar to those of “pull up”). Note how for the query “riding bike”, the top-5 results include one incorrect video of “riding horse”. Finally, the “push up” query yields three incorrect examples of “bench press”, which are similar in pose and motions to “push up”.

6.3 Action Similarity Labeling

We now show that our EXMOVES can be applied to tasks beyond action recognition by presenting results on the problem of action similarity labeling (Kliper-Gross et al. 2012).

Dataset. We use the ASLAN challenge dataset (Kliper-Gross et al. 2012) for action similarity labeling. The dataset consists of 3697 video clips of 432 action categories. Given a pair of video clips as input, the objective is to determine whether they contain the “same” action or “different” actions. Thus, this can be viewed as a binary classification problem. In Kliper-Gross et al. (2012), the authors define 10 splits of the dataset. Each split contains 300 pairs of videos with same actions and 300 video pairs with different actions. The dataset is difficult because the number of action categories is large and the action classes are fine-grained. For example, there are 29 variants of jumping, and 10 distinct categories of “sitting-up”.

Binary classification model for action similarity labeling. In Kliper-Gross et al. (2012) the authors report the performance of several features (HOG, HOF, HNF (Laptev 2005), and their combination) with 12 different distance metrics used as kernels for binary classification of video pairs. In order to maintain our approach scalable and efficient, we train a binary linear SVM on the absolute difference of the two EXMOVE descriptors extracted from the input pair. In other words, the SVM is trained on the absolute difference vector to predict whether the two videos contain the same action or not. Note that the other distances used in Kliper-Gross et al. (2012), such as the χ^2 or other non-linear kernels, are much more costly to compute. We report the labeling accuracy as well as the area under ROC curve using tenfold cross validation as used in Kliper-Gross et al. (2012).

Comparison of features for action similarity labeling. Table 7 presents the accuracy of our EXMOVES on the similarity labeling challenge. We include comparative results obtained with current state-of-the-art features using the same binary classification model, i.e., a binary SVM trained on the absolute difference vector. Our EXMOVES outperform all single feature descriptors (HOG, HOF, HNF) by 2–3 %

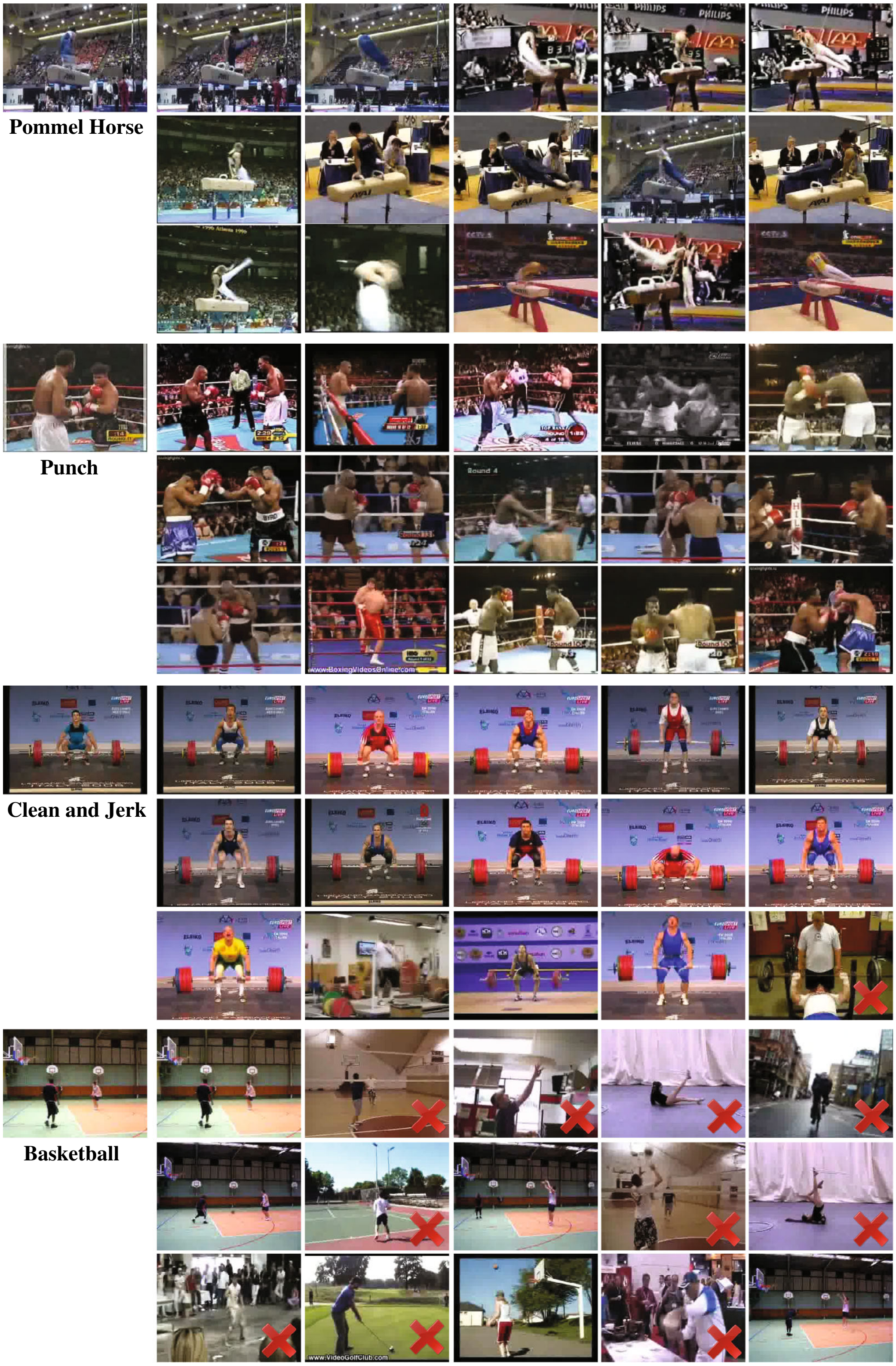

Fig. 3 Action retrieval on UCF50. The left column shows the query videos. Next to each query we show the 15-nearest neighbors retrieved using EXMOVES. The incorrectly-retrieved examples (i.e., videos belonging to a class different from that of the query) are marked with a *red cross*. The first two query are from classes that are easy to recognize, the third query is randomly chosen from a class that ranks roughly in the middle in terms of recognition accuracy. The last query is randomly chosen from the most difficult class to recognize (according to the confusion matrix). Best viewed in color (Color figure online)

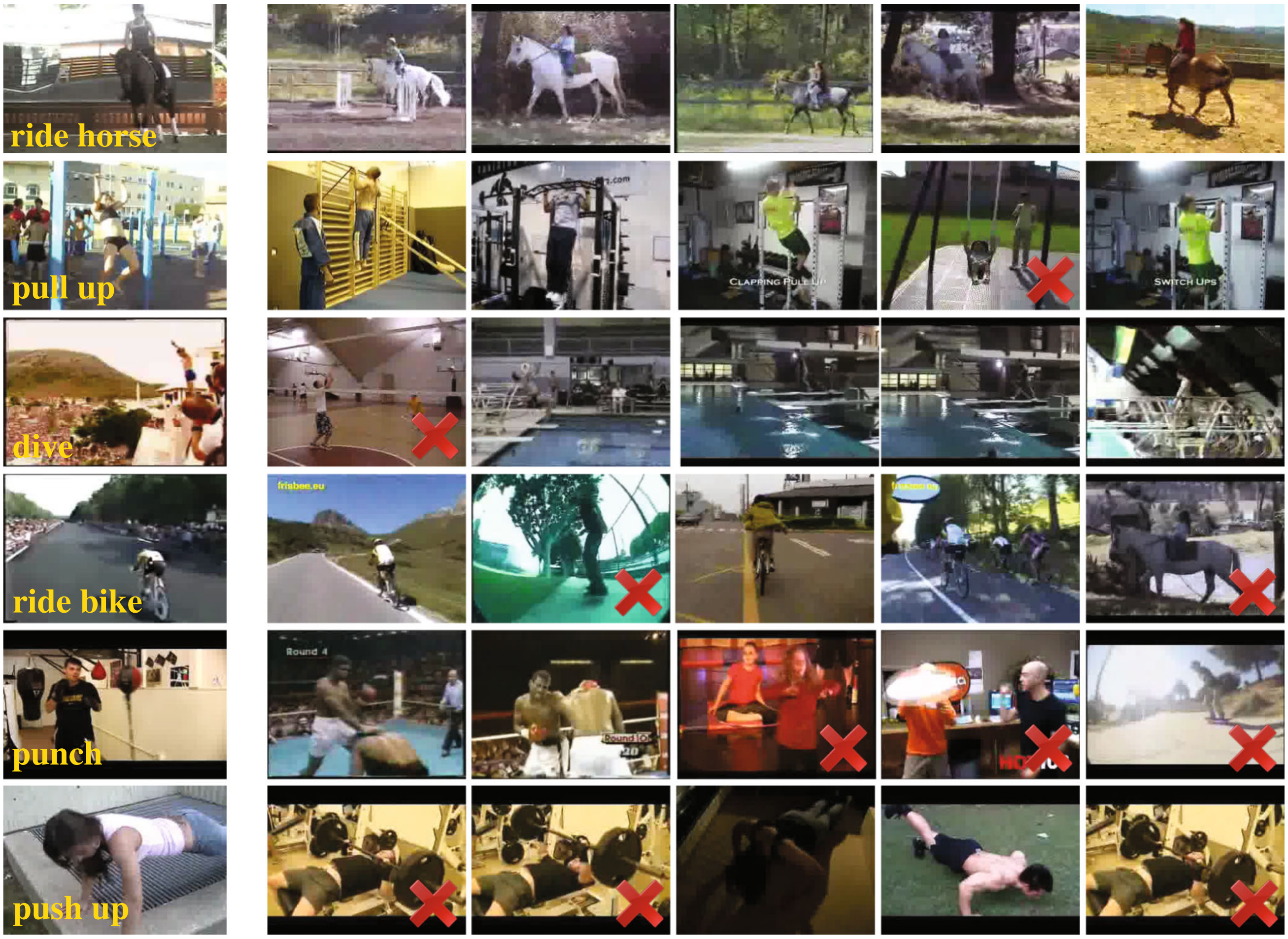

Fig. 4 Cross-dataset action retrieval. The first column shows the query videos from HMDB51. To the right of each query video we show the 5-nearest neighbors retrieved from UCF50. The retrieved examples that have incorrect label are marked with a *red cross*. Best viewed in color (Color figure online)

Table 7 Action similarity labeling results. Comparisons between EXMOVES and current state-of-the art features (Kliper-Gross et al. 2012) using a binary linear SVM trained on the absolute difference vector, i.e., $|x_1 - x_2|$ where x_1, x_2 here denote the feature vectors extracted from the two input videos. The numbers are accuracies and area under ROC curve (in parenthesis). EXMOVES outperform other single feature vectors by 3–4 %, and combined descriptors by 1 %

Feature	HOG	HOF	HNF	HOG-HOF-HNF	EXMOVES
Acc (AUC)	52.23 (54.41)	53.53 (55.59)	53.75 (55.90)	54.80 (57.01)	**55.32** (**58.06**)

Bold value indicates the best performing methods

on accuracy and 3–4 % on AUC. Our EXMOVES are even better than the *combination* of these three feature vectors, providing an improvement of 0.5 and 1 % on accuracy and AUC, respectively.

Figure 5 shows qualitative results of action similarity labeling using EXMOVES for 4 test pairs of videos. Each row shows an input test pair of video clips (we present three frame of each video clip). The ground-truth action labels are marked in blue in the right bottom corner of each image sequence. The first two test pairs are true positives, i.e., the linear SVM using EXMOVES correctly labels these pairs as “same”. It is worth noting that the second test example is quite difficult as the same actions appearing in different scales, view, and lighting condition. The third pair causes a false negative prediction: the SVM using EXMOVES fails to label this pair as “same,” probably due to the largely different viewpoints of the two video clips. The last row shows a false positive case. Our system fails to label the two videos as “different” because of the similar patterns of motions and poses.

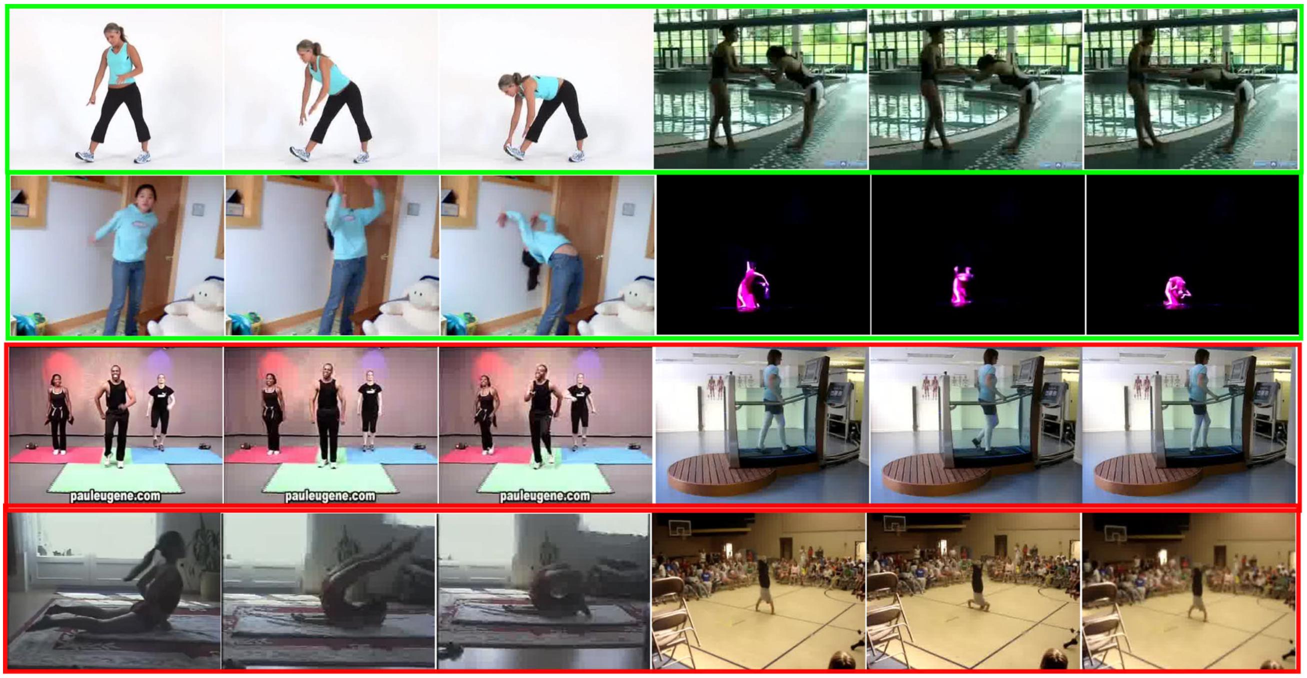

Fig. 5 Action similarity labeling. Visualizations of action similarity pairs. Each row represent a test input pair. The binary classifier using EXMOVES correctly classifies the pairs in the first two rows (true positives). The third row is a false negative, and the last row is a false positive. Note that although the two video clips in the second row have largely different scales and viewpoints, our method is able to correctly label them as containing the same action. Our method fails to label the third pair as "different" because of the different viewpoints, and the fourth pair as "same" because the two videos exhibit similar motions. Best view in color

7 Conclusions

We have presented an approach for efficient large-scale analysis of human actions. It centers around the learning of a mid-level video representation that enables good accuracy with efficient linear classification models. The benefits of our features are threefold. First, building our representation requires very little human intervention, as only one positive manual annotation is required for each feature entry. Second, our approach is easy to scale to large datasets thanks to low computational cost of EXMOVE extraction and the good accuracy obtainable with *linear* classifiers, which are fast to train and test. Last but not least, our approach is quite general, as it provides good accuracy with different types of low-level features and different problems of human action analysis. Experiments on large-scale benchmarks of action recognition and action similarity labeling show the accuracy and efficiency of our approach. To our best knowledge, this work is the first one experimented on all known large-scale benchmarks for human action analysis.

Our mid-level features are produced by evaluating a set of movement classifiers over the input video. An important question we plan to address in future work is: how many mid-level classifiers do we need to train before accuracy levels off? Also, what kind of movement classes are particularly useful as mid-level features? Currently, we are restricted in the ability to answer these questions by the scarceness of labeled data available, in terms of both number of video examples but also number of action classes. An exciting avenue to resolve these issues is the design of methods that can learn robust mid-level classifiers from weakly-labelled data, such as YouTube videos.

Additional material including software to extract EXMOVES from videos is available at http://vlg.cs.dartmouth.edu/exmoves.

Acknowledgments Thanks to Alessandro Bergamo for assistance with the experiments. We are grateful to Tomasz Malisiewicz for clarifications about Exemplar SVM and to Jason Corso for providing the Action Bank exemplars. This research was funded in part by NSF CAREER award IIS-0952943 and NSF award CNS-1205521.

References

Blank, M., Gorelick, L., Shechtman, E., Irani, M. & Basri, R. (2005). Actions as space-time shapes. In *International Conference on Computer Vision*, (pp. 1395–1402)

Chapelle, O. & Keerthi, S. (2008). Multi-class feature selection with support vector machines. In *Proceedings of the American Statistical Association*

Dalal, N. & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*

Dalal, N., Triggs, B. & Schmid, C. (2006). Human detection using oriented histograms of flow and appearance. In *European Conference on Computer Vision*

Deng, J., Berg, A., & Fei-Fei, L. (2011). Hierarchical semantic indexing for large scale image retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition*

Derpanis, K., Sizintsev, M., Cannons, K. & Wildes, P. (2010). Efficient action spotting based on a spacetime oriented structure representation. In *IEEE Conference on Computer Vision and Pattern Recognition*

Dollar, P., Rabaud, V., Cottrell, G., & Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, (pp. 65–72)

Efros, A., Berg, A., Mori, G., & Malik, J. (2003). Recognizing action at a distance. In *International Conference on Computer Vision*, (pp. 726–733)

Farhadi, A., Endres, I., Hoiem, D., Forsyth, D.A. (2009). Describing objects by their attributes. In *CVPR*, (pp. 1778–1785)

Fathi, A., & Mori, G. (2008). Action recognition by learning mid-level motion features. In *CVPR*

Felzenszwalb, P., Girshick, R., McAllester, D., & Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(9), 1627–1645.

Ferrari, V., & Zisserman, A. (2007). Learning visual attributes. In *Proceedings Neural Information Processing Systems*, (pp. 433–440)

Hu, Y., Cao, L., Lv, F., Yan, S., Gong, Y., & Huang, T. (2009). Action detection in complex scenes with spatial and temporal ambiguities. In *International Conference on Computer Vision*

Jain, A., Gupta, A., Rodriguez, M., & Davis, L. (2013). Representing videos using mid-level discriminative patches. In *IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 2571–2578)

Ke, Y., Sukthankar, R., & Hebert, M. (2005). Efficient visual event detection using volumetric features. In *International Conference on Computer Vision*

Ke, Y., Sukthankar, R., & Hebert, M. (2010). Volumetric features for video event detection. *International Journal of Computer Vision*

Klaser, A., Marszalek, M., & Schmid, C. (2008) A spatio-temporal descriptor based on 3D-gradients. In *British Machine Vision Conference*

Kliper-Gross, O., Hassner, T., & Wolf, L. (2012). The action similarity labeling challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(3), 615–621.

Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., & Serre, T. (2011). Hmdb: A large video database for human motion recognition. In *International Conference on Computer Vision*

Lampert, C., Blaschko, M., & Hofmann, T. (2009). Efficient subwindow search: A branch and bound framework for object localization. IEEE *Transactions on Pattern Analysis and Machine Intelligence*

Lampert, C.H., Nickisch, H., & Harmeling, S. (2009). Learning to detect unseen object classes by between-class attribute transfer. In *Proceedings on IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 951–958)

Laptev, I. (2005). On space-time interest points. *International Journal of Computer Vision*, *64*(2–3), 107–123.

Laptev, I., & Lindeberg, T. (2003). Space-time interest points. In *International Conference on Computer Vision*

Laptev, I., Marszalek, M., C.S., & Rozenfeld, B. (2008). Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition*

Laptev, I., & Prez, P. (2007). Retrieving actions in movies. In *International Conference on Computer Vision*

Le, Q., Zou, W., Yeung, S., & Ng, A. (2011). Learning hierarchical spatio-temporal features for action recognition with independent subspace analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*

Li, L., Su, H., Xing, E., & Fei-Fei, L. (2010). Object Bank: A high-level image representation for scene classification & semantic feature sparsification. In *Neural Information Processing Systems*

Liu, J., Kuipers, B., & Savarese, S. (2011). Recognizing human actions by attributes. In *CVPR*, (pp. 3337–3344)

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*

Malisiewicz, T., Gupta, A., & Efros, A. (2011). Ensemble of exemplar-SVMs for object detection and beyond. In *International Conference on Computer Vision*

Marszalek, M., Laptev, I., & Schmid, C. (2009). Action in context. In *IEEE Conference on Computer Vision and Pattern Recognition*

Niebles, J., & Fei-Fei, L. (2007). A hierarchical model of shape and appearance for human action classification. In *IEEE Conference on Computer Vision and Pattern Recognition*

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, MIT Press.

Ryoo, M., & Aggarwal, J. (2010). UT-interaction dataset, ICPR contest on semantic description of human activities

Sadanand, S., & Corso, J. (2012). Action bank: A high-level representation of activity in video. In *IEEE Conference on Computer Vision and Pattern Recognition*

Scovanner, P., Ali, S., & Shah, M. (2007) A 3-Dimensional SIFT descriptor and its application to action recognition. In *ACM Conference on Multimedia*, (pp. 357–360)

Soomro, K., Roshan Zamir, A., & Shah, M. (2013). UCF101: A dataset of 101 human action classes from videos in the wild. Tech. Rep. CRCV-TR-12-01, University of Central Florida

Torresani, L., Szummer, M., & Fitzgibbon, A. (2010). Efficient object category recognition using classemes. In *European Conference on Computer Vision*

Tran, D., & Sorokin, A. (2008). Human activity recognition with metric learning. In *European Conference on Computer Vision*

Veeraraghavan, A., Chellappa, R., & Roy-Chowdhury, A. (2006). The function space of an activity. In *IEEE Conference on Computer Vision and Pattern Recognition*

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*

Wang, G., Hoiem, D., & Forsyth, D. (2009). Learning image similarity from flickr using stochastic intersection kernel machines. In *International Conference on Computer Vision*

Wang, H., Kläser, A., Schmid, C., & Liu, C. (2013). Dense trajectories and motion boundary descriptors for action recognition. *International Journal of Computer Vision*

Wang, L., & Suter, D. (2007). Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. In *IEEE Conference on Computer Vision and Pattern Recognition*

Wang, Y., Tran, D., & Liao, Z. (2011). Learning hierarchical poselets for human parsing. In *IEEE Conference on Computer Vision and Pattern Recognition*

Weinberger, K., Blitzer, J., & Saul, L. (2006). Distance metric learning for large margin nearest neighbor classification. In *Neural Information Processing Systems*

Yu, G., Yuan, J., & Liu, Z. (2011). Unsupervised random forest indexing for fast action search. In *CVPR*, (pp. 865–872)

Yu, G., Yuan, J., & Liu, Z. (2012). Propagative hough voting for human activity recognition. In *European Conference on Computer Vision*

Yuan, J., Liu, Z., & Wu, Y. (2009). Discriminative subvolume search for efficient action detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, (pp. 2442–2449)

Yuan, J., Liu, Z., & Wu, Y. (2011). Discriminative video pattern search for efficient action detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*