1-1-2013

MEXSVMs: Mid-level Features for Scalable Action Recognition

Du Tran
Dartmouth College

Lorenzo Torresani
Dartmouth College

MEXSVMs: Mid-level Features for Scalable Action Recognition

Du Tran and Lorenzo Torresani
Department of Computer Science, Dartmouth College
6211 Sudikoff Lab, Hanover, NH 03755
{dutran,lorenzo}@cs.dartmouth.edu

Dartmouth Computer Science Technical Report TR2013-726

Abstract

This paper introduces MEXSVMs*, a mid-level representation enabling efficient recognition of actions in videos. The entries in our descriptor are the outputs of several movement classifiers evaluated over spatial-temporal volumes of the image sequence, using space-time interest points as low-level features. Each movement classifier is a simple exemplar-SVM, i.e., an SVM trained using a single positive video and a large number of negative sequences.*

Our representation offers two main advantages. First, since our mid-level features are learned from individual video exemplars, they require minimal amount of supervision. Second, we show that even simple linear *classification models trained on our global video descriptor yield action recognition accuracy comparable to the state-of-the-art. Because of the simplicity of linear models, our descriptor can efficiently learn classifiers for a large number of different actions and to recognize actions even in large video databases. Experiments on two of the most challenging action recognition benchmarks demonstrate that our approach achieves accuracy similar to the best known methods while performing* 70 *times faster than the closest competitor.*

1. Introduction

Human action recognition is an important but still largely unsolved problem in computer vision with many potential 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, different motions, viewing positions and angles, 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 [2, 31, 35, 36]. 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; and 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 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 [24] trained on quantized space-time interest points [19] and optimized to separate a single positive video exemplar from an army of "background" negative videos. Because only one labeled video is needed to train the 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 or 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 movement exemplar-SVMs. 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 is similar in spirit to attribute-based recognition in still images [7, 16, 18], where object and scene classes are described in terms of sets of semantic characteristics automatically detected in the photos. In particular, our approach can be viewed as extending to videos the idea of classifier-based image descriptors [5, 23, 34, 38] 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 [30], who have been the first to propose 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. These descriptors achieve state-of-the-art accuracy on several benchmarks. However, the template-matching step to extract these mid-level features is very computationally expensive. As reported in [30], extracting mid-level features from a single video of UCF50 [1] takes a minimum of 0.4 hours up to a maximum of 34 hours. 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 main contribution is to replace this prohibitively expensive procedure with a technique that is almost two orders of magnitude faster and yields a mid-level representation that is comparable in terms of action recognition accuracy. This enables the application of our descriptor for recognition in large video databases, where the Action Bank framework is simply too costly to be used.

2. Related Work

Many approaches to human action recognition have been proposed over the last ten years. Most of these techniques differ in terms of the representation used to describe the video. An important family of methods is the class of action recognition systems using space-time interest points, such as Haris3D [19], Cuboids [6], and SIFT3D [32]. Many other features for human action recognition have been proposed besides space-time interest points. Efros et al. used optical flows to represent and classify actions. Klaser et al. extended HOG [3] to HOG3D by making use of the temporal dimension of videos [14]. Ke et al. learned volumetric features for action detection [13]. Wang and Suter proposed the use of silhouettes to describe human activities [39].

On these representations, a variety of classification models have been applied to recognize human actions: bag-of-word model [25], Hidden Markov Model [10], Metric Learning [35], Support Vector Machines (SVM) [31], Deep Learning [22], Boosting-based approaches [20, 21].

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

3. Approach Overview

We now introduce our approach. While Section 4 describes formally our training and testing procedure, here we explain the approach at a high level using the schematic illustration in Figure 1. During an offline stage, our method learns N_a movement exemplar-SVMs (MEXSVMs), shown on the left side of the figure. Each MEXSVM is a binary classifier optimized to recognize a specific action instance (e.g., "biking", "ski-jetting") and it uses histograms of quantized space-time interest points as low-level features for the classification. The set of learned MEXSVMs are then used as mid-level feature extractors to produce an intermediate representation for any new input video: we evaluate each MEXSVM on subvolumes of the input video in order to compute the probability of the action at different space-time positions in the sequence. Specifically, we slide the subvolume of each MEXSVM exemplar at N_s different scales over the input video. As it will be discussed in Section 4.4, this evaluation can be performed efficiently by using *Integral Videos* [12]. Finally, for each MEXSVM 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 MEXSVM features provide a semantically-rich representation of the video, even simple linear classification models trained on our descriptor achieve good action categorization accuracy, as demonstrated in our experiments.

4. Movement Exemplar-SVMs

Our MEXSVM classifiers are linear SVMs applied to histograms of space-time interest points (STIPs) calculated from subvolumes of the video. We choose this model as it can be efficiently evaluated even on long videos, and as such it is a suitable choice for scalable action recognition.

In principle, to train each SVM classifier we need a reasonable number of both positive and negative examples in order 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 [24]. Specifically, for each pos-

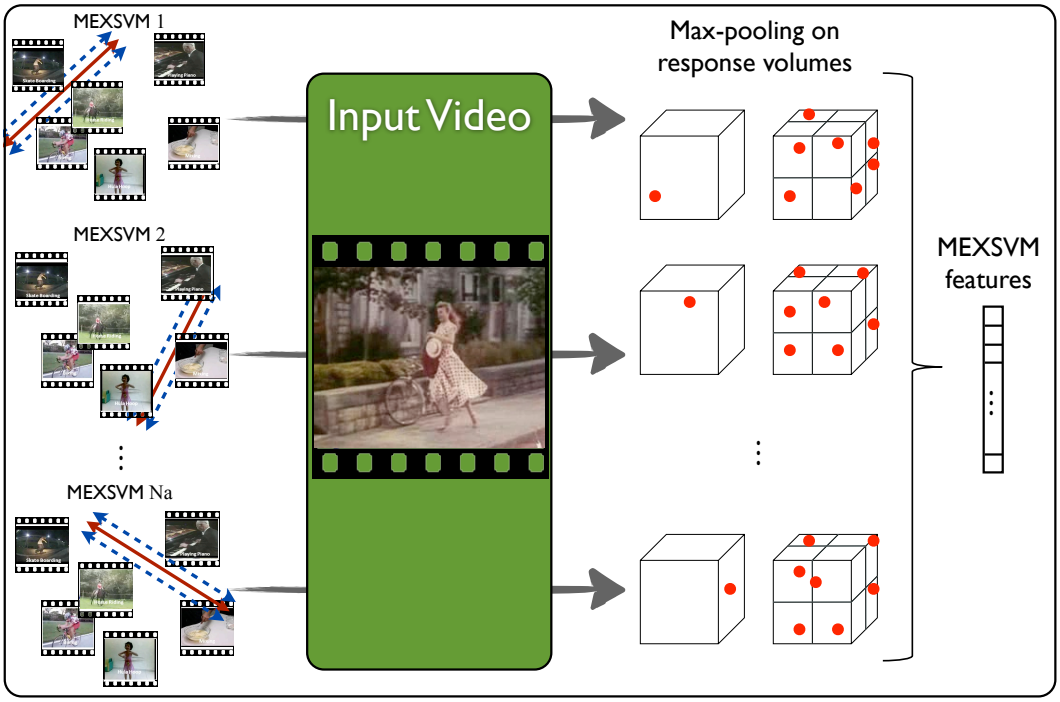


Figure 1. **Overview of our approach.** During an offline stage, a collection of movement exemplar-SVMs (MEXSVMs) is learned. Each MEXSVM is trained using a single positive video exemplar and a large number of negative sequences. These classifiers are then used as mid-level feature extractors to produce a semantically-rich representation of videos. Our approach is similar to the Action-Bank framework [30]. Our novel contribution is the learning of efficient exemplar SVMs, which replace the costly template-based classifiers of Action-Bank in the computation of the mid-level features.

itive 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 STIPs contained in this volume becomes the representation used to describe the positive exemplar. Then, intuitively, 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 [24], 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, in our context, the exemplar-SVMs are simply used as mid-level feature extractors to find movements similar to the positive exemplar so 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, thus playing a role similar to the weak-learners in boosting [37].

In the next subsections we describe in detail the low-level features used by MEXSVMs as well as the procedure to learn the exemplar-classifier and then to test them efficiently on novel videos.

4.1. Low-level features used by MEXSVM

Our MEXSVMs are applied to histograms of quantized STIPs computed within video volumes. In order to learn the dictionary used by the quantizer, we first extract STIPs from a set of training videos using the approach described in [19], which extends to video the traditional Harris operator for still images. We then compute Histogram of Oriented Gradients (HOG) [3] and Histogram of Flows (HOF) [4] within space-time neighborhoods of the detected STIPs using the implementation described in [20]. At each STIP we concatenate the HOG and the HOF descriptor to form a 162-dimensional feature vector representing the interest point. Finally, we run k-means clustering on these vectors to learn a codebook of d cluster centroids. Given the codebook, a video volume is represented in terms of the histogram of codewords occurring within that volume. We normalize the histogram using the L1 norm.

4.2. Training a MEXSVM

The input for learning a MEXSVM consists of a positive video containing a manually-specified 3D box bounding the

action of interest, and thousands of negative videos *without* action volume annotations. As we will describe in further details in the experiment section, 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.

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 elongated in the horizontal direction. 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 only short-and-wide volumes when searching for the swimming action.

Let x_E be the video volume manually-specified in the positive example and $\mathcal{N}_E$ the set of all subvolumes of negative videos. Let us denote with $\phi(x)$ the L1-norm normalized histogram of codewords within a video volume x, i.e., $\phi(x) = \frac{1}{c(x)}\left[c_1(x), \ldots, c_d(x)\right]^T$, where $c_i(x)$ is the number of codeword i occurring in volume x, and $c(x)$ is the total number of codewords in x.

Following [24], the exemplar-SVM training procedure learns a linear classifier $f(x) = w^T\phi(x)+b$, by minimizing the following objective function:

$$\begin{aligned} \min_{w,b} \|w\|^2 \quad &+ \quad C_1 h\big(w^T\phi(x_E) + b\big) \\ &+ \quad C_2 \sum_{x\in\mathcal{N}_E} h\big(-w^T\phi(x) - b\big) \end{aligned} \qquad (1)$$

where $h(s) = \max(0, 1-s)$ is the hinge loss function, and C_1 and C_2 are pre-defined parameters to equalize the unbalanced proportion of positive and negative examples.

Unfortunately, direct minimization of the objective in formula 1 is not feasible since the last term requires optimizing the SVM parameters on all possible negative subvolumes, which is typically a gigantic number making brute-force evaluation impossible. Thus, we resort to an alternation scheme similar to that used in [24] and [8]: we iterate between 1) learning the parameters (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 x_E as positive example and randomly selected subvolumes from the other videos as negative examples. At each iteration the current SVM is evaluated exhaustively on every negative video to find violating subvolumes, i.e., subvolumes yielding a positive SVM score. These subvolumes are added to the active set S to be used in the successive iterations of SVM learning. 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 $M = 7$, but we find that in more than 85% of the cases, the learning procedure converges before reaching this maximum number of iterations (see Figure 3 for further details).

Finally, our training procedure adds to the active set also subvolumes selected from the positive exemplar video: at each learning iteration we evaluate the current SVM on the positive video and add to the active set two kinds of subvolumes. The first kind concerns subvolumes whose spatial overlap with x_E is greater than 50% but that yield a negative classification score: these subvolumes are false negatives and thus are added as positive examples to the active set. We include also any subvolume having spatial overlap with x_E smaller than 20% if the classification score is positive: these are false positives which are added to the active set as negative examples in order to force the learning procedure to recognize the intrinsic properties contained in x_E and not those present in irrelevant subvolumes of the positive video.

The pseudocode of our learning procedure is given in algorithm 1. Lines $1-4$ initialize the active set. The function `svm_training` in line 6 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 false positive volumes detected in negative videos (lines 8–12). In order to maintain the learning of the SVM feasible, in practice for each negative video $\mathcal{V}_i^-$ we add to the active set only the volumes that yield the largest violations, for a maximum of 10 per video.

4.3. Calibrating the ensemble of MEXSVMs

The learning procedure described above is applied to each positive exemplar independently to produce a collection of MEXSVMs. Because of this disjoint training of the exemplar classifiers, 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 [26]: for each exemplar-SVM (w_E, b_E) we learn parameters (α_E, β_E) to produce calibrated probabilities through the sigmoid function $g(x; w_E, b_E, \alpha_E, \beta_E) = 1/[1+\exp(\alpha_E(w_E^T x+b_E)+\beta_E)]$. The fitting of parameters (α_E, β_E) is performed according to the iterative optimization described in [26] using as labeled examples the positive or negative volumes that are in the active set at the completion of the MEXSVM training procedure. As already observed by prior work using the outputs of classifiers as mid-level features for recognition [5, 24], we also found that

Algorithm 1 MEXSVM training

Input: A set of negative videos $\{\mathcal{V}_1^-, \ldots, \mathcal{V}_N^-\}$ and a manually-selected volume x_E in exemplar video $\mathcal{V}^+$.
Output: Parameters (w, b) of exemplar-SVM.

$S \leftarrow \{(x_E, +1)\}$
for $i = 1$ to N **do**
  $S \leftarrow S \cup \{(x_i, -1)\}$ with x_i randomly chosen from $\mathcal{V}_i^-$
end for
for $iter = 1$ to M **do**
  $(w, b) \leftarrow$ `svm_training`(S)
  $S_{old} \leftarrow S$
  for $i = 1$ to N **do**
    for all x in $\mathcal{V}_i^-$ s.t. $w^T x + b > 0$ **do**
      $S \leftarrow S \cup \{(x, -1)\}$ *//false positive*
    end for
  end for
  for all x in $\mathcal{V}^+$ s.t. $w^T x + b < 0$ & $\frac{|x \cap x_E|}{|x_E|} > 0.5$ **do**
    $S \leftarrow S \cup \{(x, +1)\}$ *//false negative*
  end for
  for all x in $\mathcal{V}^+$ s.t. $w^T x + b > 0$ & $\frac{|x \cap x_E|}{|x_E|} < 0.2$ **do**
    $S \leftarrow S \cup \{(x, -1)\}$ *//false positive*
  end for
  if $S_{old} = S$ **then**
    break
  end if
end for

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 MEXSVM scores

Although replacing the template matching procedure of Action Bank by linear SVMs gives 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, we use the sliding volume scheme 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 also need to slide the entire set of MEXSVM classifiers over the test video in order to extract the mid-level features for the subsequent recognition. We next describe an efficient solution to the sliding volume evaluation of the SVMs.

Let $\mathcal{V}$ be an input video of size $R \times C \times T$ where R, C, and T are the numbers of rows, columns, and frames of the video respectively. Given a MEXSVM with parameters (w_E, b_E), we need to efficiently evaluate it over all subvolumes of $\mathcal{V}$ having size equal to the positive exemplar subvolume x_E (in practice, we slide the subvolume at N_s different scales but for simplicity we illustrate the procedure assuming that we use only the original scale). It is worth noting that the branch-and-bound method of Lampert et al. [17] 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 [12] to efficiently compute the MEXSVM 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 $H(x) = \sum_{(r,c,t) \in x} h(r, c, t)$, where (r, c, t) denotes a space-time point in volume 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., $B(r, c, t) = \sum_{r' \leq r} \sum_{c' \leq c} \sum_{t' \leq t} h(r', c', t')$. This buffer can be built with complexity linear in the video size. Once built, it can be used to compute $H(x)$ for any subvolume x via only 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. Let us indicate with $P(x)$ the set of STIPs included in subvolume x of video $\mathcal{V}$ and let i_p be the codeword index of a point $p \in P(x)$. Then we can rewrite the classification score of exemplar-SVM (w, b) on a subvolume x as follows (we omit the constant bias term b for brevity):

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

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 x can be efficiently calculated using two integral videos (one for the numerator, one for the denominator), without ever explicitly computing the histogram $\phi(x)$ or the inner product between w and $\phi(x)$.

5. Experiments

5.1. Experimental setup

Implementation details of MEXSVM training: Since our approach shares many similarities with Action Bank, we adopt training and design settings similar to those used in Action Bank [30] in order to facilitate the comparison between these two methods. Specifically, our MEXSVMs are learned from the same set of UCF50 [1] videos used to build the Action Bank templates. This set consists of

189 sequences spanning a total of 50 actions. Since the Action Bank volume annotations are not publicly available, we manually selected the action volume x_E on each of these exemplar sequences. We removed from this set 4 sequences since their volume occupies almost the entire image in each frame and thus it is not suited to a sliding window approach. We obtain in this way $N_a = 185$ 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. Each MEXSVM is learned by using one of the 185 volumes as single positive exemplar; the negative volumes are drawn from all the negative videos *excluding* those containing the same action class as the category of the positive video. 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 [31]. 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 MEXSVMs. In the experiments we show that, while our descriptor is defined by a smaller number of movement classifiers (185 instead of 205), the recognition performance obtained with our mid-level features is on par with Action Bank.

As already mentioned, we apply the MEXSVM classifier to histograms of local HOG-HOF descriptors computed at STIPs. The dictionary used to quantize these descriptors is learned from the UCF50 video examples, by running k-means using $d = 5000$ centroids.

Parameters of MEXSVM features: In order to compute the MEXSVM features from a new video, we perform max-pooling of the MEXSVM 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 x_E (the scales are 1, 0.75, and 0.5), and $N_p = 73$ space-time subvolumes obtained by recursively splitting the entire video in octrees using 3 levels (this yields 1 volume at level 1, 8 subvolumes at level 2, and 64 subvolumes at level 3). Thus, the final dimensionality of our MEXSVM descriptor is $N_a \times N_s \times N_p = 40515$.

Action classification model: All of our action recognition experiments are performed by training a one-vs-the-rest linear SVM on the MEXSVM descriptors given a set of labeled training videos. We use this classification model as it is very efficient to train and test, and thus it is an appropriate choice for the scenario of large-scale action recognition that we are interested in addressing. We have also experimented with nonlinear kernels to train the SVM (we tried the histogram intersection and the χ^2 kernels), but the results were similar to (but never better than) those obtained with a linear classifier. The same observation is made in [30]. For this reason we present here only results obtained with the linear SVM model. The hyperparameter C of the SVM is tuned via cross-validation for both Action Bank and MEXSVMs.

Method	HMDB51	UT-Interaction
Graph matching [27]	-	70.8
Dynamic BoW [28]	-	78.9
Hough Forests [9]	-	82.5
Propag. Hough Voting [40]	-	**92.5**
Action Bank [30]	**26.9**	85.0
C2 [11]	23.0	-
HOG/HOF [20]	20.0	-
MEXSVMs (our approach)	**26.9**	84.2

Table 1. Comparison of recognition results on HMDB51, and UT-interaction. An efficient linear SVM trained on our MEXSVM features outperforms or approaches the state-of-the-art on both datasets.

Test datasets: We test our approach on two challenging action recognition datasets: HMDB51 [15], and UT-Interaction [29]. These datasets represent good testbeds for scalable, unconstrained action recognition since they contain a large number of videos taken in realistic conditions and spanning many different action classes.

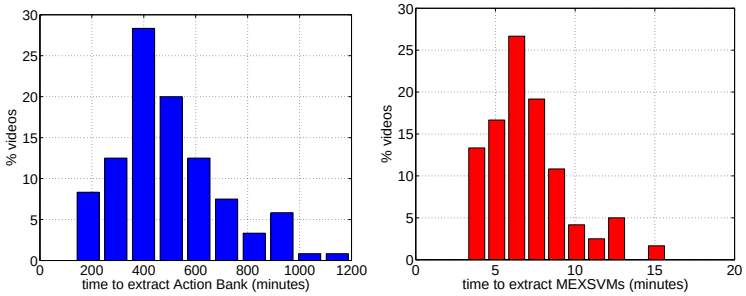
HMDB51 consists of 6849 image sequences collected from various sources, including movies as well as YouTube and Google videos, and containing 51 action categories. All results for this dataset are presented using 3-fold cross validation using the 3 publicly available splits, each containing 3580 training videos and 1540 test examples.

UT-Interaction is a smaller dataset (120 videos) containing sequences of 6 action classes: shake-hands, point, hug, push, kick, and punch. However, this test set is interesting as the actions are very challenging examples of interactions involving multiple people, often including irrelevant pedestrians. We train and test on this dataset using leave-one-out cross validation.

Note that we omit on purpose from our test evaluation the two datasets that were utilized to learn Action Bank or MEXSVMs, i.e., UCF50 [1] and KTH [31]. We decided to do so in order to evaluate the generalization capabilities of the descriptors by considering scenarios where the test database is different from the one used to learn the features, thus avoiding the often misleading effects of dataset training-testing bias [33].

5.2. Action recognition results

We now present the classification performance obtained with our approach and compare it with the best published results on the two benchmarks considered here (HMDB51, UT-Interaction). The recognition accuracies achieved by the different methods are summarized in Table 1. From these results we see that our approach achieves the highest accuracy on HMDB51 together with Action Bank but, as discussed in more detail later, it does so with a 70-fold



Runtimes (minutes)	Action Bank	MEXSVMs
mean	495	7
median	454	7
maximum	1199	16
minimum	132	3

Figure 4. Distribution of runtimes to compute the mid-level descriptors (a) Action Bank and (b) MEXSVMs. Note the very different range of runtime values. The time needed to extract MEXSVMs features for the entire UT-interaction dataset using a single CPU is only 14 hours; instead, it would take more than 41 days to compute Action Bank descriptors for this dataset.

brush hair	73.3	push	31.1	drink	13.3
golf	68.9	talk	31.1	smoke	13.3
situp	65.6	climb stairs	28.9	eat	12.2
pullup	57.8	jump	28.9	kick ball	12.2
hug	54.4	laugh	28.9	shoot ball	11.1
catch	53.3	punch	27.8	shoot gun	11.1
shake hands	52.2	ride bike	25.6	sword	11.1
hit	46.7	dive	24.4	turn	11.1
pushup	46.7	smile	24.4	walk	11.1
fencing	43.3	flic flac	23.3	handstand	10.0
fall floor	38.9	draw sword	22.2	cartwheel	8.9
shoot bow	38.9	chew	18.9	sword exercise	8.9
ride horse	36.7	sit	18.9	swing baseball	6.7
kiss	35.6	somersault	18.9	pick	4.4
pour	34.4	kick	17.8	run	4.4
clap	33.3	stand	17.8	wave	4.4
climb	31.1	dribble	14.4		

Table 2. Recognition accuracy on the individual classes of HMDB51 using linear SVMs trained on MEXSVM features. Note that random chance would yield a recognition rate of only 1.96%.

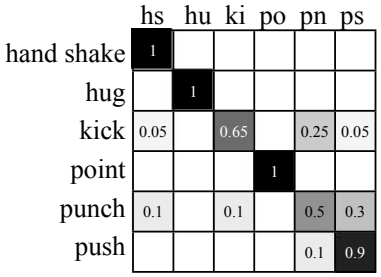
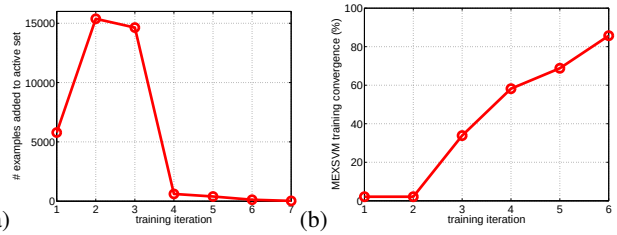


Figure 2. Confusion matrix for the UT-Interaction dataset obtained using linear SVM classifiers trained on MEXSVMs.

speedup (the reader can jump to Figure 4 for a quick look at the computational costs). Table 2 provides a listing of the accuracy achieved with our method on the individual classes of HMDB51.

On the UT-Interaction dataset, our approach is not the best method but even here our approach is a very strong performer, only about 8% worse than the best published result. Figure 2 visualizes the confusion matrix produced by our method on this dataset. Our system confuses most often punching with pushing, which are visually-similar actions.



(a) (b)

Figure 3. Training statistics of movement exemplar-SVMs: (a) the average number of labeled examples added to the active set at different training iterations during the learning of a MEXSVM; (b) the percentage of MEXSVM trainers that have converged as a function of the learning iteration; (c) runtime comparisons between MEXSVMs and Action Bank on UT-Interaction dataset.

This may indicate that more exemplars are needed to describe these two actions as distinct.

In Figure 3, we study the convergence properties of our MEXSVM training procedure. The graph in (a) shows the average number of negative volumes added to the active set as a function of the iteration number. We see that while many new violating volumes are found in the first few iterations, few examples are added after iteration 4. The graph in Figure 3(b) visualizes the percentage of times the MEXSVM training procedure converges for different iterations: after 6 iterations more than 85% of the MEXSVM trainers have converged indicating that few learning iterations are needed for most movement exemplar-SVMs.

5.3. Comparison of computational costs

In this section, we provide an empirical evaluation of the speed of our method and compare it to Action Bank (we use the software provided by the authors of [30]). The runtime statistics for both descriptors were collected on the complete set of videos of UT-Interaction using a single-core Linux machine with a CPU @ 2.66GHz. The distribution of computation times needed to extract the descriptors is shown in Figure 4. Note that in our computation we include the time needed to detect the STIPs and extract the local descriptors, so it is the complete time from the input of the video to the output of the MEXSVMs. The extraction of MEXSVMs is on average over 70 times faster than for Ac-

tion Bank. We can process the entire UT-Interaction dataset using a single CPU in 14 hours; extracting the Action Bank features would take 41 days.

We also provide some illustrative statistics on the efficiency of training and testing classifiers using our MEXSVM features. Training a linear SVM on MEXSVMs for one of the HMDB51 classes takes 6.2 seconds; the testing of the SVM on a video takes only 7 milliseconds. This efficiency of learning and testing makes our descriptor suitable for recognition in large-scale databases, even in scenarios involving many classes.

6. Conclusions

We have presented an efficient approach for large-scale human action recognition. It centers around the learning of a mid-level video representation that enables state-of-the-art accuracy even with simple and efficient linear classification models. Experiments on a large-scale benchmark and a challenging action recognition dataset show the accuracy and efficiency of our approach.

Our mid-level features are produced by evaluating a predefined 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-labeled data, such as YouTube videos. We intend to explore this research direction in the future.

The software implementing our approach and all the data used in the experiments will be released upon publication.

References

[1] http://server.cs.ucf.edu/~vision/data.html. 2, 5, 6

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. *ICCV*, pages 1395–1402, 2005. 1

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *CVPR*, 2005. 2, 3

[4] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. *ECCV*, 2006. 3

[5] J. Deng, A. C. Berg, and F.-F. Li. Hierarchical semantic indexing for large scale image retrieval. *CVPR*, 2011. 2, 4

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

[7] A. Farhadi, I. Endres, D. Hoiem, and D. A. Forsyth. Describing objects by their attributes. *CVPR*, 2009. 2

[8] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*. 4

[9] J. Gall, A. Yao, N. Razavi, L. J. V. Gool, and V. S. Lempitsky. Hough forests for object detection, tracking, and action recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2011. 6

[10] N. Ikizler and P. Duygulu. Human action recognition using distribution of oriented rectangular patches. *ICCV Workshops on Human Motion*, pages 271–284, 2007. 2

[11] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biological inspired system for human action classification. *ICCV*, 2007. 6

[12] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. *ICCV*, 2005. 2, 5

[13] Y. Ke, R. Sukthankar, and M. Hebert. Volumetric features for video event detection. *International Journal of Computer Vision*, 2010. 2

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

[15] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. *ICCV*, 2011. 6

[16] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. *ICCV*, 2019. 2

[17] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2009. 5

[18] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. *CVPR*, 2009. 2

[19] I. Laptev and T. Lindeberg. Space-time interest points. *ICCV*, 2003. 1, 2, 3

[20] I. Laptev, C. S. M. Marszalek, and B. Rozenfeld. Learning realistic human actions from movies. *CVPR*, 2008. 2, 3, 6

[21] I. Laptev and P. Prez. Retrieving actions in movies. *ICCV*, 2007. 2

[22] Q. Le, W. Zou, S. Yeung, and A. Ng. Learning hierarchical spatio-temporal features for action recognition with independent subspace analysis. *CVPR*, 2011. 2

[23] L.-J. Li, H. Su, E. P. Xing, and F.-F. Li. Object bank: A high-level image representation for scene classification & semantic feature sparsification. *NIPS*, 2011. 2

[24] T. Malisiewicz, A. Gupta, and A. A. Efros. Ensemble of exemplar-SVMs for object detection and beyond. *ICCV*, 2011. 1, 2, 3, 4

[25] J. Niebles and L. Fei-Fei. A hierarchical model of shape and appearance for human action classification. *CVPR*, pages 1–8, 2007. 2

[26] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*. MIT Press, 1999. 4

[27] M. Ryoo and J. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. *ICCV*, 2009. 6

[28] M. S. Ryoo. Human activity prediction: Early recognition of ongoing activities from streaming videos. *ICCV*, 2011. 6

[29] M. S. Ryoo and J. K. Aggarwal. UT-Interaction Dataset, ICPR contest on Semantic Description of Human Activities (SDHA), 2010. 6

[30] S. Sadanand and J. J. Corso. Action bank: A high-level representation of activity in video. *CVPR*, 2012. 2, 3, 5, 6, 7

[31] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local SVM approach. *ICPR*, 2004. 1, 2, 6

[32] P. Scovanner, S. Ali, and M. Shah. A 3-Dimensional SIFT descriptor and its application to action recognition. *ACM Multimedia*, pages 357–360, 2007. 2

[33] A. Torralba and A. A. Efros. Unbiased look at dataset bias. *CVPR*, 2011. 6

[34] L. Torresani, M. Szummer, and A. W. Fitzgibbon. Efficient object category recognition using classemes. *ECCV*, 2010. 2

[35] D. Tran and A. Sorokin. Human activity recognition with metric learning. *ECCV*, 2008. 1, 2

[36] A. Veeraraghavan, R. Chellappa, and A. Roy-Chowdhury. The function space of an activity. *CVPR*, pages 959–968, 2006. 1

[37] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *CVPR*, 2001. 3

[38] G. Wang, D. Hoiem, and D. Forsyth. Learning image similarity from flickr groups using stochastic intersection kernel machines. *ICCV*, 2009. 2

[39] L. Wang and D. Suter. Recognizing human activities from silhouettes: Motion subspace and factorial discriminative graphical model. *CVPR*, pages 1–8, 2007. 2

[40] G. Yu, J. Yuan, and Z. Liu. Propagative hough voting for human activity recognition. *ECCV*, 2012. 6