Action Recognition by Learning Mid-level Motion Features

Alireza Fathi and Greg Mori
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
{alirezaf,mori}@cs.sfu.ca

Abstract

This paper presents a method for human action recognition based on patterns of motion. Previous approaches to action recognition use either local features describing small patches or large-scale features describing the entire human figure. We develop a method constructing mid-level motion features which are built from low-level optical flow information. These features are focused on local regions of the image sequence and are created using a variant of AdaBoost. These features are tuned to discriminate between different classes of action, and are efficient to compute at run-time. A battery of classifiers based on these mid-level features is created and used to classify input sequences. State-of-the-art results are presented on a variety of standard datasets.

1. Introduction

In this paper we address the problem of human action recognition from image sequences. We aim to develop algorithms which can recognize low-level actions such as walking, running, or hand clapping from input video sequences. A reliable solution to this problem would allow for a variety of applications such as automated surveillance, human-computer interaction, and video retrieval and search.

This problem has been the subject of a vast amount of research in the computer vision literature. Approaches based on features which describe the entire human figure (e.g. [3, 8]) and those which describe local patches such as hands, feet, or elbows (e.g. [24, 19]) have been developed. As in other object recognition problems, the approaches which use features derived from local patches have the benefit of being robust to clutter and tolerant to global deformation due to varying body shapes and viewpoint. However, these local patches lack descriptive power. Features built at the scale of the entire human figure, for example based on motion [8] or a combination of shape and motion [3], are richer descriptions of human action.

We believe a promising direction lies in between these two types of features – building mid-level features which can be used to recognize actions. The focus of this paper is developing algorithms for discriminatively learning mid-level motion features.

The learning framework we use is based on the shapelet features of Sabzmeydani and Mori [22][1]. In that work, mid-level shape features were constructed from low-level gradient features using the AdaBoost algorithm. In this work, we use motion features based on the work of Efros et al. [8] in this framework, with the goal of recognizing human actions in video sequences.

The main contribution of this paper is the development of an approach for action recognition based on mid-level motion features. We show that this approach gives state-of-the-art performance on a variety of standard datasets (KTH [24], Weizmann [2], soccer [8]) and is computationally efficient.

2. Related Work

Moeslund et al. [16] provide a survey of the literature on action recognition. A variety of approaches use features which describe the motion and/or shape of the entire human figure to perform action recognition. Bobick and Davis [3] develop the temporal template representation which captures both motion and shape, represented as evolving silhouettes extracted using background subtraction. These templates are described using their global Hu moment properties. Cutler and Davis [4] and Liu et al. [14] classify periodic motions. Efros et al. [8] recognize the actions of small scale figures using features derived from blurred optical flow estimates.

Another group of methods uses features derived from small-scale patches, usually computed at a set of interest points. Schuldt et al. [24] compute local space-time features at locations selected in a scale-space representation. These

[1] There was an error in the experiments reported in [22]. In particular, this method does not outperform the HOG method of [5]. For more information please see http://www.cs.sfu.ca/~mori/research/

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

features are used in an SVM classification scheme. Blank et al. [2] represent an action by considering the shape carved by its silhouette in time. Local shape descriptors based on the Poisson equation are computed, then aggregated into a global descriptor by computing moments. Niebles et al. [19] quantize local space-time features and use a "bag-of-words" representation together with Latent Dirichlet Allocation to recognize actions. Rao et al. [21] track hands, and describe actions based on inflection points of these 2D tracks. Nowozin et al. [20] classify image sequences by finding discriminative subsequences of quantized local space-time features. Laptev and Prez [12] find actions in movies. They use AdaBoost to learn a cascade of boosted classifiers from shape and motion features. They use histograms of oriented gradients and optical flows to represent shape and motion respectively.

We build on the mid-level shapelet features [22] in which low-level spatial gradient features are combined to form discriminative mid-level features. These features share similarity with features constructed by biologically inspired methods for object recognition [13, 25, 17]. The work by Serre et al. [25] and Mutch and Lowe [17] use many randomly sampled patches as their features. These features are used as templates in a feed-forward network structure containing alternating stages of template matching and pooling operations for invariance.

The gradient-based learning methods [13] have a similar structure of alternating between layers which mimic simple and complex cell behaviour. LeCun et al. [13] learn a set of convolution kernels to be used as the simple cell features using back-propagation. The shapelet features are more complex in structure, containing combinations of gradients over a mid-level scale; and also discriminatively learned from training data, using positive and negative class examples to decide on their contents. This line of work was recently extended to action recognition by Jhuang et al. [10]. Motion gradients were added as a low-level feature in this model. In our work we show that by using motion features to build mid-level features in the shapelet feature framework, highly accurate action recognition performance, as in [10], can be achieved.

3. Method

Our method for action recognition operates on a "figure-centric" representation of the human figure extracted from an input image sequence. This figure-centric representation is obtained by running a detection/tracking algorithm over the input image sequence. The input to our action recognition algorithm will be a stabilized sequence of cropped frames, centered on the human figure. In our experiments we use a variety of previously developed algorithms to construct this representation – pedestrian detection [22] (KTH), background subtraction (Weizmann), and normalized correlation tracking (soccer).

The use of this figure-centric representation guarantees that we are in fact recognizing actions and not just simple translations. While we will have removed information that would certainly be useful in some surveillance settings, our algorithm can be directly applied to video footage captured with a moving camera, such as the soccer dataset.

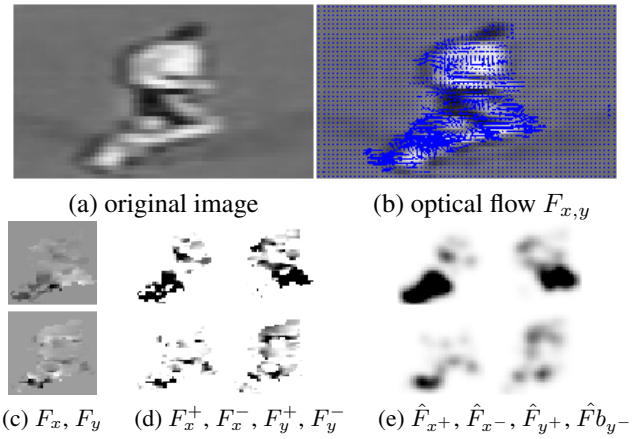
We will consider this sequence of figure-centric frames as a 3D space-time volume for future processing. We will build algorithms which attempt to classify short subsequences (5 frames long in experiments) of this figure-centric volume into low-level actions. In the exposition that follows, we will describe this in terms of a binary classification task, discriminating between two classes. In Section 3.4 we will use standard algorithms for converting our true multi-class task into such binary tasks.

The training part of our approach for learning a classifier to distinguish between two different actions consists of three layers:

1. **Low-level Motion Features as Weak Classifiers:** The input to this layer is a figure-centric volume. We compute low-level motion features on this input sequence, which will be described in Section 3.1. Any of these features, identified by its location and motion direction, can be used as a weak classifier of the two action classes by setting a threshold on its responses. These weak classifiers will be used to make more sophisticated features.

2. **Mid-level Motion Features:** For some small cuboids inside the figure-centric volume, we run Adaboost [27] to select a subset of the weak classifiers inside each figure-centric volume to construct better classifiers. By only using the features inside each cuboid, we force AdaBoost to extract as much information as possible at local neighborhoods of the image sequence. This process, described in Section 3.2, will provide us several mid-level motion features that are intended to be stronger local classifiers than the low-level features and discriminative regarding our action classes. Each mid-level feature consists of a combination of a set of motion directions at various locations.

3. **Final Classifier:** The mid-level features only act in local neighborhoods of the image sequence and therefore their overall classification power is still much below an acceptable level. By merging those features together we can combine the information from different parts of the image sequence. In order to archive this goal, we use AdaBoost for a second time to train our final classifier from the mid-level motion features. AdaBoost will choose the best subset of mid-level motion features that can separate the two action classes. The



(a) original image (b) optical flow $F_{x,y}$

(c) F_x, F_y (d) F_x^+, F_x^-, F_y^+, F_y^- (e) $\hat{F}_{x^+}$, $\hat{F}_{x^-}$, $\hat{F}_{y^+}$, $\hat{F}b_{y^-}$

Figure 1. Constructing the low-level motion features.

construction of the final classifier is described in Section 3.3.

3.1. Low-level Motion Features

To calculate the low-level motion features, we first compute a figure centric spatio-temporal volume for each person. This can be obtained by using any of a number of trackers to track the person and constructing a fixed size window around it. Given the stabilized human figure, the Lucas and Kanade [15] algorithm is employed to compute the optical flow for each frame. The optical flow vector field F is then split into horizontal and vertical components of the flow, F_x and F_y, each of which is then half-wave rectified into four non-negative channels F_{x^+}, F_{x^-}, F_{y^+}, F_{y^-}, similar to Efros et al. [8]. We add another bin corresponding to zero motion F_0 which is obtained by computing the L_2 norm of the four basic channels. These five non-negative channels are then blurred with a gaussian and normalized to obtain the five channels which will be used as our low-level motion features, $\hat{F}_{x^+}$, $\hat{F}_{x^-}$, $\hat{F}_{y^+}$, $\hat{F}_{y^-}$, $\hat{F}_0$, for each frame. Blurring the optical flows reduces the influence of noise and small spatial shifts in the figure centric volume. For each frame, low-level motion features are extracted from optical flow channels at pixel locations in that frame and a temporal window of frames adjacent to it.

These low-level features $\hat{F}_c(p)$ at individual locations p are not capable of discriminating between the positive and negative classes (e.g. two different action categories) much better than random classification. To make them more informative, AdaBoost is used to combine them together to create more informative mid-level features.

3.2. Mid-level Motion Features

Mid-level motion features are weighted combinations of thresholded low-level features. Each low-level feature $\hat{F}_c(p)$ corresponds to a location p in the figure-centric volume, a channel $c \in \mathcal{C} = \{x^+, x^-, y^+, y^-, 0\}$, and has a strength. Each mid-level feature covers a small spatio-temporal cuboid, part of the whole figure-centric volume, from which its low-level features are chosen.

We will consider k such cuboids $w_i \in \mathcal{W}$, $i = 1, \ldots, k$ inside our figure-centric volume. We will build a separate mid-level feature for each cuboid w_i. To do this, we collect all the low-level features that are inside that cuboid $\{f_c^p = \hat{F}_c(p) : p \in w_i, c \in \mathcal{C}\}$ and consider them as potential weak classifiers of an AdaBoost run.

In each iteration t of the AdaBoost [27] training algorithm, one of the features $f_t \in \{f_c^p\}$ is chosen as the feature of the weak classifier $h_t(x)$ to be added to the final classifier. This weak classifier is of the form:

$$h_t(x) = \begin{cases} 1 & \text{if } p_t f_t(x) < p_t \theta_t \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

for a figure-centric volume x, where $\theta_t \in (-\infty, \infty)$ is the classification threshold of the classifier and $p_t = \pm 1$ is a parity for inequality sign.

After all T iterations of the algorithm, we get the final classifier $H_i(x)$ for cuboid w_i. This classifier is of the form:

$$H_i(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t^i h_t^i(x) \geq 0 \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where α_t^i is the selected weight for classifier $h_t^i(x)$, as chosen by the AdaBoost algorithm. We train such a classifier for every cuboid w_i.

Each $H_i(x)$ is a local classifier, containing some of the low-level features inside the cuboid w_i. If we take a second look at the classifier form in equation 2, it can be seen that the weighted sum of weak classifiers is a continuous value. Let us call this sum $s_i(x) = \sum_{t=1}^{T} \alpha_t^i h_t^i(x)$. A useful characteristic about these classifiers is that this $s_i(x)$ contains more information than only specifying the class by its sign. The further away the value of $s_i(x)$ from the zero, the more certain we are about its estimated class. Therefore this value can be used as a confidence measure of the classification. This is similar to the confidence prediction AdaBoost that has been developed by Schapire and Singer [23].

We define our mid-level motion features as these $\{s_i(x) : i \in \{1, 2, \ldots, k\}\}$. The index i corresponds to one of the cuboids $w_i \in \mathcal{W}$, and $h_t^i(x)$ and α_t^i are the parameters associated with the classifier $H_i(x)$. Note that $s_i(x)$ is a mid-level feature that is trained specifically to distinguish between the two action classes, based on low-level motion features from its cuboid.

We train these mid-level features for a set of cuboids inside the figure-centric volume. We visualize the results of the mid-level feature learning algorithm in Figure 2, which shows the weighted sum of all the low-level features selected inside all the mid-level features over the entire the figure-centric volume.

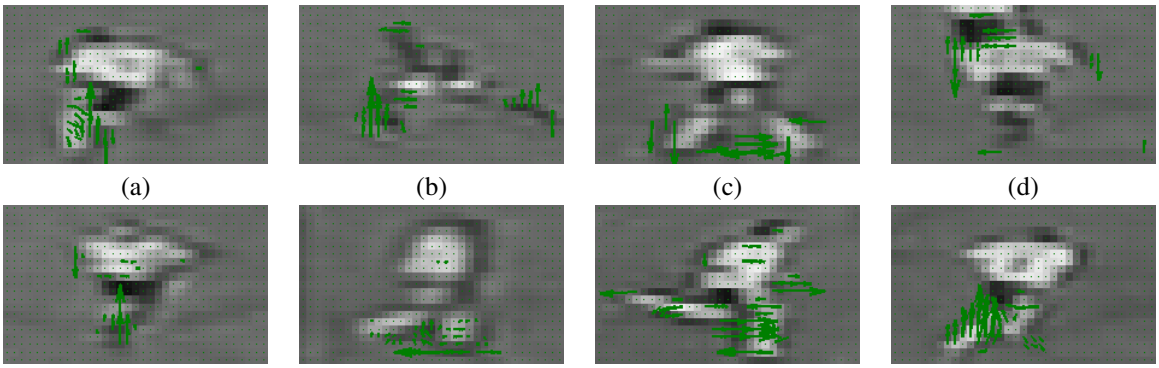


Figure 2. An illustration of low-level features selected and weighted in each mid-level feature across the figure centric volume for the soccer dataset [8]. Set of positive features selected from the final binary classifiers are shown. (a) run left 45^o, (b) run left, (c) walk left, (d) walk in/out, (e) run in/out, (f) walk right, (g) run right and (h) run right 45^o.

3.3. Final Classification

Now that we have defined our mid-level features $s_i(x)$, we use AdaBoost to create a final classifier from them. The details of creating weak classifiers $g_t(s)$ are the same as previous step. Each $g_t(s)$ consists of one of the mid-level motion features $s_t(x)$, a threshold θ_t, and a parity p_t. The final classifier is in the form of:

$$C(s) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t g_t(s) \geq 0 \\ -1 & \text{otherwise} \end{cases} \quad (3)$$

where $s = (s_1(x), s_2(x), \ldots, s_k(x))$ denotes all the mid-level motion features for input figure-centric volume x. Note that this time the weak classifiers $g_t(s)$ (equivalent of $h_t(x)$ in the previous step) are applied in the new feature domain s instead of the low level features x, and therefore the final classifier will be a combination of weighted thresholded mid-level features. Again, the selection of the mid-level feature, threshold, and parity for use in classifier $g_t(s)$, along with its weight α_t, are chosen using the variant of AdaBoost used by Viola and Jones [27].

3.4. Multiclass Action Recognition

AdaBoost is a binary classifier, however, in our problem we are interested in classifying an input figure-centric volume into one of k discrete action classes. A number of approaches exist to reduce the k-way multi-class classification problem into binary classification.

In the simplest approach, each class is compared to all others ("one versus all"). However, because of the huge size of our datasets, this approach is not practical for us for computational reasons. Dietterich and Bakiri [6] propose a general framework in which the classes are partitioned into different subsets using error-correcting codes. Allwein et al. [1] have introduced a unifying approach for margin classifiers. They proposed two methods: *Hamming decoding*, in which each binary classifier essentially contributes a binary vote, and *loss-based decoding* in which the empirical loss, a measure of the confidence of a prediction, is included in the classification decision.

We have experimented with these methods for our final multi-class classification problem. Among them the Hamming decoding scheme outperforms the others in most of the experiments. In this method, we learn a binary classifier using our algorithm, for every pair of classes. Having l binary classifiers, $f_1(x), ..., f_l(x)$, and k class labels, $\hat{y} \in 1, ..., k$, the predicted label will be

$$\hat{y} = \arg\min_r \sum_{s=1}^{l} \frac{1 - sign(M(r,s) f_s(x))}{2} \quad (4)$$

in which, M is a given coding matrix $M \in \{-1, 0, 1\}^{k \times l}$. In our case, M is a matrix which it has k rows and $\frac{k \times (k-1)}{2}$ columns. In each column all the elements are zeros except a 1 for the positive class and -1 for the negative class for binary classifier f_s.

Though we have obtained good performance using Hamming loss, a drawback of the aforementioned methods is that they don't take into account the structure of the output labels – actions classes such as "run left" and "walk left" are "closer" than "run left" and "run right". A variety of methods for handling structured outputs have been developed (e.g. using SVMs [26, 9]) and we believe these would be useful future extensions.

4. Experiments

We have tested our algorithm on four datasets: KTH human motion dataset [24], Weizmann human action dataset [2], Soccer dataset [8], and a ballet dataset collected by ourselves. For the KTH dataset, we have splitted the dataset into training and testing parts. For the other datasets,

we perform "leave one out" cross validation. KTH and Weizmann datasets contain single-action video sequences, and most of the previously published results assign a single label to each sequence (per-video classification). As a result, we will also report per-video classification on these two datasets. The per-video classification is performed by assigning a single action label aquired from majority voting, to a sequence of frames.

The running time of our algorithm is efficient. Our unoptimized implementation in Matlab, exhaustively performs classification with a range of 0.2-4 seconds per frame. Although, the speed of the classifier changes based on the set of parameters described in 4.1, even our most complicated classifier trained for soccer dataset is fast compared to other methods such as nearest neighbor used in [8], since the soccer dataset contains about 4700 frames. The speed of our classifier doesn't change relative to the size of the training data, however, this is not the case in [8].

4.1. Parameters

For each dataset, we manually set a few parameters. These parameters are, total number of weak classifiers , local area of the mid-level features defined by the cuboid w_i, and the size of the strides between the cuboids.

Our mid-level features are learned using the AdaBoost algorithm. There are different ways to limit the number of AdaBoost iterations, such as defining a fixed number of iterations or setting a performance threshold as the ending condition. We fixed the number of weak classifiers selected by each window, by assigning an equal portion of the total number of features to it.

We scan the figure centric volume with the mid-level feature's cuboid size (e.g. 15×15), with the strides of particular size (e.g. 5) between cuboids. This dense scan will provide us many local cuboids inside the figure centric volume. Each cuboid is considered as the effective region of one of the mid-level features. That mid-level feature is learned by using only the low-level features inside its cuboid.

4.2. Evaluation

For each dataset we have evaluated our results by comparing it to previous methods, which are current state of the arts for action classification.

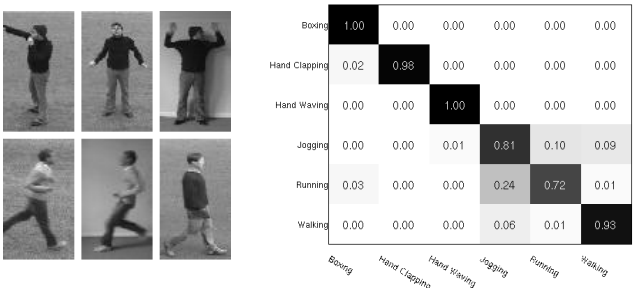
KTH dataset: The KTH human motion dataset, contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping). Each action is performed several times by 25 subjects in four different conditions: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. We first track and stabilize the video sequences using the method in Sabzmeydani and Mori [22]. Representative frames of this dataset are shown in Figure 3(a).

Methods	Training method	Accuracy(%)
Our method	Splits	90.50
Jhuang et al. [10]	Splits	91.70
Nowozin et al. [20]	Splits	87.04
Niebles et al. [19]	Leave one out	81.50
Dollár et al. [7]	Leave one out	81.17
Schuldt et al. [24]	Splits	71.72
Ke et al. [11]	Splits	62.96

Table 1. Comparison of different reported results on KTH dataset.

Unfortunately, most of the previous methods do not describe precisely how they split the KTH dataset into training and testing parts. This makes the fair comparison impractical. In our experiments, we use $\frac{2}{3}$ of the dataset for training and the other $\frac{1}{3}$ for test. We prevent the same subject doing the same action to be in both training and testing portions. The confusion matrix for the per-video classification on the KTH dataset is shown in Figure 3(b). The most confusion is between the last three actions: running, jogging and walking. We have compared our results with the current state of the art in Table 1. Our results are comparable to Jhuang et al. [10] and significantly outperform other methods.

In the KTH dataset, for each frame, we concatenate the motion descriptors of the five adjacent frames to it and use it as our figure centric volume. So the size of the volume will be $85 \times 50 \times 5$. We set the total number of weak classifiers to 1200, mid-level feature size to $15 \times 15 \times 5$, and the length of the strides to 5. Each mid-level feature will consist of 10 weak classifiers. The final classifier runs with a speed of 0.75 seconds per frame. We observed in our experiences, that we get better results by increasing the depth of figure centric volume (concatenating more frames). However, since the KTH dataset has a huge size, it was impractical for us to increase the depth of volume to more than 5.



(a) (b)

Figure 3. Results on KTH dataset: (a) sample frames. (b) confusion matrix for per-video classification (overall accuracy of 90.5%). Horizontal rows are ground truth, and vertical columns are predictions.

Weizmann dataset: The Weizmann human action

	per-frame(%)	per-video(%)
Our method	99.9	100
Jhuang et al. [10]	N/A	98.8
Niebles & Fei-Fei [18]	55	72.8

Table 2. Comparison of the overall accuracy on Weizmann dataset with previous work.

dataset contains 93 low-resolution (180×144 pixels) video sequences showing nine different people, each of which performing 10 different actions. We have tracked and stabilized the figures using background subtraction masks that come with the dataset. In figure 4(a) we have shown some sample frames of the dataset. The confusion matrix of our results for per-frame classification is shown in Figure 4(b). Our method has achieved a 100% accuracy for per-video classification and 99.9% for per-frame classification. Since the confusion matrix for per-video classification, is simply a perfect diagonal matrix, we have just shown the confusion matrix for per-frame classification. We have compared our results with previous work in Table 2. The discriminative low level features are visualized in figure 5.

Soccer dataset: The soccer dataset contains several minutes of a World Cup soccer game which is digitized from an NTSC video tape. We have used the tracked and stabilized dataset provided by Efros et al. [8], which consists of 66 video sequences, each corresponding to a person moving in the center of the field of view. Each frame is hand-labeled with one of the 8 action labels: run left 45^o, run left, walk left, walk in/out, run in/out, walk right, run right, run right 45^o.

This dataset is the very noisy and complicated, and it is hard to discriminate between the classes manually. The run left/right 45^o, are confused by other classes as the running angle significantly varies in different sequences of these two classes. Also it is very hard to discriminate between walking in/out versus running in/out, especially since there are fewer instances of these classes in the dataset. AdaBoost doesn't perform as well in situations that the number of training instances is unbalanced. Since our algorithm had difficulty classifying the walk in/out and run in/out, we merged these two into a single walk/run in/out class. Note that we do much better at certain classes, when we have greater number of frames. We have shown some sample frames of this dataset in Figure 6(a).

We rescale each frame to 3 different sizes. Afterwards, we concatenate the motion descriptors of each frame with its 8 adjacent frames. The size of the volume is $21 \times 35 \times 9$. We have set the total number of the weak classifiers to 2040, mid-level feature size to $4 \times 4 \times 9$, and the length of the strides to 2. The confusion matrix of our results is shown in Figure 6(b). Our results are compared with the results of Efros et al. [8] in Table 3. The results in Figure 6(b) are calculated by applying the Hamming decoding for multi-classification, however, we get 1% accuracy gain by applying the loss-based decoding method of Allwein et al. [1].

Action	F	Our method	Efros et al. [8]
run left 45^o	567	0.63	0.67
run left	567	0.59	0.58
walk left	844	0.86	0.68
walk/run in/out	740	0.89	0.85
walk right	844	0.85	0.68
run right	567	0.65	0.58
run right 45^o	567	0.53	0.66
Overall		0.71	0.67

Table 3. The main diagonal of the confusion matrix of our method and the method in Efros et al. [8] are compared. (F) presents the number of frames existing in the dataset for each action.

Ballet dataset: We have tested our algorithm on a Ballet dataset we collected from an instructional Ballet DVD (unfortunately, the Ballet dataset used in [8] is not available). There are 44 sequences and we have manually labeled each frame with 8 different actions: R-L presenting, L-R presenting, presenting, Jump & swing, Jump, Turn, Step, and Stand still. In this dataset there exist 3 subjects: 2 men and a woman.

The sample frames and the confusion matrix of our results is shown in Figure 7. For the Ballet dataset, we have used the 5 frame concatenation. We have resized all frames to 50×50. We have set the total number of features to 1620, the mid-level feature size to $10 \times 10 \times 5$, and the size of the strides to 5. Our performance on this dataset is not as good as the previous ones, which might be because of the complexity of actions in this dataset, and significant variation in clothing (the woman is wearing a skirt).

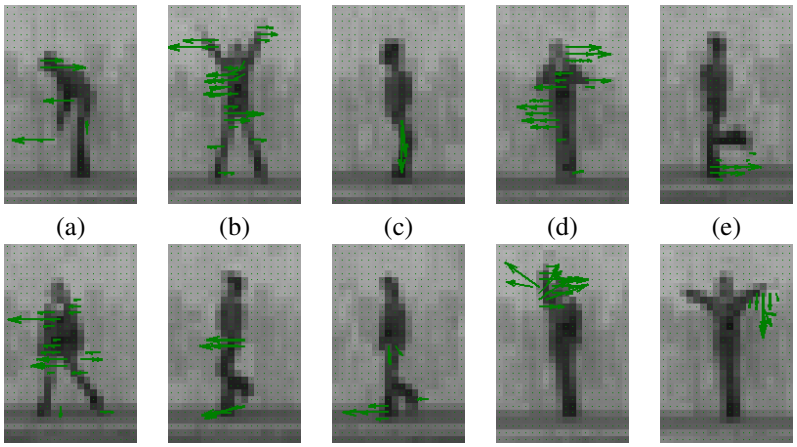
5. Conclusion

In this paper we presented a method for human action recognition using mid-level motion features. These features are computed on a figure-centric representation, in which the human figure is stabilized inside a spatio-temporal volume. These mid-level motion features were constructed from low-level optical flow features computed on this volume. Each mid-level feature is focused on a small cuboid inside the figure-centric volume, and is built from the low-level features which best discriminate between pairs of action classes.

We demonstrated that these mid-level features can be used to obtain action recognition results which are equivalent to the state-of-the-art on the KTH [24] and Weizmann [2] datasets. On the challenging soccer dataset [8], we obtain results which are superior to the nearest-neighbour classifier of Efros et al. [8] on categories for which sufficient

(a) (b)

Figure 4. Results on Weizmann dataset: (a) sample frames. (b) confusion matrix for per-frame classification (overall accuracy of 99.9%).



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 5. The representation of the positive low-level features from the final binary classifiers for the Weizmann dataset. (a) bend, (b) jack, (c) jump, (d) pjump, (e) run, (f) side, (g) skip, (h) walk, (i) wave1 and (j) wave2.

training data exists. The classifiers built using the mid-level features are also computationally efficient, and are much faster than this nearest-neighbour approach.

References

[1] E. Allwein, R. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000. 4, 6

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Proc. 10th Int. Conf. Computer Vision*, 2005. 1, 2, 4, 6

[3] A. Bobick and J. Davis. The recognition of human movement using temporal templates. *IEEE Trans. PAMI*, 23(3):257–267, 2001. 1

[4] R. Cutler and L. Davis. Robust real-time periodic motion detection, analysis, and applications. *IEEE Trans. PAMI*, 22(8), 2000. 1

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 2005. 1

[6] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995. 4

[7] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *VS-PETS*, 2005. 5

[8] A. Efros, A. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Proc. 9th Int. Conf. Computer Vision*, volume 2, pages 726–733, 2003. 1, 3, 4, 5, 6

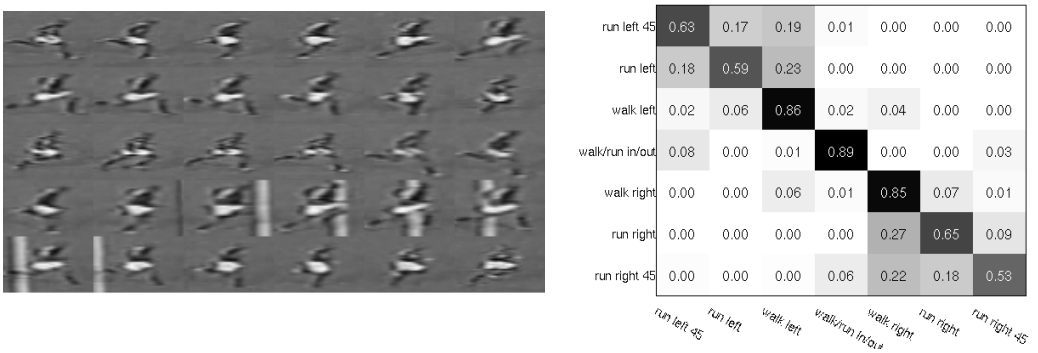
[9] P. Geurts, L. Wehenkel, and F. d'Alch Buc. Gradient boosting for kernelized output spaces. In *ICML*, 2007. 4

[10] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologicallyinspired system for action recognition. In *ICCV*, 2007. 2, 5, 6

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

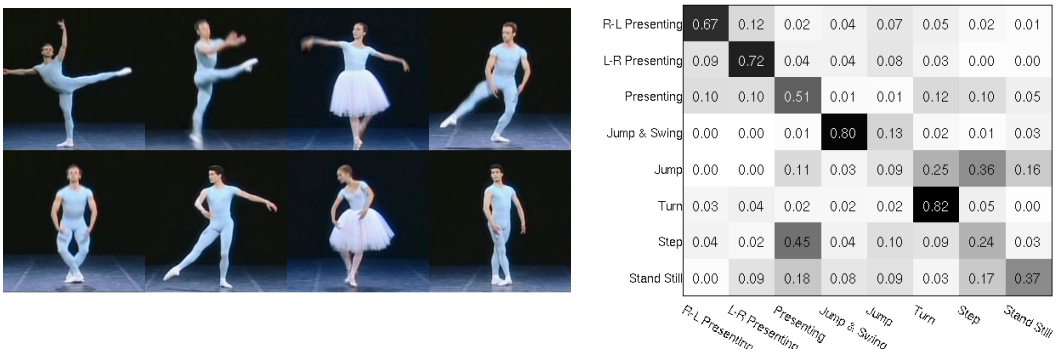
[12] I. Laptev and P. Prez. Retrieving actions in movies. In *Proc. Int. Conf. Comp. Vis.(ICCV'07)*, Rio de Janeiro, Brazil, October 2007. 2

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceed-*



(a) (b)

Figure 6. Results on Soccer dataset: (a) sample frames (run right). (b) confusion matrix for per-frame classification (overall accuracy of 71.3%).



(a) (b)

Figure 7. Results on Ballet dataset: (a) sample frames. (b) confusion matrix for per-frame classification (overall accuracy of 51%).

ings of the IEEE, 86(11):2278–2324, November 1998. 2

[14] Y. Liu, R. Collins, and Y. Tsin. Gait sequence analysis using frieze patterns. In *Proceedings of the 7th European Conference on Computer Vision (ECCV'02)*, May 2002. 1

[15] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA Image Understanding Workshop*, April 1981. 3

[16] T. B. Moeslund, A. Hilton, and V. Kruger. A survey of advances in vision-based human motion capture and analysis. *CVIU*, 104:90–126, 2006. 1

[17] J. Mutch and D. G. Lowe. Multiclass object recognition with sparse, localized features. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 2006. 2

[18] J. C. Niebles and L. Fei-Fei. A hierarchical model of shape and appearance for human action classification. In *CVPR*, 2007. 6

[19] J. C. Niebles, H. Wang, and L. Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. In *BMVC*, 2006. 1, 2, 5

[20] S. Nowozin, G. Bakir, and K. Tsuda. Discriminative subsequence mining for action classification. In *ICCV*, 2007. 2, 5

[21] C. Rao, A. Yilmaz, and M. Shah. View-invariant representation and recognition of actions. *Int. Journal of Computer Vision*, 50(2), 2002. 2

[22] P. Sabzmeydani and G. Mori. Detecting pedestrians by learning shapelet features. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 2007. 1, 2, 5

[23] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *COLT' 98: Proceedings of the eleventh annual conference on Computational learning theory*, pages 80–91. ACM Press, 1998. 3

[24] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *17th International Conference on Pattern Recognition*, 2004. 1, 4, 5, 6

[25] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Trans. PAMI*, 29(3):411–426, 2007. 2

[26] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning*, 2004. 4

[27] P. Viola and M. Jones. Robust real-time object detection. In *2nd Intl. Workshop on Statistical and Computational Theories of Vision*, 2001. 2, 3, 4