Human Action Recognition from a Single Clip per Action

Weilong Yang, Yang Wang, and Greg Mori
School of Computing Science
Simon Fraser University
Burnaby, BC, Canada
wya16@sfu.ca, ywang12@cs.sfu.ca, mori@cs.sfu.ca

Abstract

Learning-based approaches for human action recognition often rely on large training sets. Most of these approaches do not perform well when only a few training samples are available. In this paper, we consider the problem of human action recognition from a single clip per action. Each clip contains at most 25 frames. Using a patch based motion descriptor and matching scheme, we can achieve promising results on three different action datasets with a single clip as the template. Our results are comparable to previously published results using much larger training sets. We also present a method for learning a transferable distance function for these patches. The transferable distance function learning extracts generic knowledge of patch weighting from previous training sets, and can be applied to videos of new actions without further learning. Our experimental results show that the transferable distance function learning not only improves the recognition accuracy of the single clip action recognition, but also significantly enhances the efficiency of the matching scheme.

1. Introduction

The ability to generalize from a small training set is an important feature of any recognition system. While this statement can be made of recognition problems in general, in this paper we focus on human action recognition from video data. There has been much recent progress in this field, with high performance on standard benchmark datasets. However, this level of performance has been achieved with techniques that utilize a large amount of training data. We argue that while this may be appropriate for certain tasks (e.g. action fingerprinting, or distinguishing subtle differences in action), it is problematic to assume that such large training sets exist for the task of discriminating between broadly different categories of actions. In this work we focus on learning to recognize actions from a single clip, leveraging knowledge acquired from previous action categories. We focus on experiments on standard action recognition datasets, though the same principles could be applied to video retrieval and surveillance tasks.

As has been noted in the object recognition literature [8], humans are adept at learning new categories given small numbers of examples. Attempting to endow computer vision algorithms with similar abilities is appealing. This line of research is also of practical importance – gathering volumes of training data for unusual actions, e.g. for surveillance or video retrieval tasks, is an expensive, labour-intensive process.

In this paper we attempt to push the boundaries of action recognition performance with a single, short video clip as training data for a particular action. We work with a *figure-centric* representation in which a human detection and tracking algorithm has been run as a pre-processing step. The main contributions of this paper involve the development of a parameterized distance function for comparing video clips. The distance function is defined as a weighted sum of distances between a densely sampled set of motion patches on frames of the video clips. This distance function is effective for action recognition in the impoverished training data setting. We further develop an algorithm for learning these weights, *i.e*. the distance function parameters. We develop a novel margin-based *transfer learning* algorithm, inspired by the work of Frome *et al*. [11] and Ferencz *et al*. [10]. The learnt weights are a function of patch features and can be generically transferred to a new action category without further learning. This learning greatly improves the efficiency of our algorithm, and can improve recognition accuracy.

2. Previous Work

A variety of action recognition algorithms have obtained high recognition accuracy on the standard KTH [23] and

This work is supported by grants from NSERC.

Weizmann [1] benchmark datasets. The literature in this area is immense; we provide a brief set of closely related work here. The vast majority of these methods use large amounts of training data, with either a leave-one-out (LOO) strategy or other splits of the data involving large amounts of training data for each action.

Efros et al. [4] recognize the actions of small scale figures using features derived from blurred optical flow estimates. Fathi & Mori [7] learn an efficient classifier on top of these features using AdaBoost. Our method uses the same figure-centric representation, and defines patch distances using blurred optical flow. We learn a generic transferrable distance function rather than individual classifiers, on smaller training sets.

A number of methods run interest point detectors over video sequences, and describe this sparse set of points using spatial and/or temporal gradient features[18, 3, 23]. In contrast with these methods, we use a densely sampled set of patches in our distance function. Our transfer learning algorithm places weights on these patches, which could be interpreted as a type of interest point operator, specifically tuned for recognition.

Shechtman and Irani [24] define a motion consistency designed to alleviate problems due to aperture effects. Distances between pairs of video clips are computed by exhaustively comparing patches centered around every space-time point. In our work we learn which patches are important for recognition, leading to a more efficient algorithm – though one could use motion consistency in place of blurred optical flow in a distance function.

Ke *et al*. [15, 16] define a shape and flow correlation based on matching of segmentations. Classification is done using a parts-based model [15] and an SVM trained on template distances in a LOO setting [16].

Jhuang *et al*. [13] describe a biologically plausible model containing alternating stages of spatio-temporal filter template matching and pooling operations. Schindler and Van Gool [22] examine the issue of the length of video sequences needed to recognize actions. They build a model similar to Jhuang *et al*. [13] and show that *short* snippets can be effective for action recognition. Both of these methods use large splits for training data. Our work focuses instead on the amount of data needed, rather than the temporal length of the clips. Weinland and Ronfard [26] classify actions based on distances to a small set of discriminative prototypes selected in a LOO experiment.

Tran and Sorokin [25] propose a metric learning method for action recognition from small datasets. Our experiments use fewer frames (25 per training clip), and compare favourably in terms of accuracy.

Our approach of learning distance functions is inspired by the work of Frome *et al*. [12, 11] and Ferencz *et al*. [10]. The work by Ferencz *et al*. [10] introduces the notion of "hyper-features" for object identification. In a nutshell, hyper-features are properties of image patches that can be used to estimate the saliencies of those patches. These saliency measurements can be used later in matching based object identification. In our work, we use a similar idea to estimate the relative weights (*i.e*. saliency) of motion patches extracted from video frames. We define the hyper-feature of a motion patch using a codebook representation. The main difference of our hyper-feature model with Ferencz *et al*. [10] is that our model is directly tied to the distance function used for the matching.

We use Frome *et al*.'s *focal learning* framework which considers similar-dissimilar triplets of training data points. Rather than learning distance functions specific to each image, we learn them generically based on patch *hyper-features*. This allows us to transfer them to new videos without re-training, and to use them even in cases where we have only one training example for a class (focal learning requires at least 2).

Previous applications of transfer learning in vision include the one-shot learning of Fei-Fei *et al*. [8], in which object recognition models are built using priors learned from previously seen object classes. Farhadi *et al*. [6, 5] use comparative features for transferring distances between templates for sign language and multi-view action recognition. Quattoni *et al*. [21] perform transfer learning using kernel distances to unlabeled prototypes.

3. Motion Descriptors and Matching scheme

We will classify the test video (we will call it *query video*) using the nearest neighbor (NN) classifier after computing the distances between the query video and each clip in the template set. The reason to use the NN classifier is that most other learning based approaches rely on complicated models with a large number of parameters, and thus cannot deal with the situation of very small training sets.

3.1. Motion Descriptors

In this work, we use a figure-centric representation of motion in which a standard human detector and tracking algorithm has been applied. The motion descriptors in Efros *et al*. [4] are used to represent the video frames. We first compute the optical flow at each frame. The optical flow vector field F is then split into two scalar fields, F_x and F_y corresponding to the x and y components of the flow vector. F_x and F_y are further half-wave rectified into four non-negative channels F_x^+, F_x^-, F_y^+, F_y^-, so that $F_x = F_x^+ - F_x^-$ and $F_y = F_y^+ - F_y^-$. Then, those four channels are blurred using a Gaussian kernel to obtain the final four channels Fb_x^+, Fb_x^-, Fb_y^+, Fb_y^-.

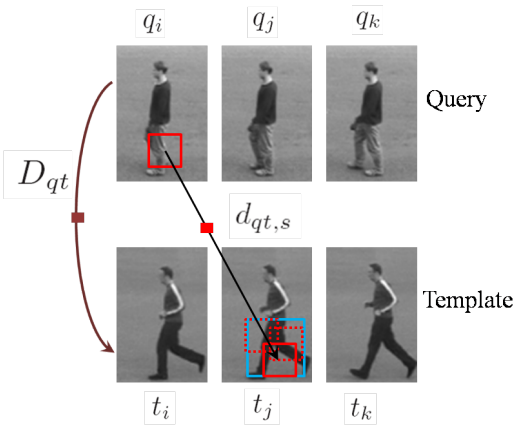


Figure 1. The comparison process between the query and template clips. $d_{qt,s}$ denotes the distance between the s-th patch on the query clip to its corresponding patch on the template clip. D_{qt} denotes the distance between query and template clips. The distance between clips is the sum of the distance from query frames to their best matched template frames. The frame-to-frame distance is the sum of the distance between best matching patches.

3.2. Patch based Action Comparison

We compute the distance between two video clips by comparing the patches from both clips. Patch based methods are very popular in object recognition, due to the fact that local patches are more robust to pose variation than the whole object. We represent each patch using the four channel motion descriptor. Suppose the four channels for patch i are $\mathbf{a}_1$, $\mathbf{a}_2$, $\mathbf{a}_3$, $\mathbf{a}_4$, and each channel has been concatenated to a vector. Similarly, the four channels for patch j are $\mathbf{b}_1$, $\mathbf{b}_2$, $\mathbf{b}_3$, $\mathbf{b}_4$. We denote $\hat{\mathbf{a}_k} = [a_k^1 - \bar{a_k}, a_k^2 - \bar{a_k}, ..., a_k^n - \bar{a_k}]$, and $\hat{\mathbf{b}_k} = [b_k^1 - \bar{b_k}, b_k^2 - \bar{b_k}, ..., b_k^n - \bar{b_k}]$, where $\bar{a_k}$ and $\bar{b_k}$ are the mean values of channel $\mathbf{a}_k$ and $\mathbf{b}_k$ respectively, a_k^i denotes the i-th element in channel vector $\mathbf{a}_k$. The similarity between patch i and j is computed using normalized correlation, and the distance is given by

$$d(i,j) = C - \sum_{k=1}^{4} \frac{\hat{\mathbf{a}_k}^T \hat{\mathbf{b}_k} + \varepsilon}{\sqrt{(\hat{\mathbf{a}_k}^T \hat{\mathbf{a}_k} + \varepsilon)(\hat{\mathbf{b}_k}^T \hat{\mathbf{b}_k} + \varepsilon)}} \qquad (1)$$

where C is a positive constant to make the distance non-negative, and ε is a small constant.

Different people may perform the same action differently. Take the walking action as an example, different people may have different strides, so the legs may appear in different positions of cropped frames. In order to alleviate the effect of such variations, we choose a local area search scheme. It is illustrated in Fig. 1. The distance between query and template clips is:

$$D_{qt} = \sum_{i=1}^{M} \min_{j \in [1,N]} \left\{ \sum_{s=1}^{S} \min_{r \in R_s} d(q_{is}, t_{jr}) \right\} \qquad (2)$$

where q_{is} denotes the s-th patch on the query frame i, and t_{jr} denotes the r-th patch on the template frame j. R_s is the corresponding search region of s-patch (the blue rectangle in Fig. 1). M and N are the frame numbers of query clip and template clip respectively. S is the total number of patches on the query frame.

In order to compute the clip-to-clip distance D_{qt} from query to template, we need to know the frame correspondence first. By considering temporal constraints, one can apply dynamic time warping or other dynamic programming methods. However, in this work, for simplicity, we correspond each query frame to its closest neighbor among the template frames. This can result in several query frames corresponding to the same template frame. But it is reasonable since the query clip may contain repetitive actions and have variations in speed.

After obtaining the frame correspondence and local patch correspondence, D_{qt} is the sum of the elementary patch-to-patch distance as $D_{qt} = \sum_{s=1}^{M \times S} d_{qt,s}$, where $M \times S$ is the total number of patches on the query clip over space and time, $d_{qt,s}$ denotes the distance from the s-th patch on the query clip to its corresponding patch on the template clip.

In Section 6.2, we will show that even with such a simple motion descriptor and matching scheme, we can achieve very good results on three different datasets by only using one clip as template per action. The results are comparable to previously published results using large training sets.

4. Learning a Transferable Distance Function

In the scenario described in Section 1, only one clip is available for each action in the template. However, at the same time, for some simple actions such as walking and hand-waving, a large number of clips can be easily obtained from standard benchmark datasets, *i.e.* KTH and Weizmann datasets. Although the direct comparison of the query and template clips is able to achieve relatively good results, we would still like to exploit the available labeled datasets to assist in action recognition from a single template clip, even when the action of the template clip is totally different from the actions in the labeled datasets. In machine learning, this is known as transfer learning. The goal is to leverage the knowledge from related tasks to aid in learning on a future task.

Following the terminology of transfer learning, we denote the fully labeled action data we already have at hand as the *source training set*. Note that the class labels of actions in the source training set and the template set are different.

4.1. Transferable Distance Function

The human visual system is amazingly good at learning transferable knowledge. For example, humans are adept

at recognizing a person's face after seeing it only once. One explanation for this amazing ability is that people have learnt to focus on discriminative features (e.g., eyes, nose, mouse) of a face, while not being distracted by other irrelevant features [10]. This idea of knowledge transfer has been exploited in the context of object recognition and identification [10, 19, 8]. In particular, Ferencz *et al*. [10] propose to predict the patch saliency for object identification by its visual feature called *hyper-feature*. The relationship between the hyper-feature and the patch saliency is modeled using a generalized linear model.

Similarly, in human action recognition, we believe there exists a certain relationship between the saliency and the appearance of a patch. For example, for a boxing action, the region around the punching-out arm is much more salient than the still leg. In a hand-waving action, the arm parts are salient too. Given a source training set, our goal is to learn the knowledge, such as "stretched-arm-like" or "bent-leg-like" patches are more likely to be salient for action recognition. This knowledge will be "transferable" to unknown actions in the template and query datasets, since the algorithm will look for these patches and assign them high weights for the matching based recognition.

Inspired by work on learning distance function [11], we formulate our problem of learning the relationship into the framework of max-margin learning of distance functions. But the goal of our learning problem is different from that of Frome *et al*. [11]. The output of Frome *et al*. [11] is the weight associated with each image patch in the training data. In our problem, although we do get the weight as a by-product, we are more interested in learning the relationship between the patch appearance and its saliency.

We define the hyper-feature of the i-th patch as $\mathbf{f}_i$, the weight assigned to this patch as w_i. The construction of the hyper-feature will be discussed in Section 5. We assume that $\mathbf{f}_i$ and w_i have the following relationship via the parameter $\mathbf{P}$:

$$w_i = \langle \mathbf{P} \cdot \mathbf{f}_i \rangle \tag{3}$$

Then we will have $\mathbf{w} = \mathbf{P}^T\mathbf{F}$, where each column of $\mathbf{F}$ refers to the hyper-feature vector of a patch, $\mathbf{w}$ denotes the vector which is the concatenation of the weights w_i. Our goal is to learn $\mathbf{P}$ from the source training set. Then given any new action video, even if its action does not exist in the source training set, we will be able to compute the weight (*i.e*. saliency) of each patch in the new video by Eqn. 3. In our work, we would like to estimate the saliencies of patches on the query video.

Combined with the learnt distance function, the final clip-to-clip distance D_{qt} is defined as a weighted sum of all the elementary distances

$$D_{qt} = \sum_{s=1} w_{q,s} d_{qt,s} = \langle \mathbf{w}_q \cdot \mathbf{d}_{qt} \rangle \tag{4}$$

where $\mathbf{d}_{qt}$ is the distance vector, and each element denotes the elementary patch-to-patch distance $d_{qt,s}$. Note $w_{q,s}$ is the weight of the s-th patch on the query clip.

4.2. Max-Margin Formulation

The learning of $\mathbf{P}$ follows the focal learning framework in [11]. The distance function obtained by $\mathbf{w} = \mathbf{P}^T\mathbf{F}$ will satisfy the constraint that the distance between similar actions is smaller than dissimilar actions by the margin 1, that is

$$\begin{aligned} \langle \mathbf{w}_i \cdot (\mathbf{d}_{ij} - \mathbf{d}_{ik}) \rangle > 1 \\ \langle \mathbf{P}^T\mathbf{F}_i \cdot (\mathbf{d}_{ij} - \mathbf{d}_{ik}) \rangle > 1 \end{aligned} \tag{5}$$

where $\mathbf{d}_{ik}$ is the distance vector between the similar action i and k, and $\mathbf{d}_{ij}$ is the distance vector between the dissimilar action i and j. To avoid the problem of large patch-to-patch distances implying a high similarity, we enforce the non-negativity of the weights, $\langle \mathbf{P} \cdot \mathbf{f}_m \rangle \geq 0$. For simplicity, we replace $\mathbf{d}_{ij} - \mathbf{d}_{ik}$ as $\mathbf{x}_{ijk}$.

The max-margin optimization problem can be formulated as

$$\begin{aligned} \min_{\mathbf{P},\xi} \quad & \frac{1}{2}\|\mathbf{P}\|^2 + C\sum_{ijk}\xi_{ijk} \\ s.t.: \quad & \forall i,j,k: \ \langle \mathbf{P}^T\mathbf{F}_i \cdot \mathbf{x}_{ijk} \rangle \geq 1 - \xi_{ijk} \\ & \forall m: \ \langle \mathbf{P} \cdot \mathbf{f}_m \rangle \geq 0 \\ & \forall i,j,k: \ \xi_{ijk} \geq 0 \end{aligned} \tag{6}$$

where ξ_{ijk} is the slack variable and C is the trade-off parameter, similar to those in SVM. The hyper-feature $\mathbf{F}_i$ is known so we can write $\mathbf{Y}_{ijk} = \mathbf{F}_i\mathbf{x}_{ijk}$. The first constraint can be re-written as $\langle \mathbf{P} \cdot \mathbf{Y}_{ijk} \rangle \geq 1 - \xi_{ijk}$.

If we remove the second constraint, the optimization problem in Eqn. 6 will be similar to the primal problem of the standard SVM. The optimization problem is very similar to the one in Frome's work [11], but differs in the second constraint. Instead of the simple non-negative constraint $\mathbf{P} \geq \mathbf{0}$, like the one in [11], our constraints involve linear functions of the hyper-feature vectors.

It helps to solve the problem in Eqn. 6 by examining its dual, we write the dual problem as follows

$$\begin{aligned} \max_{\alpha,\mu} \quad & -\frac{1}{2}\|\sum_{ijk}\alpha_{ijk}\mathbf{Y}_{ijk} + \sum_{m}\mu_m\mathbf{f}_m\|^2 + \sum_{ijk}\alpha_{ijk} \\ s.t. \quad & \forall i,j,k: \ 0 \leq \alpha_{ijk} \leq C \\ & \forall m: \ \mu_m \geq 0 \end{aligned} \tag{7}$$

where the α_{ijk} and μ_m are the dual variables corresponding to the first and second constraints in Eqn. 6 respectively. The primal variable $\mathbf{P}$ can be obtained from the dual variables as

$$\mathbf{P} = \sum_{ijk}\alpha_{ijk}\mathbf{Y}_{ijk} + \sum_{m}\mu_m\mathbf{f}_m. \tag{8}$$

4.3. Solving the Dual

Similar to [11], we solve the dual problem by iteratively performing updating on two dual variables. By taking the derivative of the dual with respect to one of the dual variables α_{abc} and then setting it to zero, we can obtain the updating rule for the dual variable α_{abc}. Similarly, we can get the updating rule for the dual variable μ_a. The two updating rules are as follows:

$$\begin{aligned}\alpha_{abc} &\leftarrow \frac{1 - \sum_{ijk \neq abc} \alpha_{ijk} \langle \mathbf{Y}_{ijk} \cdot \mathbf{Y}_{abc} \rangle - \sum_m \mu_m \langle \mathbf{f}_m \cdot \mathbf{Y}_{abc} \rangle}{\|\mathbf{Y}_{abc}\|^2} \\ \mu_a &\leftarrow \frac{-\sum_{ijk} \alpha_{ijk} \langle \mathbf{Y}_{ijk} \cdot \mathbf{f}_a \rangle - \sum_{m \neq a} \mu_m \langle \mathbf{f}_m \cdot \mathbf{f}_a \rangle}{\|\mathbf{f}_a\|^2}\end{aligned} \tag{9}$$

After each round of update, we can simply clip the dual variables to their feasible regions. α_{abc} will be clipped to 0 if negative and to C if larger than C. μ_m will be clipped to zero if negative. See [12] for more details. After solving this dual problem, we can obtain **P** through Eqn.8.

5. Hyper-Features

Inspired by codebook approaches in object and scene categorization, we represent the hyper-feature of each patch as a $|V|$-dimensional vector $\mathbf{f}$, where $|V|$ is the codebook size. The i-th element of $\mathbf{f}$ is set according to the distance between the feature vector of this patch and the i-th visual word. The feature vector of each patch consists of histogram of oriented gradient (HOG) [2] and patch positions in the form of $h = \{g, x, y\}$, where g denotes the HOG descriptor of the patch. x and y are the coordinates of the patch in the frame. To construct the codebook vocabulary, we randomly select a large number of patches from the source training set, then run *k*-means clustering. The center of each cluster is defined as a codeword. The hyper-feature $\mathbf{f}_m$ for the m-th patch is constructed as follows

$$\mathbf{f}_m(v_i) = \frac{K_\sigma(D(v_i, h_m))}{\sum_{j=1}^{|V|} K_\sigma(D(v_j, h_m))} \tag{10}$$

where $\mathbf{f}_m(v_i)$ denotes the i-th element in the hyper-feature vector $\mathbf{f}_m$. $D(v_i, h_m)$ denotes the Euclidean distance between the i-th codeword and the patch m. K_σ is the Gaussian-shape kernel as $K_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{x^2}{2\sigma^2})$. Note that Eqn. 10 leads to a generalized linear patch weighting model using Gaussian radial basis functions.

6. Experiments

We test our algorithms on three different datasets: KTH human action dataset [23], Weizmann human action dataset [1], and the cluttered action dataset [15]. We first give a brief overview of these three datasets, then present the experimental results.

6.1. Datasets

KTH dataset: The KTH human action dataset contains six types of human actions (boxing, hand-waving, hand-clapping, jogging, running and walking) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. In total, there are 599 videos. Following the original setup, each video is divided into four sequences. After computing the motion descriptor, we run the human detection and tracking using the code provided by Felzenszwalb *et al*. [9]. All the frames and motion descriptors have been cropped to 90×60 and the human figure is put in the center of the frame.

On this dataset, the performance is saturating, with results from $90\% - 94\%$ [22, 13]. However, most of those methods choose either a half-half or leave-one-out cross validation scheme to split the training and testing sets. For example, in each round of the leave-one-out testing, 575 videos are used for training, and the remaining 24 videos are used for testing. Besides, for each video, there are $300-500$ frames in which the actor repeatedly performs one single action. If we assume one complete action lasts 30 frames, the actual training set for the above leave-one-out scheme contains at least 5750 samples, and for each action category, there are 960 samples. In many real-world applications, it is impossible to collect equivalently large training sets for any given action.

Weizmann dataset: The Weizmann human action dataset contains 93 sequences of nine actors performing ten different actions. Each sequence contains about $40 - 120$ frames. In the figure-centric representation, all frames have been normalized to 90×60. The best performance published is 100% by using the large training set [7].

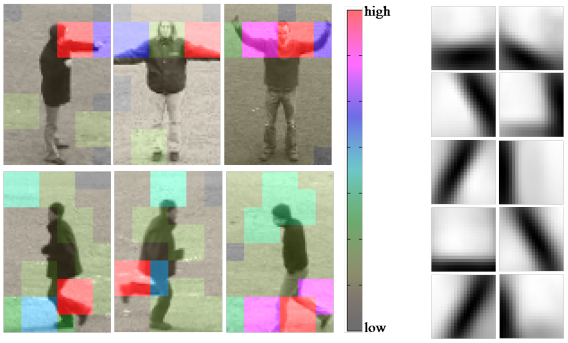
Cluttered human action dataset: The cluttered human action dataset is a variant of the dataset collected by Ke *et al*. [15], which was initially designed for action detection in the crowed environment. It contains not only cluttered static backgrounds, but also cluttered dynamic backgrounds, such as moving cars or walking people. In order to test the robustness of our action recognition methods, we use it for recognition. From each raw video sequence in the original dataset, we manually crop out the actions of interest. This dataset contains 95 sequences with five actions, jumping jack, pushing elevator button, picking-up, one-hand waving, and two-hand waving. Each sequence contains about $30 - 50$ frames. Representative frames are shown in Fig. 2.

6.2. Experimental Results

We perform the following experiments to evaluate our patch based comparison method and the transferable distance function learning:

1. Evaluate the patch based comparison method on all three datasets;

Figure 2. Sample frames of cluttered human action dataset [15]



(a) (b)

Figure 3. (a) Illustration of the learnt weights on the six actions of KTH. (b) The learnt **P** allows us to rank the visual words in the vocabulary. The top ten words are visualized. Note that our visual words consist of appearance and spatial location features. Only appearance is illustrated. Please refer to text for more details.

2. Train the transferable distance function on Weizmann, and test on KTH;
3. Train the transferable distance function on KTH, and test on the cluttered action dataset.

Direct Comparison on KTH. In this experiment, we evaluate the patch based direct comparison method on the KTH dataset. We first randomly select one actor, then randomly choose one clip per action from this actor as the template set. The clip contains at most 25 frames, *i.e*. $1 - 1.5$ complete action cycles. The sequences of the remaining actors are used as the query set. We decompose each frame into 40 patches. The patch size is 20×20 and the length of strides is 10.

We run the experiment five times and for each round we select a different actor as the template. The results are shown in the row of **Dc** of Table 1. The average result over the five rounds is 72.48%, which is comparable to the previously published results using large training set, as shown in Table 2. Note that due to the action variation in person, the performance depends on how distinguishable the templates are.

Training on Weizmann and Testing on KTH. In this experiment, we train a transferable distance function from Weizmann and test it on KTH. In order to meet the requirement of the transfer learning scenario, *i.e*. the source training set does not contain the actions of the template set, we remove walking, running, and two-hand waving from the Weizmann dataset. We build the codebook vocabulary on the remaining sequences of Weizmann as described in Section 5. The number of codewords is set to 100. We used other codebook sizes and found they do not affect the performance substantially. In training, the parameters are set as, $\sigma = 0.5$ and $C = 0.0001$. Through training on Weizmann, we can obtain the relation **P**, which parameterizes the transferable distance function .

After the training, we first compute the hyper-features of the query videos in KTH using the codebook constructed from Weizmann. Then, we can obtain the distance function through Eqn. 3. For the purpose of illustration, we visualize

	1	2	3	4	5	Avg.	Std.
Dc	0.776	0.709	0.829	0.564	0.746	0.725	0.100
Tr	0.784	0.767	0.829	0.617	0.789	0.757	0.073

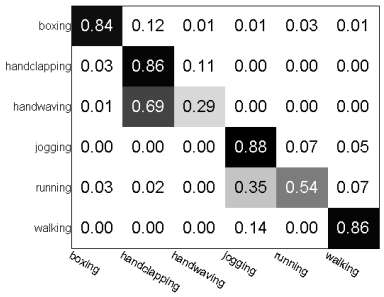
Table 1. The accuracy of five rounds of experiments on KTH. The top row denotes the round index. The row of **Dc** refers to the results of direct comparison, and the row of **Tr** refers to the results of training on Weizmann and testing on KTH. Std. denotes the standard deviation.

the learnt weights in Fig. 3(a). The red patches refer to high weights. Note that patches on the frames are overlapping, we only show the highest weight for an overlapping region. For the six actions in KTH, we can see most of the patches with high weights lie on the most salient human parts, such as out stretched arms or legs. Unlike other motion based interest point detection methods [18], the learnt weight for the moving body is lower than moving legs. This is more intuitive since the moving body does not help to distinguish running, jogging and walking. Moreover, the learnt **P** allows us to rank the visual words in the codebook vocabulary. We visualize the appearance feature of top ten words in Fig. 3(b). We can observe that these words are all "outstretched-limb-like".

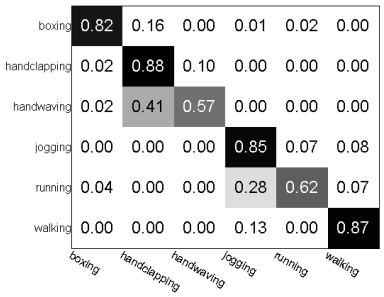
The recognition accuracies of five rounds of experiments are given in the row of **Tr** of Table 1. Note that for each round, we use the same templates as the direct comparison experiments. The largest improvement made by the transferable distance function is almost 6%. We can observe that in experiment round 1 and 3, the improvements made by the transferable distance function are minor. This is reasonable since the direct comparison has already achieved very good results. We also show the confusion matrices of experiment round 2 in Fig. 4. We can see that the transferable distance function significantly mitigates the confusion of the most difficult actions, such as hand-clapping *vs*.

methods	accuracy	remark
Liu & Shah [17]	0.9416	LOO
Schindler & Van Gool [22]	0.9270	LOO
Jhuang *et al*. [13]	0.9170	Split
Nowozin *et al*. [20]	0.8704	Split
Neibles *et al*. [18]	0.8150	LOO
Dollar *et al*. [3]	0.8117	LOO
Ours (**Tr**)	**0.7571**	**One clip**
Ours (**Dc**)	**0.7248**	**One clip**
Schuldt *et al*. [23]	0.7172	Split
Ke *et al*. [14]	0.6296	Split

Table 2. Comparison of different reported results on KTH. We remark the setup of the training set. LOO refers to the "Leave-one-out" cross validation. Split refers to other split strategies of training and testing sets. Note that these numbers are not directly comparable due to variations in training/testing. setup



(a)

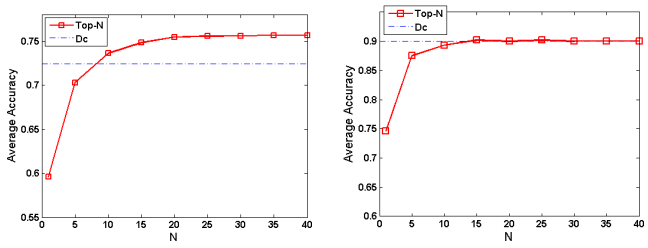


(b)

Figure 4. Confusion matrices on KTH of experiment round 2. Horizontal rows are ground truths, and vertical columns are predictions. (a) Direct comparison. (b) Training on Weizmann and testing on KTH.

hand-waving, and jogging *vs*. running. In particular, we see an improvement of almost 30% for the hand-waving. The comparison with previously published results are given in Table 2.

Another benefit of learning transferable distance function is that it can be used to speedup the comparison. In the patch based direct comparison method, for each patch on the query frame, we need to search its corresponding area on the template frame and find the best matched one. This process is time-consuming since there exist 1000 patches over



(a) (b)

Figure 5. (a) The average accuracy of five rounds of experiments on KTH using only top N patches of each frame; (b) The average accuracy of five rounds of experiments on cluttered action dataset using only top N patches on the frame. The dash-dot line denotes the average accuracy of the direct comparison using all patches.

	1	2	3	4	5	Avg.	Std.
Dc	0.928	0.892	0.916	0.819	0.795	0.870	0.060

Table 3. The accuracy of five rounds of experiments on Weizmann using patch based direct comparison. The top row denotes the round index. Std. denotes the standard deviation.

	Dc	1NN [25]	1NN-M [25]
FE-1	0.8699	0.5300	0.7231

Table 4. Comparison of the average accuracy on Weizmann using one exemplar per action with [25].

the sequence of 25 frames. With learnt distance function of the query sequence, we can sort the patches on each frame by their weights. Instead of using all patches for matching, we only choose the top N patches with high weights from each frame. We change N from 1 to 40 and compute the average accuracy over the five rounds of experiments. The results are illustrated in Fig. 5 (a). Using only ten patches on each frame, we can achieve a better result than the patch-based direct comparison using all patches on the frame. This would save $3/4$ matching time, significantly increases the efficiency of whole recognition process.

Direct Comparison on Weizmann The setup we use in this experiment is exactly the same as the direct comparison experiment on KTH. In each round of the experiment, we randomly select one actor and use one clip per action with 25 frames from this actor as the template. The sequences of the remaining actors are used as the query set. The results are shown in Table 3. We compare our results with the work of Tran and Sorokin [25], as shown in Table 4. Our result outperforms both "1-Nearest Neighbor + motion context descriptor (1NN)" and "1-Nearest Neighbor with metric learning + motion context descriptor (1NN-M)". Note that we only use a 25 frame clip as the template rather than the whole video as in [25].

Unfortunately, a fair transfer learning experiment training on KTH and testing on Weizmann is not possible. After

	1	2	3	4	5	Avg.	Std.
Dc	0.944	0.900	0.844	0.900	0.911	0.900	0.036
Tr	0.944	0.900	0.856	0.900	0.900	0.900	0.031

Table 5. The accuracy of five rounds of experiments on the cluttered human action dataset. The top row denotes the round index. Std. denotes the standard deviation.

removing overlapping actions, there are only three actions left in the KTH (boxing, hand-clapping and jogging). The number of actions is too small to contain enough generic knowledge. So we do not run the experiments of training on KTH and testing on Weizmann.

Direct Comparison on cluttered action dataset. The goal of this experiment is to evaluate the robustness of our patch based direct comparison on more challenging datasets with cluttered backgrounds. For each action, we randomly choose one clip with 25 frames as the template and the remaining sequences as the query set. The same patch decomposition scheme is used. Similarly, we perform five rounds of experiments by choosing different templates. The results are shown in the **Dc** row of Table 5. We can see the patch based direct comparison achieves very high accuracy on this dataset.

Training on KTH and testing on cluttered action dataset. This experiment follows the same protocol as training on Weizmann and testing on KTH. We first remove the two-hand waving action from KTH since it also exists in the cluttered action dataset. KTH contains a large number of sequences, we choose only five actors' sequences to form the source training set. The results are shown in the **Tr** row of the Table 5. As expected, the transferable distance function learning achieves almost identical results as the direct comparison, since direct comparison has achieved very promising results. However, the transferable distance function can be used to sort the patches and choose the patches with top N highest weights, and thus improve the efficiency of the recognition system. As illustrated in Fig. 5(b), we are able to use only top 5 patches on each frame and achieve 86.67% accuracy. The efficiency is boosted significantly (saving 7/8 matching time) with the cost of only 3% accuracy decrease.

7. Conclusion

In this paper we have presented an action recognition algorithm based on a patch-based matching scheme. A set of motion patches on input query clips and template clips with known actions is matched. This matching scheme proves to be effective for action recognition in the difficult case of only a single training clip per action. Further, we have demonstrated that learning a transferable weighting on these patches could improve accuracy and computational efficiency. These weights, based on patch hyper-features, are generic, can be directly applied to novel video sequences without further learning, and hold promise for recognition in small training set scenarios such as video retrieval and surveillance.

References

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

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

[3] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *ICCV Workshop on VS-PETS*, 2005.

[4] A. A. Efros, A. C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *ICCV*, 2003.

[5] A. Farhadi, D. Forsyth, and R. White. Transfer learning in sign language. In *CVPR*, 2007.

[6] A. Farhadi and M. K. Tabrizi. Learning to recognize activities from the wrong view point. In *ECCV*, 2008.

[7] A. Fathi and G. Mori. Action recognition by learning mid-level motion features. In *CVPR*, 2008.

[8] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE T-PAMI*, 28(4):594–611, April 2006.

[9] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008.

[10] A. Ferencz, E. Learned-Miller, and J. Malik. Learning to locate informative features for visual identification. *IJCV*, 2006.

[11] A. Frome, Y. Singer, and J. Malik. Image retrieval and classification using local distance functions. In *NIPS*, 2007.

[12] A. Frome, Y. Singer, F. Sha, and J. Malik. Learning globally-consistent local distance functions for shape-based image retrieval and classification. In *ICCV*, 2007.

[13] H. Jhuang, T. Serre, L. Wolf, and T. Poggio. A biologically inspired system for action recognition. In *ICCV*, 2007.

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

[15] Y. Ke, R. Sukthankar, and M. Hebert. Event detection in crowded videos. In *ICCV*, 2007.

[16] Y. Ke, R. Sukthankar, and M. Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Visual Surveillance Workshop*, 2007.

[17] J. Liu and M. Shah. Learning human actions via information maximization. In *CVPR*, 2008.

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

[19] E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. In *CVPR*, 2007.

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

[21] A. Quattoni, M. Collins, and T. Darrell. Transfer learning for image classification with sparse prototype representations. In *CVPR*, 2008.

[22] K. Schindler and L. Van Gool. Action snippets: How many frames does action recognition require? In *CVPR*, 2008.

[23] C. Schuldt, L. Laptev, and B. Caputo. Recognizing human actions: a local SVM approach. In *ICPR*, 2004.

[24] E. Shechtman and M. Irani. Space-time behavior based correlation. In *CVPR*, 2005.

[25] D. Tran and A. Sorokin. Human activity recognition with metric learning. In *ECCV*, 2008.

[26] D. Weinland and E. Boyer. Action recognition using exemplar-based embedding. In *CVPR*, 2008.