Adaptive Structured Pooling for Action Recognition

Svebor Karaman¹
svebor.karaman@unifi.it
Lorenzo Seidenari¹
lorenzo.seidenari@unifi.it
Shugao Ma²
shugaoma@bu.edu
Alberto Del Bimbo¹
alberto.delbimbo@unifi.it
Stan Sclaroff²
sclaroff@bu.edu

¹ MICC (Media Integration and Communication Center)
University of Florence
Florence, Italy
² Boston University
Boston, USA

Abstract

We propose an adaptive structured pooling strategy to solve the action recognition problem in videos. Our method aims at individuating several spatio-temporal pooling regions each corresponding to a consistent spatial and temporal subset of the video. Each subset of the video gives a pooling weight map and is represented as a Fisher vector computed from the soft weighted contributions of all dense trajectories evolving in it. We further represent each video through a graph structure, defined over multiple granularities of spatio-temporal subsets. The graph structures extracted from all videos are finally compared with an efficient graph matching kernel. Our approach does not rely on a fixed partitioning of the video. Moreover, the graph structure depicts both spatial and temporal relationships between the spatio-temporal subsets. Experiments on the UCF Sports and the HighFive datasets show performance above the state-of-the-art.

1 Introduction

Automatic human action recognition in videos is an important and popular research topic in computer vision, with potential applications in video analytics, video retrieval and video surveillance. While near perfect performance has been achieved in simplistic lab video datasets [1, 2], recognizing human action in realistic videos such as sports videos and TV programs is still quite challenging due to camera and object motion, background distraction, occlusion and viewpoint variation. To tackle these issues, several local space-time features have been proposed, e.g. space-time interest points [3] and dense trajectories [4].

The most common approach to exploit these features is to build a Bag-of-Words [4] like representation of the whole video. These representations rely on the definition of a vocabulary of local space-time features. Then the local features are encoded with respect to the vocabulary. The Fisher encoding method [7] has proven to be very powerful when applied on
images and recently has been applied successfully in videos. All top performers [8, 16, 32] of the recent THUMOS Action Recognition challenge used Fisher encoding.

However, many action recognition approaches discard the space, time and hierarchical relationships among the local video subvolumes that these space-time features represent. These overlooked relationships may constitute discriminative structures of local space-time features, which could be very useful for correctly recognizing human actions, e.g. complex but structural movements of different body parts in sports such as bench swing or an interaction between two persons. Indeed, a video contains highly dynamic content, and the visual content corresponding to semantic concepts of the target class may appear at different position, time and speed for different videos of the same class. Hence, the computation of a single global representation may be harmed by some noisy elements in the surrounding (both spatially and temporally) of the event of interest.

In this work, we aim at exploiting multiple dynamic pooling regions that adapt to the spatial layout and dynamics of each video. Each pooling region is represented through a soft Fisher encoding. The relationships between the pooling regions are captured in a graph structure that represents the whole video. Our method is illustrated in Fig. 1. The video structures are efficiently comparable with the state-of-the-art GraphHopper kernel [4]. Our adaptive structured pooling shows an improvement over the state of the art for action recognition.

2 Related works

The most powerful video representation is based on local spatio-temporal features either sampled on a regular grid [33] or along trajectories [34]. Typically local features are encoded following a bag-of-words approach such as [12, 14, 31, 34]. Although simple and relatively successful, these methods largely left the relationships among local features unexplored, and thus potentially omitted discriminative information contained in such relationships.
To better exploit local features and their relational information, many works have sought to encode or model space-time structures of local features. Some works explicitly model local feature neighborhoods, such as [6, 10, 15]. Kovashka et al. [10] construct higher-level features composed of local features and their neighborhoods. Class-specific distance functions are learned for defining neighborhoods so that they cover the most informative configurations. Gilbert et al. [6] group simple 2D corners spatially and temporally using a hierarchical process, and discriminative features are then mined at each stage of that hierarchy. In [15] quantized local features are augmented with relative space-time relationships between pairs of features. More recently, Ma et al. [14] proposed the hierarchical space-time segments (HSTS) that can extract both static and non-static relevant human body regions and exclude irrelevant background regions.

Another major branch of works, such as [17, 22, 29, 30, 37] learn structural models for human actions. For instance, Wang et al. [37] use a hidden conditional random field to model a human action as a constellation of parts, where the model parameters are learned in a max-margin framework. Raptis et al. [22] cluster dense trajectories and learn a graphical model per action where the nodes correspond to latent variables selecting appropriate dense trajectory clusters for an action. Some recent works propose to represent a video as a graph [1, 13] of local features and use graph kernel or graph matching techniques for action classification. For example, in [13] the nodes are clusters of dense trajectories and the edges represent temporal relationships, and the random walk graph kernel is used for training action classifier. In [1] the nodes are space-time subvolumes of video and the edges depict the space, time and hierarchical relationships among the subvolumes, and action classification is cast as a graph matching problem.

Approaches that use local features for action recognition typically extract a set of features from a video, while the feature set cardinalities can vary from video to video. To get fixed size feature vectors from videos to ease the learning step, e.g. SVM classifier learning, a feature pooling step is usually used. The straightforward approach is to pool features from the whole video [14, 31, 34], but the space-time configuration of the local features is then lost. To encode the space and time configuration information among the local features, one popular approach is to pool features within pre-defined fixed sized space-time grids [14, 23]. In [14] space-time interest point descriptors are pooled within space-time grids that are arranged as a space-time pyramid, and in [23] action detector responses are pooled in grids of a similar pyramid. Simple temporal grids are also used [14, 23], where local features are pooled in temporal segments of a video. More recently, Fisher vector encoding method has been effectively applied as a feature encoding method for action recognition [14, 16, 18, 28, 31, 32, 36], but all these methods use fixed grids (or the whole video) for pooling the encoded Fisher vectors. Obviously, such fixed grids do not adapt to the content of the video, especially the spatial-temporal layout of the regions that contain the human action, so it is quite possible that significantly different feature vectors may be produced through feature pooling of videos that contain the same action.

Instead of using fixed grids, some methods perform feature pooling within space-time clusters of local features such as [5, 22]. In [22] dense trajectories are grouped according to their space-time overlaps and then feature pooling is performed on feature descriptors of group members. In [5] a hierarchical clustering is carried out on tracklets in a video and feature pooling is done in every cluster in the cluster tree to make a Bag-of-Words tree. Our pooling method is closely related to this line of research, but with the following three important distinctions. First, we cluster HSTS [5], which also preserves both the moving and static relevant regions within a video, while the clusters in [5, 22] are based on trajectories
that focus on moving regions and relevant static regions may be missing. Secondly, in [5, 22] each feature is weighted equally during pooling, while we use HSTS to compute a pooling map that emphasizes the regions of human action and the features are weighted according to the map. In this way, the pooling result may better represent the human action contained in the video. Finally, our structured representation is not limited to be a tree and the node representation is not limited to histograms as in [5]. This flexibility allows us to model different kind of relationship among video regions. Moreover our approach does not need to learn a specific graph representation for each action as [17, 22, 29, 30, 37].

3 Adaptive structured pooling

In this section, we first formalize how to apply Fisher encoding with a weighting of the local features in 3.1. We then detail our strategy to obtain pooling regions and weights in 3.2.1. We rely on HSTS to define spatio-temporal pooling regions (STPR). Each pooling region defines a weighting map for all local features to be encoded in a video. We further define several granularities of STPR and a graph on top of them, thus yielding a structural representation of the whole video.

3.1 Fisher encoding with soft pooling

A Fisher vector representation requires a generative model for local features in a video. In our case we fit a Gaussian Mixture Model $u_{\lambda}$ on a set of randomly sampled local features from videos. The Fisher kernel between two sets of local features $X$ and $Y$ is defined as $K_{FV} = G^X_\lambda F^{-1}_\lambda G^Y_\lambda$, where $F_\lambda$ is the Fisher information matrix of $u_{\lambda}$ and $G^X_\lambda$ is the gradient of the log-likelihood of the data $X$ with respect to the parameters $\lambda$ of the generative model $G^X_\lambda = \nabla_\lambda \log u_{\lambda}(X)$.

To obtain the Fisher vector representation one can apply the Cholesky factorization of $F^{-1}_\lambda = L_\lambda^T L_\lambda$ and defining $G^X_\lambda = L_\lambda G^X_\lambda$ we can write $K_{FV}(X,Y) = G^X_\lambda G^Y_\lambda$. Assuming that descriptors in $X$ are independent, the Fisher vector of a video $X$ is a normalized sum of gradients at each point $x \in X$ with respect to the model parameters $\lambda$:

$$G^X_\lambda = \sum_{x \in X} L_\lambda \nabla_\lambda \log u_{\lambda}(x)$$

Fisher vectors are usually computed for a set of features from a whole video or image [18, 21]. The ideas of spatial pyramid (SPM) and spatio-temporal pyramid matching (STPM) can be easily adapted simply computing Fisher encoding over the subset of features that have coordinates inside the boundary of the pyramid sub-region. In [6] a weighted Fisher encoding approach has been proposed for object detection in image, where the weights are based on multiple segmentation outputs. In this work we formalize a flexible way of pooling for Fisher vectors that adapts to arbitrary spatio-temporal regions in video.

We obtain soft-pooling by computing a weight $w_m$ for each feature $x_m \in X$ to encode. Given the GMM $u_{\lambda} = \sum_{n=1}^{N} \omega_n u_n(x; \mu_n, \sigma_n)$ and the $M$ features of $X$, we compute for each
component $u_n$ the mean $G_n^\mu(X)$ and covariance elements $G_n^\sigma(X)$ of a Fisher vector as:

\begin{align}
G_n^\mu(X) &= \frac{1}{\sqrt{\omega_n}} \sum_{m=1}^{M} w_m \gamma_n(x_m) \left( \frac{x_m - \mu_n}{\sigma_n} \right), \\
G_n^\sigma(X) &= \frac{1}{\sqrt{2\omega_n}} \sum_{m=1}^{M} w_m \gamma_n(x_m) \left( \frac{(x_m - \mu_n)^2}{\sigma_n^2} - 1 \right),
\end{align}

where $\gamma_n(x_m)$ is the posterior probability of the feature $x_m$ for the component $n$ of the GMM:

\begin{equation}
\gamma_n(x_m) = \frac{\omega_n u_n(x_m)}{\sum_{j=1}^{N} \omega_j u_j(x_m)}. \tag{4}
\end{equation}

### 3.2 Spatially and temporally structured pooling of a video

In this section, we first present how we define pooling regions and corresponding weights from a set of HSTS in 3.2.1. We then detail our approach to obtain a structured representation of the video as a graph linking local pooling regions in 3.2.2.

#### 3.2.1 Soft weighted pooling regions

Given a video, we extract a set of HSTS $S$ via the method in [14], which comprises two major steps: hierarchical video frame segment extraction and video frame segment tracking.

In the first step, a hierarchical segmentation [1] is produced for each video frame using both motion and color channels [13]. After some initial pruning of oversized or undersized segments, a set of segment trees $T_t$ is extracted from each video frame ($t$ is the frame index). Each segment tree $T_i^t \in T^t$ is considered as a candidate segment tree of human body and we denote $T_i^t = \{s_{ij}^t\}$ where each $s_{ij}^t$ is a video frame segment. $T^t$ is then pruned using shape, motion and color cues. Note that each $T_i^t \in T^t$ is either pruned altogether or retained with all its segments. For example, all segments in $T_i^t = \{s_{ij}^t\}$ will be preserved as long as at least one segment $s_{ij}^t$ contains motion. In this way, relevant but static segments can be maintained, e.g. the torso in a hand shaking action.

In a second step, every segment $s_{ij}^t$ of all surviving segment trees of a video frame is tracked separately both forward and backward in time in the video sequence based on its motion and color. Finally, the space-time segment $s_{ij}^t$ is the set of bounding boxes obtained from the tracking process. Since the number of segments may be large and many of them cover non-static body parts, the tracking procedure is designed to be efficient and allow non-rigid deformation of the target segment. Due to the extraction algorithm described above, the resulting HSTS set $S = \{s_{ij}^t \mid \forall t, i, j\}$ is dense with many overlapping space-time segments.

Given a set of HSTS $S_k \subseteq S$ we can define a weighted pooling map $M_k$ by accumulating how many segments of $S_k$ are present in each frame at each position. Formally, let us denote $S_k^t$ all segments of $S_k$ existing at frame $t$. For every pixel $p = (x, y)$ of frame $t$, we compute the corresponding pooling map value $M_k^t(p)$ as the count of segments enclosing this position:

\begin{equation}
M_k^t = \sum_{s \in S_k^t} \Psi_s \tag{5}
\end{equation}
where for each segment \( s \in S_k \) we define the function of pixels \( p \) in an image

\[
\Psi_s(p) = \begin{cases} 
1 & \text{if } p \in s \\
0 & \text{otherwise}
\end{cases}
\]  

(6)

The pooling map \( M_t \) is further normalized by the total number of segments in the frame and square-rooted. This pooling map represents at any moment of the video, how much each pixel is relevant with respect to the set \( S_k \). The more segments overlap in one position the more likely this pixel is significant for the action taking place. A pooling map with all segments of the corresponding frame is depicted in Fig. 2. Finally, for a video with \( T \) frames we define the spatio-temporal pooling map as

\[
M_k(x,y,t) = \{M^1_k(x,y) \ldots M^T_k(x,y)\}
\]  

(7)

For each local feature \( x_m \) to be encoded, we estimate the weight \( w_m \) with respect to set \( S_k \) as a small local integral of the pooling map \( M_k \) around its centroid. The parameters \( (v_x, v_y, v_t) \) are respectively the horizontal, vertical and temporal span of integration. That is for each \( x_m \in X \) with the spatio-temporal coordinates of its centroid being \((x_{xm}, y_{xm}, t_{xm})\), \( w_m \) is estimated as:

\[
w_m = \int_{x_{xm}-v_x}^{x_{xm}+v_x} \int_{y_{xm}-v_y}^{y_{xm}+v_y} \int_{t_{xm}-v_t}^{t_{xm}+v_t} M_k(x,y,t) \, dx \, dy \, dt
\]  

(8)

Finally, all weights of a pooling region are normalized to sum to one in order to have a comparable representation no matter how many features are present within the region.

### 3.2.2 Structured representation of videos

The previous section detailed how to obtain weights to encode local features from a given set of HSTS. We want to build a structured representation of each video. We therefore need to divide the whole set of segments into meaningful subsets. We propose to find coherent subsets by grouping together segments according to their overlap. This will create a set of local (both spatially and temporally) pooling regions.

We first compute an affinity matrix \( A \) of all segments \( S \) of a video. The affinity of two segments \( s_i \) (alive from frame \( t_{is} \) to \( t_{ie} \)) and \( s_j \) (alive from frame \( t_{js} \) to \( t_{je} \)) is computed as:

\[
A(s_i, s_j) = \frac{1}{\min(t_{ie}-t_{is},t_{je}-t_{js})} \sum_{t \in [\max(t_{is},t_{js}), \min(t_{ie},t_{je})]} \frac{s_i^t \cap s_j^t}{s_i^t \cup s_j^t}
\]  

(9)
The affinity between two segments will be equal to one only if a segment is temporally fully included in the other and with a perfect spatial overlap in every frame of coexistence. The affinity is zero if two segments do not overlap in any frame. Given this affinity matrix we run the normalized cuts algorithm \[26\] to obtain the subsets of segments.

Instead of choosing one fixed number of subsets, we propose to use multiple increasing sizes that will each provide a set of finer local representations of the video. We hence run the normalized cuts algorithm with different number of clusters (namely 4 and 16 subsets) to obtain multiple levels of representation. This setting is inspired by the spatial pyramid usually defined with corresponding numbers of cells. Given all the HSTS clusters obtained from the previous step (including the complete set of HSTS as the single cluster of the first level), we can build a graph representing the whole video. We represent each cluster as a node in the graph, and each node attribute is the soft pooling of dense trajectories features weighted by the map computed on all segments of this cluster. We link clusters based on their overlap, we create a link between all clusters that have at least a pair of overlapping segments (even partially). An illustration of one video graph is shown in Fig. 1.

To compare the video graphs we use the efficient GraphHopper kernel from \[4\]. The GraphHopper kernel is a scalable kernel that can deal with vector values for the nodes attributes. The comparison between two graphs relies on the similarity of the nodes and the computation of shortest paths in each graph. Formally, given two graphs \(G = (V, E)\) and \(H = (V', E')\) the GraphHopper kernel \(K_{gh}(G,H)\) is decomposed as a sum of node kernels:

\[
K_{gh}(G,H) = \sum_{v \in V} \sum_{v' \in V'} w(v,v') k_n(v,v')
\]

where \(w(v,v)\) counts the number of times \(v\) and \(v'\) appear at the same hop in shortest paths of equal discrete length and \(k_n(v,v')\) is a kernel between the node attributes, as proposed \[4\]. In our experiments, \(k_n(v,v')\) is a simple dot product between the soft-pooled Fisher vectors of \(v\) and \(v'\).

## 4 Experiments

### 4.1 Experimental protocol

We test our approach on two widely-used action recognition benchmark datasets. The UCF Sports Dataset contains 150 videos and 13 classes. Experiments are typically conducted using either the split proposed in \[11\] or using leave-one-out (LOO). The three “Golfing” classes are merged in a single “Golf” class and the same is done for “Kicking” actions. Considering the smaller size of this dataset, we augmented the training data by flipping training videos as in \[34\]. Performance is reported as mean per class accuracy.

The HighFive dataset is a challenging dataset of human interactions. The dataset is collected from TV series and contains 4 classes plus a set of negative examples. The dataset contains 300 videos and experiments are performed following the split proposed by \[19\]. Results are reported in terms of mean average precision on the 4 classes of interest.

We employ features computed along improved dense trajectories \[31\]. We use Fisher vector coding for HOG, HOF, MBH and Trajectory (TR) features similarly to \[18\]. We apply PCA to each descriptor vector preserving 64 components for HOG, HOF, MBH and 20 for TR and train a GMM with 256 components on a random set of 200k samples for each of the five descriptors. Each graph node is represented with a soft-pooled Fisher vector, as
described in Section 3.1, with $L^2$ and power normalization [21]. The parameters $v_x$, $v_y$ and $v_t$ are all set to 7, the rational for $v_t$ being that dense trajectories are defined over 15 frames.

As a feature fusion strategy we adopt a simple kernel fusion approach. For each feature $f$ we compute a graph matching kernel $K_f(G,H)$, comparing graphs $G$ and $H$. Each kernel is normalized by applying:

$$K'_f(G,H) = \frac{K_f(G,H)}{\sqrt{K_f(G,G)K_f(H,H)}}$$  \hspace{1cm} (11)

The final kernel, for an ensemble of features $\mathcal{F}$, is $K(G,H) = \sum_{f \in \mathcal{F}} K'_f(G,H)$. Finally, a one-vs-all SVM is learnt for each class using the fused kernel.

4.2 Results

Table 2 reports results on the UCF Sports dataset. On this easier dataset we get very strong results. Our FV baseline gets 89.4% (88.6% using LOO) accuracy while the structured representation obtains 90.8% (90.4% using LOO). Note that our approach does not rely on any person bounding box annotation.

Table 1 reports the results for the HighFive dataset. In this challenging dataset we obtain an improvement of 3% mean average precision (mAP). Note that our Fisher vector baseline (FVB) is very close to previous state-of-the-art but our structured representation improves over our FV baseline by 4%. We include results with our proposed method using only the MBH feature channel to get a fair comparison with [5, 14] that only exploit the MBH feature.

Our method performs better and requires less supervision than [13], which needs discriminatively trained head pose estimators and people detectors. Our approach also has an edge over the methods of [5] and [14], which also extract the spatio-temporal structure in the video. The method of Ma et al. [14] exploit space-time segments to sample features but discard spatio-temporal relationships. The video model proposed by Gaidon et al. [5] only captures hierarchical relationship in a top-down manner while our method models the relationship between video sub-volumes at different granularity. To the best of our knowledge these are the best results on these datasets.

Fig. 3 shows the top five ranked videos for each action on the HighFive dataset. The videos are represented by their central frame for simplicity. The top two videos are correct for all actions, while the top five videos are correct for both “highFive” and “hug”. The fourth video for the kiss action is actually a video of the hug action, but one can see that these two actions do share similar characteristics. Note also that this central frame may hide ambiguous elements happening before or after that frame within the video.

5 Conclusions

We have introduced a powerful and generic representation of videos for action recognition. Our structured representation is adaptive to the content of the video and does not rely on a fixed partition of neither space nor time. Our method only requires video level label annotation. Indeed we exploit an unsupervised procedure to generate a structured representation of the video. Our representation jointly models the hierarchical and spatio-temporal relationship of videos without imposing a strict hierarchy. Moreover, our method does not require a specific learnt graph model for each action.
Table 1: Comparison with the state of the art on the HighFive dataset. Results are reported as mean average precision over the 4 classes.

<table>
<thead>
<tr>
<th>Method</th>
<th>mean AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our (all features)</td>
<td>65.4</td>
</tr>
<tr>
<td>Our (MBH only)</td>
<td>62.8</td>
</tr>
<tr>
<td>FVB (all features)</td>
<td>61.3</td>
</tr>
<tr>
<td>Gaidon et al. [5]</td>
<td>62.4</td>
</tr>
<tr>
<td>Ma et al. [14]</td>
<td>53.3</td>
</tr>
<tr>
<td>Wang et al. [34]</td>
<td>53.4</td>
</tr>
<tr>
<td>Laptev et al. [12]</td>
<td>36.9</td>
</tr>
<tr>
<td>Patron-Perez et al. [19]</td>
<td>42.4</td>
</tr>
</tbody>
</table>

Table 2: Comparison with the state of the art on the UCF Sports dataset. Results are reported as mean per-class accuracy over the 10 classes.

<table>
<thead>
<tr>
<th>Method</th>
<th>LOO</th>
<th>Split</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our (all features)</td>
<td>90.4</td>
<td>90.8</td>
</tr>
<tr>
<td>Our (MBH only)</td>
<td>88.3</td>
<td>90.0</td>
</tr>
<tr>
<td>FVB (all features)</td>
<td>88.6</td>
<td>89.4</td>
</tr>
<tr>
<td>Lan et al. [11]</td>
<td>83.7</td>
<td>73.1</td>
</tr>
<tr>
<td>Kovashka et al. [10]</td>
<td>87.3</td>
<td>-</td>
</tr>
<tr>
<td>Klaser et al. [9]</td>
<td>86.7</td>
<td>-</td>
</tr>
<tr>
<td>Wang et al. [34]</td>
<td>85.6</td>
<td>-</td>
</tr>
<tr>
<td>Yeffet et al. [38]</td>
<td>79.3</td>
<td>-</td>
</tr>
<tr>
<td>Rodriguez et al. [19]</td>
<td>69.2</td>
<td>-</td>
</tr>
<tr>
<td>Wang et al. [35]</td>
<td>-</td>
<td>85.2</td>
</tr>
<tr>
<td>Ma et al. [14]</td>
<td>-</td>
<td>81.7</td>
</tr>
<tr>
<td>Raptis et al. [22]</td>
<td>-</td>
<td>79.4</td>
</tr>
<tr>
<td>Tian et al. [30]</td>
<td>-</td>
<td>75.2</td>
</tr>
</tbody>
</table>

Figure 3: Top five ranked videos (leftmost is higher ranked) by our method for each action (handShake, highFive, hug and kiss) of the HighFive dataset.
Experiments conducted on two standard datasets for action recognition show a significant improvement over the state-of-the-art. In the future, we would like to see if our structured representation could also be used to solve the action localization problem by identifying the paths and/or nodes that are most relevant for the action.

References


