Action-Gons: Action Recognition with A Discriminative Dictionary of Structured Elements with Varying Granularity

Yuwang Wang[1]⋆, Baoyuan Wang[2], Yizhou Yu[3],Qionghai Dai[1], Zhuowen Tu[4]

yw.wang2011@gmail.com, baoyuanw@microsoft.com
yizhouy@acm.org, qhdai@tsinghua.edu.cn, ztu@ucsd.edu

[1]BBNC Lab, Dept. of Automation, THU, [2]Microsoft Research
[3]Dept. of Compute Science, HKU, [4]Dept. of CogSci, UCSD

Abstract. This paper presents "Action-Gons", a middle level representation for action recognition in videos. Actions in videos exhibit a reasonable level of regularity seen in human behavior, as well as a large degree of variation. One key property of action, compared with image scene, might be the amount of interaction among body parts, although scenes also observe structured patterns in 2D images. Here, we study high-order statistics of the interaction among regions of interest in actions and propose a mid-level representation for action recognition, inspired by the Julesz school of n-gon statistics. We propose a systematic learning process to build an over-complete dictionary of "Action-Gons". We first extract motion clusters, named as action units, then sequentially learn a pool of action-gons with different granularities modeling different degree of interactions among action units. We validate the discriminative power of our learned action-gons on three challenging video datasets and show evident advantages over the existing methods.

1   Introduction

Human action recognition has received an increasing amount of attention in the computer vision community as developing a practical action recognition system is vital for many applications ranging from mobile applications, surveillance, interactive gaming, to video annotation and retrieval. Although much progress has been made in the past few years, existing approaches are still far from being satisfactory and practical. Similar to the situation in other recognition problems, finding the right representation is still the key to deal with the challenges due to intra-class variations in viewing condition, illumination, spatial and temporal scale, and camera motion.

To tackle the problem of action recognition, one often designs novel low-level features [1] or applies learning techniques to mine discriminative mid-level or

---

⋆ This work was done when Yuwang Wang was an intern at Micrsoft Research
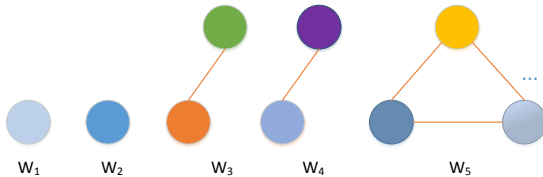
Fig. 1. Action-gons with different granularities for modeling different degrees of interaction. Action-gons shown on real images can be found in Figs. 3 and 6.

high-level features, as in [2, 3]. Recent advances [4–6, 2] show that learning mid-level action units, which are essentially spatio-temporal regions of interest, leads to large improvement in performance.

Actions in videos observe sparse, well-structured, strong temporal coherence, and dynamic interactions, which are different from objects and scenes in 2D images. First, human actions consist of coordinated movements of body parts and accessories. For example, a high jump requires precisely coordinated movements of the arms, torso and legs. It is important to model the co-occurrences and interactions among the movements of different body parts. Second, the number of interacting parts in different actions may vary. For example, the aforementioned high jump involves all body parts while certain actions, such as drinking, only involve the upper body and arms. A complex action, such as a gymnastic routine, can be decomposed spatially and temporally into a number of elementary movements, each of which may involve a different number of body parts. In addition, in the absence of 3D information, 2D videos taken from different viewpoints typically need multiple modes. Thus, we need to study statistics with respect to a variety of interactions for action analysis. In the past, not much attention has been given to explicitly characterizing the statistics of the interactions among regions of interests in actions. And there has been even little attempt to explore the graph-based dictionary with varying granularity to capture the intrinsic motion structures for action recognition.

The Julesz school of two-gon and tri-gon statistics models the intrinsic patterns of textures [7], which has rarely been studied lately in computer vision for action analysis. Our work is inspired by the n-gon statistics of Julesz. We first perform unsupervised learning to extract a set of informative motion clusters, called action units, from videos. Our focus is then to learn a dictionary of interactions among the units, named as "Action-Gons", which model co-occurring and potentially interacting regions of interest, as shown in Fig. (1). To account for varying degrees of complexity, different orders of interaction for the action units are studied. The number of co-occurring action units within an action-gon is defined as its granularity.

For the detailed pipeline of our work, we first partition the motion trajectories into canonical clusters, each of which is called an action unit; then we sequentially learn the interaction structures with varying granularities for each action class to

form our proposed action-gons. Specifically, each action-gon is defined as a latent graph structure with nodes representing action units and edges representing interactions between action units. The feature vector for an action-gon includes features for the individual nodes, as well as features for the edges. Latent graphs with the same granularity for the same action class are learned simultaneously. To achieve better discriminative representation power, a classifier is eventually learned for each action-gon, which is used as an entry in the dictionary. During the recognition stage, we apply the classifier associated with every action-gon on the video and perform max pooling over the response maps to generate features for final action classification.

We validate our framework for learning action-gons on three challenging video datasets : HMDB51 [8], Youtube [9], and UCF-Sports [10]. Extensive experiments show that our approach achieves significantly improved performance over existing state-of-the-art methods, indicating the effectiveness of having a mid-level dictionary of part interactions.

In summary, this paper makes the following three contributions: 1) We propose to learn action-gons with varying granularities for action recognition, inspired by the Julesz n-gon statistics which has not been frequently studied. Each action-gon characterizes co-occurring and interacting regions of interest. 2) We introduce a principled learning method for building an informative yet discriminative dictionary of action-gons. 3) Building on top of the action-gon representation, our overall method produces significantly improved results on benchmark datasets for action recognition.

2   Related Work

Conventional video-based action recognition methods typically extract sparse (sometimes dense) spatio-temporal interest points (STIP) [11] and compute low-level appearance and motion features, followed by training a classifier on top of either the Bag-of-Words (BoW) model or the SPM feature representation. The most popular low-level features, including HOG/HOF [12], HOG3D [13], and MBH [1], indeed turn out to be very informative yet discriminative representations. Recently, to overcome the drawbacks of the traditional Bag-of-Words model, researchers [14, 15] have proposed to add pairwise spatial-temporal relational features between quantized base features (i.e.,HOG/HOF) to express potential interactions. A similar idea has also been employed in [16] to explore contextual features. Although those methods have been proven to be effective on certain datasets, they still lack the flexibility to adaptively infer and localize the most discriminative action parts. Therefore, they do not necessarily derive the optimal representation for characterizing natural human interactions.

Complementary to this line of effort, people have also been trying to develop better feature learning methods or models to tackle the challenges caused by differing scales, viewpoints, illumination or even occlusions. For example, action recognition by learning middle level features has recently become a popular research topic in computer vision, a variety of methods [2, 4–6, 17] have been
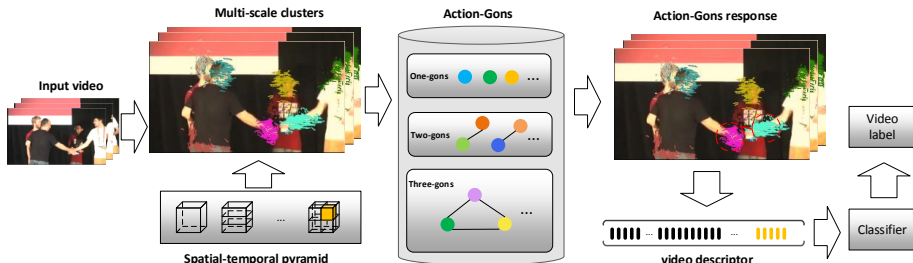
Fig. 2. Our pipeline for building the mid-level action-gon representation and training action classifiers.

proposed to learn the so-called "action parts". Such work can be roughly divided into two categories. One is based on unsupervised learning, and the other is based on supervised or weakly supervised learning.

Action Bank [3] performs action recognition by comparing a test video against a collection of manually designed action templates. Only low-level features are used during such comparison. In contrast, our work learns a mid-level representation by extracting abstract information from low-level features. Inspired by Action Bank [3], motionlets [4] adopt unsupervised learning to discover action parts. Instead of manually designing action templates, it identifies action parts with motion saliency detection and a greedy motionlet ranking method. The method in [18] also adopts unsupervised learning to learn an AND-OR grammar to build the contextual relationships for semantic video understanding, which can be regarded as one typical way of middle level feature learning.

More recently, [17] proposes to harvest middle level parts via weakly supervised learning. However, this method does not incorporate interactions among body parts, which are vital for complex human action modeling. Interactions between pairwise trajectory clusters are modeled as latent graphs in [2]. Nonetheless, it always adopts a single fixed graph structure (e.g. 3 nodes), which becomes inadequate for representing complex human actions. In comparison, this paper builds a mid-level dictionary, each word of which is defined as a graph with a potentially distinct granularity to capture one typical interaction among action units. Given a testing video, our method adaptively localizes potential interactions via optimization.

3   Action-Gon Dictionary Learning

3.1   Overview

In this section, we present a systematic approach for learning multi-granularity action-gons, each of which corresponds to a graph structure. The graph structure is a mid-level representation that models the joint occurrence and interaction among action units. In addition, to make the action-gons sufficiently diverse so

that they can handle different scales, viewpoints as well as different numbers of interacting units, we study graphs with different granularities. For example, a single-node graph describes a relatively simple action unit while a two-node graph can model a higher complexity of interaction and co-occurrence. We first extract dense trajectories from each video, and then apply a multi-scale clustering method to obtain trajectory clusters that serve as action units. Given those action units, we compute action-gons with multiple granularities in separate steps through constrained optimization, and then stack these multi-granularity action-gons together into an overcomplete dictionary. Figure 2 shows an overview of our pipeline for action recognition based on action-gons. Before delving into further details, let us formally define a few terms that we frequently use in the following sections.

Terminology Definition An action unit is defined as one trajectory cluster, which is a basic motion or shape primitive. Let V be a set of action units (graph nodes, see Fig. 1), and E be the set of edges connecting all pairs of action units. An action-gon is defined as $G = \{(V, E); h\}$, where h is a latent variable that chooses a subset of trajectory clusters to be the action-gon nodes. We define the granularity of an action-gon to be $G_{gran} = |V|$. Let us further define B_λ as a dictionary which contains a collection of action-gons with the same granularity λ, and define $B = B_1 \bigcup B_2 \bigcup ... \bigcup B_\lambda, ...$ as a larger dictionary that contains action-gons with all different granularities. To better localize such action-gons in each action video ϑ, we define a linear classifier f as a filter to localize action-gon G in ϑ, and the filter response $\rho = f(G * \vartheta)$ reflects whether G exists in ϑ. Throughout this paper, we call SPM features built on top of the descriptors constructed along with the trajectories [1] as the low-level representation (also called the first-layer in [17]), and the action-gon filter responses as the mid-level representation.

3.2   Action-Gon Modeling

Action Units For each video, we first extract dense trajectories based on [19] (which is an improved version of [1]). Then we apply a hierarchical clustering procedure [20] to group the trajectories into a hierarchical cluster tree solely based on the trajectory geometry features, specifically, a trajectory is represented as a sequence of pixel locations in the spatio-temporal volume as follows, $(x_t, y_t; ...; x_{t+L}, y_{t+L};)$, where t is the starting frame of the trajectory and L is the number of frames along the trajectory. To increase the capability of describing actions with varying scales, we consider clusters at all different levels within the hierarchical cluster tree as potential graph nodes. A higher-level cluster occupies a larger spatial and temporal volume and overlaps with lower-level clusters. In practice, graphs built on hierarchical clusters give rise to higher recognition performance than the single-level clusters used in [2]. We name each cluster as one action unit.

For each trajectory, we extract five low-level features, including trajectory geometry, HOG, HOF, and MBH (Motion Boundary Histograms) along both x
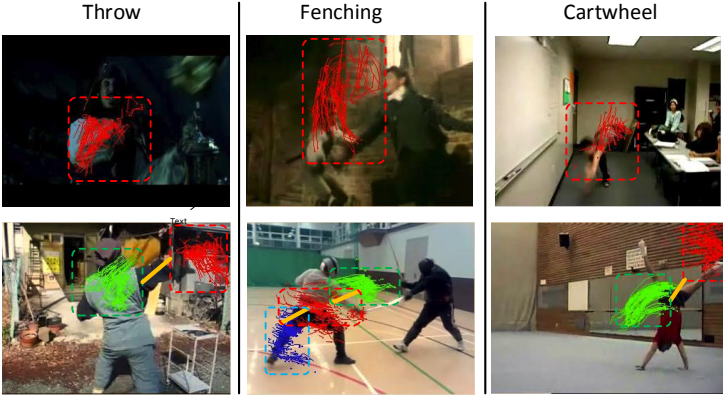
Fig. 3. Examples showing the inferred latent graphs. Note that, even for the same action class, it is still much desired to use multi-granularity graphs to tackle intra-class variations.

and y image axes [1], and build their individual dictionaries through K-means. These low-level features will be used for training and detecting action-gons. We apply localized soft assignment quantization (LSAQ) [21] to code all low-level features. Given any trajectory t_j within a cluster U_i, we use F_{t_j} to denote the concatenated codes of all low-level features along t_j, and 'max' to denote the element-wise maximum operation over a set of vectors. Then the node(trajectory cluster) feature, Γ_{U_i}, of U_i is defined as the max-pooling result over all trajectories within U_i, then

$$\Gamma_{U_i} = \max_{t_j \in U_i} F_{t_j}. \tag{1}$$

Edges and Features The edge feature for a pair of nodes within the graph describes the relative motion and location between the nodes. We first define the edge features for a pair of trajectories t_i and t_j. Suppose a trajectory t_i spans a time interval from T_s^i to T_e^i (s, e are the frame indices). Let us denote the centroid of the trajectory as (x_m^i, y_m^i), the average image-space velocity along the trajectory as (v_x^i, v_y^i). Suppose the distribution of every type of attributes (denoted as z) follows its own Gaussian mixture model, $\mathcal{P}_z(X) = \sum_{k=1}^N \pi_k \mathcal{N}(X|\mu_k, \sigma_k)$. We define the features P_z for a type of attributes (z) using the probability values returned by all the Gaussian components in its own Gaussian mixture model. Let us further define $(\boldsymbol{v_1} * \boldsymbol{v_2}^T)(:)$ as the result of unfolding the outer product between $\boldsymbol{v_1}$ and $\boldsymbol{v_2}$ into a row vector. Then the feature vector for the edge between two trajectories is defined as

$$\phi(t_i, t_j) = [\ P_T(|T_s^i - T_s^j|)^T \tag{2}$$
$$P_x(|x_m^i - x_m^j|) * P_y(|y_m^i - y_m^j|)^T(:)$$
$$P_{v_x}(|v_x^i - v_x^j|) * P_{v_y}(|v_y^i - v_y^j|)^T(:)\ ]^T.$$

Finally, the edge feature vector, $\Phi(U_i, U_j)$, for clusters U_i and U_j is simply defined as the average feature vector of all trajectory pairs across the two clusters, namely

$$\Phi(U_i, U_j) = \frac{1}{|U_i||U_j|} \sum_{t_m \in U_i, t_n \in U_j} \phi(t_m, t_n). \tag{3}$$

Note that $P_T, P_x, P_y, P_{v_x}, P_{v_y}$ can all be estimated using the EM algorithm on a set of sampled trajectories. We empirically set N=6 to prevent overly long edge feature vectors. And it turned out to work well in all our experiments.

Latent Graph Representation Given N candidate graph nodes within a video X, let $\mathcal{H}$ be the space of graphs defined over subsets of these candidate nodes. In this paper, every graph is a complete graph with edges connecting all possible pairs of nodes. Therefore, the configuration of a graph is uniquely determined by the nodes in the graph. Even under this assumption, there are an exponential number of graphs that can be composed by any subset of N nodes, i.e., the number of graphs with $M(M < N)$ nodes is $\binom{N}{M}$. However, only few of them have real discriminative power and can serve as mid-level action-gons. Let h be one of the graph configurations from $\mathcal{H}$. Now that we have defined the node feature as well as edge features, for a graph with fixed M nodes, we define its feature map $\psi(X, h) = [\Gamma_{U_0}, \Gamma_{U_1}, ..., \Gamma_{U_{M-1}}; \Phi(U_0, U_1), ..., \Phi(U_i, U_j), ..., \Phi(U_{M-2}, U_{M-1})]$ Once h is known, $\psi(X, h)$ can be easily established. In reality, however, h is a latent variable that should be inferred, and it is infeasible to perform exhaustive search when both M and N are large. Therefore, we need a more efficient way to infer h during both the training and classification stages, and the inference should be based on the discriminative power of h by learning a supervised classifier. Suppose we already have such a classifier $\mathbf{w}_k$ in the form of a linear SVM, then latent graph h can be inferred by the following operation,

$$h^* = \arg\max_{h \in \mathcal{H}} \mathbf{w}_k^T \psi(X, h). \tag{4}$$

The above optimization is essentially the NP-hard Quadratic Integer Programming problem, and exact inference could be computationally demanding. We therefore adopt the TRW-S method [22] to obtain an approximate solution. Fig. 3 shows examples of inferred latent graphs.

3.3   Learning Multi-Granularity Action-Gons

As indicated by earlier discussion and Equation (4), we use a classifier (also called a filter) to identify the most discriminative latent graph with a predefined granularity (number of nodes) in a video or action class. However, a single classifier has limited generalization capability while actions typically have large intra-class variations due to different viewpoints and scales among other factors. Therefore, we propose to harvest intrinsic mid-level actions for every action class by training multiple latent graph configurations with a predefined granularity in

that class. The collection of classifiers for all classes are included in a mid-level dictionary, named action-gon dictionary. To learn the classifier for each action-gon, we train one-versus-all binary SVM classifiers by taking all the other classes as negative examples.

Suppose we are given a set of videos with binary class labels $S = \{(X_i, y_i)_{i=1}^n\}$, where $y_i \in \{1, -1\}$. Let $\mathcal{H}_i$ be the latent graph space defined over the trajectory clusters in X_i. To make the filters discriminative, we require each positive video X_i^+ contain at least one latent structure $h_i \in \mathcal{H}_i$ that can be identified by one of the filters $w_k, k \in \mathcal{I} = \{1, 2, ..., K\}$, while each negative video X_i^- should not contain any latent structure that can be identified by any of the classifiers. Based on these requirements, we learn the graph classifiers by solving the following optimization problem,

$$\min_{\mathbf{w}_k, \xi_i \geqslant 0, \zeta_i \geqslant 0} \frac{1}{2} \sum_{k=1}^K \|\mathbf{w}_k\|^2 + C_1 \sum_{i=1}^N \xi_i + C_2 \sum_{y_i = +1} \zeta_i \tag{5}$$

s.t.

$$\forall\, i = 1, 2, \cdots, n : y_i \max_{h \in \mathcal{H}_i} \max_{k \in \mathcal{I}} \mathbf{w}_k^{\mathrm{T}} \psi(X_i, h) \geqslant 1 - \xi_i,$$

$$\forall\, y_i = +1, k \in \mathcal{I} : \max_{h \in \mathcal{H}_i} \left[\mathbf{w}_k^{\mathrm{T}} \psi(X_i, h) - \frac{1}{K-1} \sum_{k' \neq k} \mathbf{w}_{k'}^{\mathrm{T}} \psi(X_i, h) \right] \geqslant 1 - \zeta_i,$$

$$\forall\, k_1, k_2 \in \mathcal{I} : \left| \sum_{y_i = +1} \frac{1}{J_i} \sum_{h \in \mathcal{H}_i} (\mathbf{w}_{k_1}^{\mathrm{T}} \psi(X_i, h) - \mathbf{w}_{k_2}^{\mathrm{T}} \psi(X_i, h)) \right| \leqslant \eta,$$

where J_i represents the number of potential graphs within the latent space $\mathcal{H}_i$. The first set of constraints enforce a multiple-instance-based margin for each bag (video) X_i; the second set of constraints try to maintain diversity among the filters, making different filters generate strong responses on different latent structures; and the last set of constraints enforce a balance among filters to avoid a trivial solution that assigns most latent structures to the same filter. Note that the above formulation can be viewed as a generalization of the learning method in [5] because we can treat a VOI (volume of interest) as a single-node graph. For more general cases, we need to infer the hidden graphs by solving an MRF labeling problem. Specifically, we use TRW-S [22] to identify graphs with maximum responses from the latent space $\mathcal{H}_i$, as shown in Equation (4). The entire training process is solved by the Convex-Concave Cutting Plane (CCCP) algorithm [23], which alternates the following two steps, inferring latent structures with Equation (4) and solving a structured SVM problem based on the cutting plane method.

In comparison with our simultaneous action-gon training, [2] only learns one single filter, so its learning algorithm does not impose the second and third set of constraints defined in our optimization Equ.5. Another significant difference is that our method considers all the learned filters as codewords in a mid-level dictionary upon which a higher-level video representation is built for final classification while [2] directly takes the learned filter as the final video classifier.

Since different actions may exhibit different levels of co-occurrence and interaction among regions of interest, the number of co-occurring or interacting regions may vary. Hence, it is obviously suboptimal to use a single graph granularity across all action classes. This further inspired us to adaptively use graph configurations with different granularities during both the learning and classification stages. Ideally, our method is capable of supporting graphs with an arbitrary number of nodes. However, in practice, we restrict the granularity of an action-gon to be 1, 2 or 3 to achieve a better tradeoff between accuracy and computational cost. To build a multi-granularity dictionary, we run dictionary learning via the optimization in (5) for each supported graph granularity separately. All the filters learned from these separate runs are stacked together to form the action-gon dictionary. To the best of our knowledge, this is the first time to build a multi-granularity dictionary of structured elements.

4   Video Representation via Action-Gons

Suppose we have obtained an action-gon dictionary, which contains **g** classifiers for multi-granularity graphs. Given an input video X, we divide it into P spatiotemporal pyramid volumes as defined in [1]. We perform latent structure inference for every volume in the pyramid by taking all the trajectory clusters in the volume as input and estimating Equation (4). This results in a confidence map, where each entry is the estimated confidence that there exists a corresponding action-gon inside the considered volume. Specifically, for the l-th volume, we obtain such a vector $F_l \in R^g$, and $F_l = [\alpha_1^l, ..., \alpha_k^l, ..., \alpha_g^l]$, where $\alpha_k^l = \max_{h \in \mathcal{H}_l} \mathbf{w}_k^T \psi(X, h)$, w_k is the k-th filter within the dictionary, and $\mathcal{H}_l$ is the latent graph space defined over the trajectory clusters in the l-th volume. The final video representation Θ_i is obtained by simply concatenating the confidence maps for all the spatio-temporal pyramid volumes. That means $\Theta_i = [F_1, F_2, \ldots, F_P]$. As most of the previous methods, i.e. [17], a linear SVM classifier built on Θ_i performs the final video category classification.

5   Experiments

In this section, we perform detailed evaluation of the discriminative power of our proposed action-gon representation on three popular and challenging action datasets: HMDB51[8], Youtube[9] and UCF Sports [10].

5.1   Experimental Setup

For all the datasets used in our experiments, we extract refined dense trajectories as in [19] and compute low-level feature descriptors (i.e. Trajectory, HOG, HOF and MBH) with exactly the same parameters given in [1]. As we use bags of words built on top of these low-level features, we train a codebook with K-means for each type of low-level descriptors using 100,000 randomly sampled features, and

set the codebook size to 4000. The parameter setting of our LSAQ coding is identical to that defined in [21] (i.e., $\beta = 10, n = 5$). We adaptively select the threshold for pruning background trajectories to make sure we can collect at least 100 clusters from each video.

For each dataset, we learn a relatively large pool of action-gons using E-quation (5). Instead of learning one set of action-gons using five concatenated low-level descriptors [1], we learn five sets of action-gons. Each set is based on one of the five descriptors. As most of previous low-level dictionary learn-ing, at present we empirically determine the size of the action-gon dictionary. Specifically, we learn action-gons with $K = 3$ (used in Eq. (5)) different granu-larities (i.e., one-gon,two-gon, three-gon). This results in a large dictionary with 5*51*(3+3+3)=2295 action-gons for the HMDB51 dataset. Likewise, we learn 1000 action-gons respectively for the Youtube and UCF Sports datasets. The problem of learning an action-gon dictionary with an optimal size requires much further investigation and, therefore, is left as future work.

Most of the other parameters in our method can take values from a relatively large range without affecting the final performance significantly. We empirically set each of them to a constant value within its working range. For example, we set $C_1 = 256$ and $C_2 = 32$ in Eq. (5) in all our experiments. We follow the same parameter settings proposed in [17] when learning final action classifiers.

Our method has been implemented primarily using Matlab except TRW-S [22][1]. We use the SVM^{struct} package[2] to perform the optimization in Eq. (5). On a modern PC, the training stage of our method spends about 24 hours on the HMDB51 dataset (6776 video clips). Nevertheless, it only takes less than 5 seconds for our trained model to classify a test video clip.

5.2   Performance Evaluation

We first evaluate the overall performance of the action-gon representation through extensive comparisons against existing methods in the action recognition litera-ture. For the HMDB51 dataset, we follow the default splitting rule to perform three rounds of training and testing, and report the average per-class classifica-tion accuracy. When only the mid-level action-gon representation is used, our results in the three rounds are 57.8%, 57.8%, and 58.4%, respectively, resulting in an average performance of 58.0%. As shown in Table 1, compared with other mid-level representations, action-gons have achieved significantly better results, i.e., its result is respectively 7.3% and 15.9% higher than those in [17](50.7%, the second layer performance) and [4](42.1%). The low-level representation in our current implementation is based on the descriptors in [19] but with one major difference, which is the replacement of fisher vectors with bag of words in LSAQ coding. Hence the average per-class classification accuracy achieved with our low-level representation is 57.0%, which is slightly lower than the best

---

[1] Code is available from: http://research.microsoft.com/en-us/downloads/dad6c31e-2c04-471f-b724-ded18bf70fe3

[2] Code is based on http://www.cs.cornell.edu/people/tj/svm_light/svm_struct.html

performance (57.2%) reported in [19]. Nevertheless, our proposed middle-level representation alone outperforms the high-dimensional low-level features used in [19]. In addition, when combined with our low-level features, the average performance of our method can be further elevated to 58.9%, which indicates that our action-gon representation has a strong discrimination power complementary to low-level features, as shown in Figure 5.

HMDB51[8] (.%)		Youtube[9] (.%)	
Oneata et al.[24]	54.8	Le et al.[12]	75.8
Motionlet[4]	42.1	MIL-Bof[25]	80.4
Shi et al[26]	47.6	Dense Traj.[1]	85.4
Jian et al.[27]	52.1	Brendel et al.[28]	77.8
Zhu et al.[17]	54.0	Zhu et al.[17]	89.4
Dense Traj.[19]	57.2	Oneata et al.[24]	89.0
Action-Gons	58.0	**Action-Gons**	89.7
Action-Gons+Low	**58.9**	**Action-Gons+Low**	**92.1**

Table 1. Performance of the proposed action-gon representation and comparisons with state-of-the-art methods. We report average per-class classification accuracy on the datasets. '*mid*' represents action-gons alone, '*mid + low*' represents the combination of action-gons and low-level features.

Methods	Splitting Rule in [10]	Leave-One-Out in [1]
Lan et al.[10]	73.1%	n/a
Raptis et al.[2]	79.4%	n/a
Kovashka et al.[29]	n/a	87.3%
DenseTraj.[1]	n/a	89.1%
Wu et al.[30]	n/a	92.5%
ActionBank[3]	n/a	95.0%
Action-Gons	**83.0%**	**100%**

Table 2. Performance comparison on the UCF-Sports dataset based on two common dataset splitting rules. Note that the middle level features facilitated by Action-Gon generates the state-of-the-art performance under both data splitting rules.

In the experiments on the UCF-Sports dataset, we apply the same setting recently proposed in [10]. As shown in Table 2, the average per-class classification accuracy achieved with our mid-level action-gon representation is 83%, which is significantly higher than the state-of-the-art result (79.4%) [2] among all existing work that adopts the same data splitting rule as in [10]. To fully validate the performance of action-gons, we further apply "leave-one-out" data splitting as in [1], and see that action-gons achieve a 100% classification accuracy, which is a significant improvement over the best "leave-one-out" result reported in [1] (89.1%). Such a large performance gain is primarily achieved with the adaptive action unit localization ability enabled by action-gons. Refer to [10] for reasons why "leave-one-out" splitting generates better classification accuracy. Figure 4 shows the per-class classification accuracy obtained with the action-gons alone.
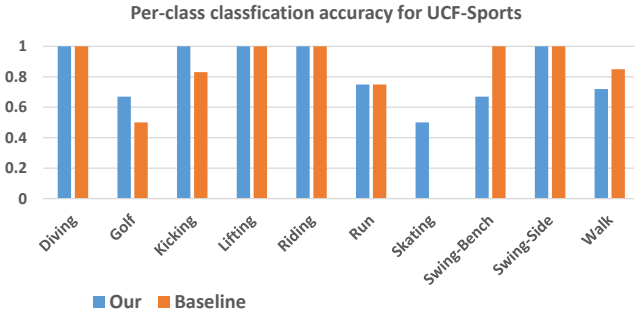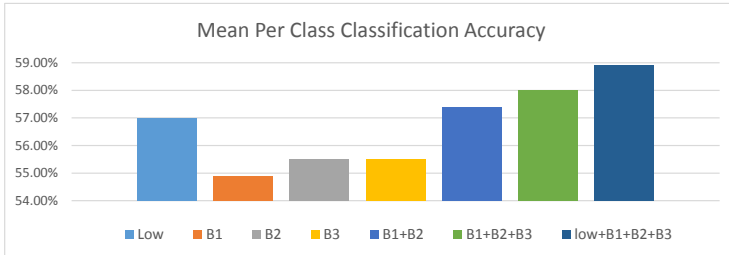
Fig. 4. Per-class classification accuracy on the UCF-Sports dataset according to the dataset splitting rule proposed in Lan et al. [10]

We have also observed advantages of using action-gons on the Youtube dataset. As shown in Table 1, by using the action-gon representation alone, we can achieve an average per-class accuracy of 89.7%, which is superior to any existing methods that rely on low-level features alone according to [19, 24]. When action-gons are combined with low-level features, the performance can be further improved to 92.1%, indicating the two types of feature representations are complementary.

5.3   Analysis and Discussion

Analysis of Multi-Granularity Action-Gons To validate the effectiveness of multi-granularity action-gons, we compare classification performance achieved using various combinations of the action-gon dictionaries B_1, B_2 and B_3. Such combinations include any individual dictionary of the three or any group formed by two or more of these individual dictionaries, such as $B_1 + B_2$ and $B_1 + B_2 + B_3$. The testing results are shown in Figure 5, where we can see that with increasing levels of granularity, we achieve increasing classification accuracy on HMD-B51. This indicates that different action-gon granularity can capture different complexity of interactions (see Figure 3), and therefore provide complementary feature representations. Interestingly, we notice that one-gon dictionary (B_1) achieve slightly worse performance than two-gon dictionary (B_2) and three-gon dictionary (B_3) on HMDB51. However, when combined together, they can increase representation diversity and boost the overall performance.

Analysis of the Action-Gons Size The size of Action-Gon Dictionary is actually controlled by the parameter K (used in Eq. (5)). If K=3, we would have 2295 action gons. We evaluated the performances with respect to different K, take HMDB51 as an example, when K=1,2,3,4, their corresponding accuracy are 55.3%, 57.6%, 57.8% and 55.8% respectively. When K is larger, the Action-Gon

Fig. 5. Average per-class classification accuracy on HMDB51 using action-gons with increasing levels of granularity.

|	B_1	B_2	B_3	B_1,B_2	B_1,B_2,B_3
ITF [19]	54.9	55.5	55.5	57.4	58.0
DTF [1]	52.5	52.1	52.3	53.6	54.1

Table 3. Performance correlation between mid-level and low-level representations. Performance is measured on HMDB51. As ITF [19] improves over DTF [1], we found the performance of their corresponding action-gons improves as well.

based video representation would be very high dimensional, which increases the risk of over fitting. So empirically we set K=3.

Performance Correlation with Low-Level Features The process of learning an action-gon dictionary represents a general pipeline for building a mid-level action representation. Similar to deep learning, an action-gon learns the compositions and abstractions of low-level features. Because of this, although being complementary to low-level features, the discrimination power of action-gons is also expected to be correlated with that of the low-level features. That is, the better the low-level features, the stronger discrimination power the learned action-gons would be equipped. To demonstrate this performance correlation, we have compared the effect of the refined trajectories [19] against that of the baseline trajectories [1] on the quality of the action-gons using the HMDB51 dataset. Given the results in Table 3, we can observe that using more powerful low-level features also boosts the classification performance of the action-gons.

6   Conclusions

In this paper, we have presented a novel approach for middle level action representation and use graphs with varying granularity to serve as the mid-level dictionary which we call action-gon dictionary. Action-gons have been proven to have strong discrimination power in action classification on several popular yet challenging datasets. Extensive experiments and comparisons show that action-gons are complementary to low-level features, indicating that using more
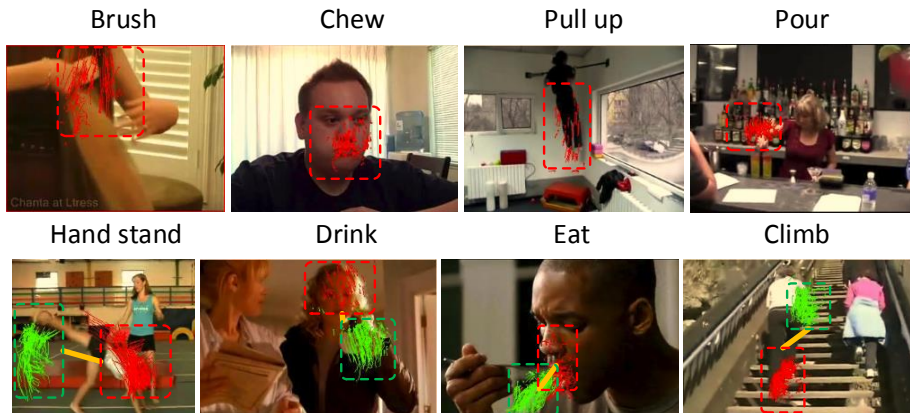
Fig. 6. Inferred action-gons for typical videos. Note that, for simple actions, we may only need a one-gon model while, for relatively complex actions, we need higher-order action-gons.

powerful low-level feature descriptors would boost the performance of action-gons at the same time. Although we have observed connections between our mid-level action-gons and other high-level representations [3], a thorough investigation on questions like "what are the optimal granularities of action-gons" and "how many action-gons should be chosen" are very much desired. In addition, it is worthwhile to explore layered representations to further improve performance correlation between action-gons and low-level features.

7   Acknowledgment

References

1. Wang, H., Klser, A., Schmid, C., Liu, C.L.: Dense trajectories and motion boundary descriptors for action recognition. IJCV (2013)
2. Raptis, M., Kokkinos, I., Soatto, S.: Discovering discriminative action parts from mid-level video representations. In: CVPR. (2012)
3. Sadanand, S., Corso, J.: Action bank: A high-level representation of activity in video. In: CVPR. (2012)
4. Wang, L., Qiao, Y., Tang, X.: Motionlets: Mid-level 3d parts for human motion recognition. CVPR '13 (2013)
5. Gaidon, A., Harchaoui, Z., Schmid, C.: Temporal localization of actions with actoms. IEEE Transactions on Pattern Analysis and Machine Intelligence **35** (2013) 2782–2795

6. Yuan, F., Xia, G.S., Sahbi, H., Prinet, V.: Mid-level features and spatio-temporal context for activity recognition. Pattern Recognition (2012)
7. Julesz, B., Gilbert, E.N., Victor, J.D.: Visual discrimination of texture with identical third-order statistics. Biol. Cybernetics **31** (1978) 137–140
8. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: ICCV. (2011)
9. Liu, J., Luo, J., Shah, M.: Recognizing realistic actions from videos in the wild. In: ICCV. (2009)
10. Lan, T., Wang, Y., Mori, G.: Discriminative figure-centric models for joint action localization and recognition. In: Computer Vision (ICCV), 2011 IEEE International Conference on. (2011) 2003–2010
11. Laptev, I.: On space-time interest points. International Journal of Computer Vision (2005)
12. Le, Q., Zou, W., Yeung, S., Ng, A.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: CVPR. (2011)
13. Klaser, A., Marszalek, M., Schmid, C.: A spatio-temporal descriptor based on 3d gradients. In: BMVC. (2008)
14. Bilinski, P., Bremond, F.: Contextual statistics of space-time ordered features for human action recognition. In: Proceedings of the 2012 IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance. AVSS '12 (2012) 228–233
15. Matikainen, P., Hebert, M., Sukthankar, R.: Representing pairwise spatial and temporal relations for action recognition. In: Proceedings of the 11th European Conference on Computer Vision: Part I. ECCV'10 (2010) 508–521
16. Sun, J., Wu, X., Yan, S., Cheong, L.F., Chua, T.S., Li, J.: Hierarchical spatio-temporal context modeling for action recognition. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. (2009) 2004–2011
17. Zhu, J., Wang, B., Yang, X., Zhang, W., Zhuowen, T.: Action recognition with actons. In: ICCV. (2013)
18. Si, Z., Pei, M., Yao, Z., Zhu, S.C.: Unsupervised learning of event and-or grammar and semantics from video. ICCV (2011)
19. Wang, H., Schmid, C.: Action Recognition with Improved Trajectories. In: International Conference on Computer Vision, Sydney, Australia (2013)
20. Tabatabaei, S.S., Coates, M., Rabbat, M.G.: Ganc: Greedy agglomerative normalized cut. CoRR **abs/1105.0974** (2011)
21. Liu, L., Wang, L., Liu, X.: In defense of soft-assignment coding. In: ICCV. (2011)
22. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Trans. Pattern Anal. Mach. Intell. **28** (2006) 1568–1583
23. Yuille, A., Rangarajan, A.: The concave-convex procedure (cccp). Neural Computation (2003)
24. Oneata, D., Verbeek, J., Schmid, C.: Action and Event Recognition with Fisher Vectors on a Compact Feature Set. In: IEEE Intenational Conference on Computer Vision (ICCV), Sydney, Australia (2013)
25. Michael Sapienza, F.C., Torr, P.H.: Learning discriminative space-time actions from weakly labelled videos. In: BMVC. (2012)
26. Shi, F., Petriu, E., Laganiere, R.: Sampling strategies for real-time action recognition. CVPR '13 (2013)
27. Jain, M., Jegou, H., Bouthemy, P.: Better exploiting motion for better action recognition. CVPR '13 (2013)

28. Brendel, W., Todorovic, S.: Activities as time series of human postures. In: ECCV. (2010)
29. Kovashka, A., Grauman, K.: Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. In: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. (2010) 2046–2053
30. Wu, X., Xu, D., Duan, L., Luo, J., Jia, Y.: Action recognition using multilevel features and latent structural svm. Circuits and Systems for Video Technology, IEEE Transactions on **23** (2013) 1422–1431