Directed Acyclic Graph Kernels for Action Recognition

Ling Wang
Institut Mines-Télécom; Télécom ParisTech; CNRS LTCI
46 rue Barrault, 75013 Paris, France
ling.wang@telecom-paristech.fr

Hichem Sahbi
CNRS LTCI; Télécom ParisTech
46 rue Barrault, 75013 Paris, France
hichem.sahbi@telecom-paristech.fr

Abstract

One of the trends of action recognition consists in extracting and comparing mid-level features which encode visual and motion aspects of objects into scenes. However, when scenes contain high-level semantic actions with many interacting parts, these mid-level features are not sufficient to capture high level structures as well as high order causal relationships between moving objects resulting into a clear drop in performances.

In this paper, we address this issue and we propose an alternative action recognition method based on a novel graph kernel. In the main contributions of this work, we first describe actions in videos using directed acyclic graphs (DAGs), that naturally encode pairwise interactions between moving object parts, and then we compare these DAGs by analyzing the spectrum of their sub-patterns that capture complex higher order interactions. This extraction and comparison process is computationally tractable, resulting from the acyclic property of DAGs, and it also defines a positive semi-definite kernel. When plugging the latter into support vector machines, we obtain an action recognition algorithm that overtakes related work, including graph-based methods, on a standard evaluation dataset.

1. Introduction

Human action recognition is one of the major tasks in multimedia content analysis. It comprises a broad range of applications including video surveillance, robotic vision, video search and human-machine interaction. This task is also very challenging due to the complexity of actions into scenes and their large intra-class variability.

Efforts have been undertaken during the last two decades in order to build models both for action representation and classification. Many existing solutions successfully apply under specific conditions (including static background, single person and periodic actions). However, more powerful algorithms are needed to deal with videos in complex uncontrolled environment (such as multiple views, moving platforms, and cluttered backgrounds) and to handle the exponential growth of video collections in the current multimedia supports including social networks. In this work, we address these issues for complex actions including non-periodic ones and involving multiple persons; for that purpose, recognition is known to be difficult as it requires observing and modeling both local action parts as well as their contextual relationships in order to correctly capture their intra-class variability.

1.1. Related Work

Among the representative action recognition methods (see for instance [18, 1]), a considerable part of them focus attention on utilizing local features. Early methods, even though relatively successful, neglected structural aspects and contextual relationships into video scenes. Many of them were applied under the bag-of-words scheme and relied either on sparse or dense sampling (see for instance [21, 14, 26, 25]). Subsequent works consider structural and spatio-temporal relationships in order to upgrade local features [11, 16]; in these works, structural information between feature points has been used both at local and global scales. Among existing methods, bottom-up ones consider relationships in local neighborhoods in order to build higher order features ([17, 11, 15, 7]). The latter summarize content as well as context from groups of neighboring features and result into more descriptive features for recognition compared to the initial ones. These methods, relying also on the bag-of-words scheme during learning, still suffer from their inability to sufficiently handle structural information into scenes. Other existing methods proceed in a top-down way, by applying graph clustering on dense trajectories in order to build mid level components which again are described using bag-of-words [27, 5, 19].

More recent works learn from global structures in order to achieve action recognition. For instance authors in [3, 19] use graphical models in order to capture dependency between different components; these models are used under the assumption that actions, belonging to the same category, share common structures and fixed number of parts. Kernel methods are also used in order to compare videos while handling global structures; In [4], videos are described by arranging frame representations into matrices and kernel values are computed based on auto-correlation distances between these matrices. In the work of [5], complex actions are considered as decomposed spatio-temporal parts and a binary tree is built on video data. A kernel method, based on the convolution of all subtree patterns, is used in order to compare videos. Another work [27], considers components as human body parts, and combines them with context-dependent kernels in order to capture structural and causal relationships around components. This method uses a convolution kernel between components and iteratively learns a more effective kernel matrix in order to achieve action recognition. In the work of [23], authors describe videos as spatio-temporal graphs, with nodes corresponding to individual elements (e.g. single trajectories) and graph links corresponding to both directed and undirected relations. In the learning phase, authors use tensor product graph in order to infer their graphical model.

1.2. Motivation and Contribution

Among action recognition techniques those based on graph comparison, using kernel machines, are particularly interesting but their success is very dependent on the used kernels [5]. The kernels defined as symmetric and positive semi-definite functions, should reserve high values only if two given actions are very similar. Considering graph-based kernels, two families of these kernels can be found in the literature. In the first family, similarity between two given graphs is defined by matching the underlying sub-patterns (nodes, edges, random walks, etc.). Random walk graph kernels [6, 24] belong to this family and have been deeply studied theoretically and successfully applied mainly in bioinformatics and chemistry data where nodes and edges have simple labels. The general principle of these kernels considers that similarity between two given graphs should be proportional to the number of common paths modeled with product graphs. Beside the computational overhead of these kernels, their convergence is not always guaranteed for general graph structures and their performances are highly dependent on the relevance of labels in graphs.

The second family of graph-based kernels proceeds differently and considers similarity between two given graphs as a decreasing function of the distance between first or high order statistics of their sub-patterns. This family of methods includes graphlets [22, 12] which can capture more complex sub-patterns compared to simple path patterns, but their application to general graph structures is limited only to small orders (usually less than 5 nodes) and this is due to combinatorial issues. Note that kernels, belonging to this second family, usually guarantee the positive definiteness by construction while this property is not straightforward and not always guaranteed on the first family of kernels.

In this work, we introduce a novel graph kernel that attempts to overcome the limitations of these two aforementioned families of kernels while keeping their strengths for action recognition problem. In our proposed solution, we use a particular graph structure, known as directed acyclic graphs in order to model spatio-temporal relationships between action components. In this representation, a given video is described with a DAG, where its nodes correspond to the mid-level action components and links characterize spatial as well as temporal relationships between them. Then, we use these DAGs, in order to compare videos by defining a novel graph-based kernel function. Note that the acyclic property of DAGs, allows us to guarantee important theoretical properties as well as convergence while making the proposed kernel computationally efficient and also effective.

Finally, our model is built upon mid-level features resulting from grouping local action elements. The latter correspond to spatio-temporal local features, short trajectories, etc. [14, 25]. All these local features are widely used due to their robustness to noise, partial occlusion, camera jittering, change of illumination, etc. Considering these advantages, we follow up the line in [19, 27] in order to extract these mid-level features by grouping trajectories into components resulting into semantically meaningful moving body parts in videos.

The rest of this paper is organized as follows. We propose our methodology in the three following sections: In section 2 we present details about DAG construction from video data. In section 3, we introduce our directed acyclic graph kernel based on walk patterns. We show and discuss experimental results in section 4 and we conclude in section 5.

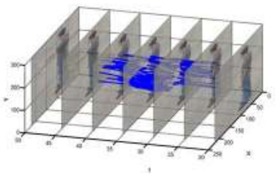
2. Our Directed Acyclic Graph Construction

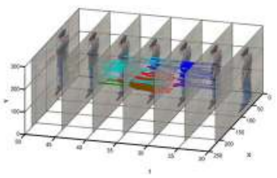
In this section, we present our video description method based on DAGs. We first extract dense trajectories [25], then we group them using agglomerative clustering in order to build mid-level feature components. We tackle in this section several issues including the automatic selection of the number of mid-level components into a given video as well as the modeling of spatio-temporal relationships between components using DAGs.

2.1. From Dense Trajectories to Mid-level Components

Following the line in [25], trajectories are obtained by tracking densely sampled points in successive video frames using optical flow and each trajectory is described by features extracted from local space-time volume around it. All

(a) Frame example


(b) Dense trajectories


(c) Trajectory clusters

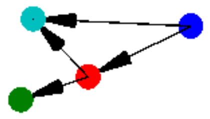

(d) Action component graph

Figure 1. This figure shows our DAG construction process: (a) given a video clip which contains an action; (b) extract dense trajectories in the spatio-temporal domain; (c) cluster trajectories into components and each of the components corresponds to a moving body part; (d) construct a directed acyclic graph on these components.

of these trajectories are limited to a small fixed length in order to alleviate trajectory drifting and to obtain homogeneous components in the subsequent steps.

Considering a finite collection of trajectories extracted on a given video, we build an adjacency graph where each node corresponds to a trajectory and an edge exists between two nodes iff the underlying trajectories are neighbors in a particular feature space; in that feature space, a trajectory with L points, is described by its normalized 3D space-time coordinates and velocities as well as its appearance and motion features (described with HOG, HOF and MBH respectively [25]).

Using this adjacency graph, we cluster trajectories into components with a graph-based agglomerative method proposed in [28]. This clustering method allows us to build components by hierarchically merging similar trajectories in that graph (see Figs. 1 and 2). The choice of this particular graph-based clustering algorithm was mainly driven by its ability to achieve clustering inside the manifold enclosing the space of trajectories, and this provides better estimate of distances in contrast to other methods which rely mainly on Euclidean distances. Extra details about this clustering algorithm, out of the main scope of this paper, are deliberately omitted and can be found in [28].

The number of clusters in the above algorithm reflects the number of components[1]. In order to automatically select this number, we use a simple criterion, similar to the one in [9], that balances intra-cluster and inter-cluster variances.

[1]Selecting the number of components of actions in a given video scene is not trivial at least because we have no priori knowledge about how many persons are interacting in that scene (see also [5, 19]).

Considering a given partition of the set of trajectories into K clusters, the intra-cluster variance measures the average distance between trajectories and their K cluster centers while the inter-cluster variance measures the average distance between the K cluster centers and the global center. Using these measures, we incrementally cluster trajectories for different and decreasing values of K and we keep the value of K that makes the difference, between intra and inter-cluster variances, the smallest.

2.2. DAG Construction

Components obtained from trajectories only reveal motion of local parts, however, complex actions depend not only on the local motion parts, but also on their relationships. Current literature relies mainly on undirected graph structures in order to model dependencies between action components and use graph-based learning techniques in order to infer action classes. However, these general graph structures are not always appropriate. On the one hand, undirected graph based representations are highly redundant (for instance simple causal relationships, such as a component "precedes"/"follows" another one can equivalently be described with one "relation" only.). On the other hand, and more importantly, general undirected graph structures do not always guarantee some important theoretical properties (for instance, inference based on diffusion on graphs is not always guaranteed to converge when graphs include loops.). Following the previous arguments, we choose directed acyclic graphs, in order overcome the two above limitations.

Given a constellation of components extracted from a given video as shown in section 2.1, we build an adjacency DAG (denoted $\mathcal{G} = (\mathcal{V}, \mathcal{E})$), where each node in $\mathcal{V}$ corresponds to a component (a cluster of trajectories described by its cluster center) and a directed edge, in $\mathcal{E}$, exists between two components v, v' in $\mathcal{V}$ if the following relationship exists:

Spatio-temporal causal relationship. As described in section 2.1, each component corresponds to a cluster of trajectories, and occupies a local volume in the XYT space (see Fig. 1 (c)). Two components, associated to nodes v, v' in $\mathcal{V}$, are considered as neighbors if their clusters join together at the XYT space. In the case v and v' are neighbors but not overlapping in time, we create a link directed from v to v' that explicitly means "v precedes v'" (and also implicitly means that "v' follows v") or vice-versa.

Notice that unlinked nodes in $\mathcal{G}$ may either correspond to action components running independently or to weakly dependent components (i.e., connected through intermediate nodes). This setting guarantees that the obtained graph structure is acyclic. As each node represents a bunch of trajectories in the XYT space, the links capture also rich spatio-temporal relationships. We treat two nodes v and v' overlapping in time but meeting in the XY space as weakly dependent because they lack causal relationship so they are less discriminative for video actions.

Finally, and as will be shown through the next section, walks/paths in this DAG structure have bounded lengths and this is again crucial in order to make the proposed graph-based kernel evaluation process convergent, computationally efficient while still effective.

3. Graph Kernels

Given a collection of video sequences $\mathcal{S} = \{V_i\}_i$, we describe each video V_i in $\mathcal{S}$ using a directed acyclic graph $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$ as shown in section 2. For each node $v \in \mathcal{V}_i$, we extract a feature vector, denoted $\psi(v) \in \mathbb{R}^s$ (see section 2.1), corresponding to the average of feature vectors taken from all trajectories in v. We also define an elementary kernel $k_{\mathbf{e}}$, between nodes in $\cup_i \mathcal{V}_i$ as $k_{\mathbf{e}}(v, v') = \exp(-\|\psi(v) - \psi(v')\|_2^2/(2\sigma))$; here σ is the scale of the gaussian kernel and $\|.\|_2$ is the ℓ_2 norm.

Our goal is to design a graph kernel function which returns similarity between two given graphs $\mathcal{G}_i$, $\mathcal{G}_j$. Afterwards, we plug this kernel, into support vector machines (SVMs), in order to achieve action classification.

In the remainder of this paper, unless explicitly mentioned, we denote a given graph $\mathcal{G}_i$ (resp. $\mathcal{G}_j$) simply as $\mathcal{G}$ (resp. $\mathcal{G}'$).

3.1. Random Walk Graph Kernel

Let's consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of nodes $\mathcal{V} = \{v_1, \dots, v_n\}$ and edges $\mathcal{E} = \{e_1, \dots, e_m\}$ (m, n may vary with graphs in $\{\mathcal{G}_i\}$). A walk of length t in $\mathcal{G}$ is defined as a sequence of nodes $\pi = (v_{k_0}, v_{k_1}, \dots, v_{k_t}) \in \mathcal{V}^{t+1}$, with $k_0, \dots, k_t \in \{1, \dots, |\mathcal{V}|\}$ and $(v_{k_i}, v_{k_{i+1}}) \in \mathcal{E}$.
Random walk kernels [6] compare graphs by counting the number of common walks in these graphs[2]. These kernels measure how similar are two given graphs according to the frequency of their common substructures (i.e., walks). This family of kernels has a solid theoretical background [6, 24] and relatively efficient algorithmic solutions which are also able to handle walks with infinite lengths [24, 10].

Definition 1 (Tensor product graph [6])**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}), \mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ be two graphs. The tensor product of $\mathcal{G}, \mathcal{G}'$ is defined as $\mathcal{G}_\times = (\mathcal{V}_\times, \mathcal{E}_\times)$ with*

$$\begin{aligned} \mathcal{V}_\times &= \{(v, v') : v \in \mathcal{V}, v' \in \mathcal{V}'\} \\ \mathcal{E}_\times &= \{((v, v'), (u, u')) \\ &\quad : (v, u) \in \mathcal{E}, (v', u') \in \mathcal{E}'\}. \end{aligned}$$

[2]A common walk between two graphs $\mathcal{G}, \mathcal{G}'$ corresponds to simultaneous walks in $\mathcal{G}$ and $\mathcal{G}'$, of the same length, that generate the same sequence of node and/or edge labels.

Tensor product of two graphs can be defined using tensor product of matrices; Let $\mathbf{E}, \mathbf{E}'$ be the adjacency matrices of $\mathcal{G}$ and $\mathcal{G}'$ respectively, the adjacency matrix $\mathbf{E}_\times$ of $\mathcal{G}_\times$ can be written as $\mathbf{E}_\times = \mathbf{E} \otimes \mathbf{E}'$, with $\otimes$ being the Kronecker tensor product operator.

Definition 2 (Random walk graph kernel [6])**.** *Let $\mathcal{G}$, $\mathcal{G}'$ be two graphs. For any $T \in \mathbb{N}^+$, a random walk graph kernel between $\mathcal{G}$, $\mathcal{G}'$ is defined as $\mathcal{K}(\mathcal{G}, \mathcal{G}') = \sum_{i,j} \mathbf{K}_{\times ij}$, with $\mathbf{K}_\times = \sum_{t=0}^T g(t)\mathbf{E}_\times^t$, $\mathbf{E}_\times^t = \mathbf{E}_\times^{t-1}\mathbf{E}_\times$, $\mathbf{E}_\times^0 = \mathbf{I}_{p\times p}$ ($p = |\mathcal{V}_\times|$), and $g(t) \geq 0$, $\forall t \in \{0, \dots, T\}$.*

For graphs with node labels in $\mathcal{L} = \{1, 2, \dots, d\}$, the kernel matrix $\mathbf{K}_\times$ can be restricted to walks, with the same node labels, as $\mathbf{K}_\times = \sum_{t=0}^T g(t)(\mathbf{L}_\times\mathbf{E}_\times)^t\mathbf{L}_\times$; here $\mathbf{L}_\times = \sum_{\ell\in\mathcal{L}} \mathbf{L}_\ell \otimes \mathbf{L}_\ell^{'}$ with $\mathbf{L}_\ell$ (resp. $\mathbf{L}_\ell^{'}$) being a diagonal matrix indicating whether nodes in $\mathcal{G}$ (resp. $\mathcal{G}'$) are assigned a label ℓ [10]. In this new definition of $\mathbf{K}_\times$, the matrix $\mathbf{L}_\times$ is used to characterize label consistency between nodes in $\mathcal{G}$, $\mathcal{G}'$. We use this definition and extend the random walk kernel to unlabeled graphs as described below.

3.2. Our Generalized Random Walk Graph Kernel

In this section, we generalize the standard random walk kernel [6, 24] to unlabeled graphs. Instead of labels, we consider an elementary kernel function $k_{\mathbf{e}}$ which provides us with a similarity between nodes in $\cup_i \mathcal{V}_i$. A similar kind of kernel was proposed in [2] based on dynamic programming for fixed and limited length of walk patterns. This work emphasized on a generalization to tree-walk patterns which was applied successfully in image recognition [8]. In our method, by utilizing the tensor product framework, the graph kernel can also handle walk patterns with any (possibly infinite) lengths.

Definition 3 (Generalized random walk graph kernel)**.** *Given two graphs $\mathcal{G}$, $\mathcal{G}'$, we define the generalized random walk graph kernel as $\mathcal{K}(\mathcal{G}, \mathcal{G}') = \sum_{i,j} \mathbf{K}_{\times ij}$ with*

$$\mathbf{K}_\times = \sum_{t=0}^{T} g(t)(\mathbf{W}_\times\mathbf{E}_\times)^t\mathbf{W}_\times, \qquad (1)$$

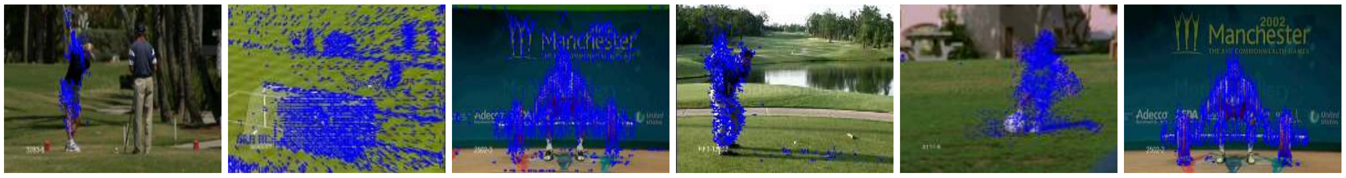
and $\mathbf{W}_\times \in \mathbb{R}^{p\times p}$ being a diagonal matrix with diagonal entries set to $\{k_{\mathbf{e}}(v, v'), \forall (v, v') \in \mathcal{V}_\times\}$ and $(\mathbf{W}_\times\mathbf{E}_\times)^0 = \mathbf{I}_{p\times p}$.

In Eq. (1), $T \ll \infty$ corresponds to walks with finite lengths. A special case is when $T = 0$, so $\mathcal{K}(\mathcal{G}, \mathcal{G}')$ can be rewritten as $g(0)\sum_{i,j} \mathbf{W}_{\times ij} = g(0)\sum_{(v,v')\in\mathcal{V}_\times} k_{\mathbf{e}}(v, v')$ and this corresponds to the convolution kernel on graph nodes.

3.3. Properties

Proposition 1. *Provided that $g(t) \equiv 1$ ($\forall t \in \{0, \dots, T\}$), the generalized random walk graph kernel defined in Def. 3*

(a) Frame examples


(b) Short trajectories

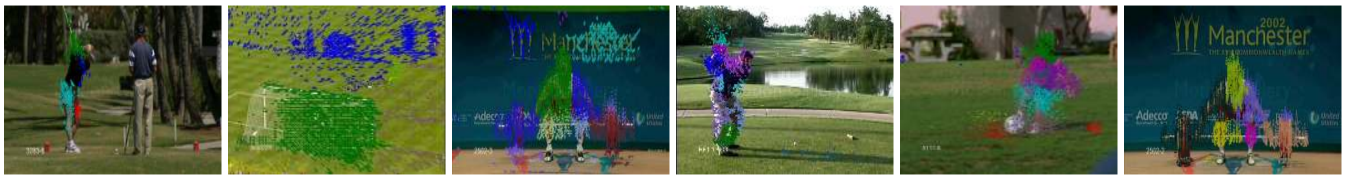

(c) Action components

Figure 2. This figure shows examples of frames, trajectories and action components (using different colors) in different categories taken from training (left three columns) and testing (right three columns) data.

measures similarity between $\mathcal{G}$, $\mathcal{G}'$ by summing up all the similarity between walks of increasing lengths; i.e.,

$$\mathcal{K}(\mathcal{G},\mathcal{G}') = \sum_{t=0}^{T} \sum_{(\pi_t,\pi'_t)\in\mathcal{V}^{t+1}\times\mathcal{V}'^{t+1}} k_{\mathbf{w}}(\pi_t,\pi'_t), \tag{2}$$

with $\pi_t = (v_{k_0}, v_{k_1}, \dots, v_{k_t})$, $\pi'_t = (v'_{l_0}, v'_{l_1}, \dots, v'_{l_t})$, *and* $k_{\mathbf{w}}(\pi_t,\pi'_t) = \prod_{z=0}^{t} k_{\mathbf{e}}(v_{k_z}, v'_{l_z})$.

Proof. We proceed by induction.
Let's denote $(\mathbf{W}_\times\mathbf{E}_\times)^t\mathbf{W}_\times$ simply as $\mathbf{K}_\times^{(t)}$. If is easy to see that the diagonal of $\mathbf{K}_\times^{(0)} = \mathbf{W}_\times$ includes all similarities between nodes (i.e., between walks of length zero). For $t=1$, $\mathbf{K}_{\times\mathbf{ij}}^{(1)} = \mathbf{W}_{\times\mathbf{ii}}\mathbf{E}_{\times\mathbf{ij}}\mathbf{W}_{\times\mathbf{jj}}$ is the similarity of walks, of length one, connecting nodes $\mathbf{i}$ and $\mathbf{j}$ in the product graph $\mathcal{G}_\times$.
Similarly, $\mathbf{K}_{\times\mathbf{ij}}^{(2)} = \sum_{\mathbf{k}} \mathbf{W}_{\times\mathbf{ii}}\mathbf{E}_{\times\mathbf{ik}}\mathbf{W}_{\times\mathbf{kk}}\mathbf{E}_{\times\mathbf{kj}}\mathbf{W}_{\times\mathbf{jj}}$ sums up similarities of all walks of length two starting from node $\mathbf{i}$ and ending at node $\mathbf{j}$. Assuming $\mathbf{K}_{\times\mathbf{ij}}^{(t-1)}$ sums up similarities of all walks of length $t-1$ connecting node $\mathbf{i}$ and node $\mathbf{j}$, then $\mathbf{K}_{\times\mathbf{ij}}^{(t)} = \sum_{\mathbf{k}} \mathbf{K}_{\times\mathbf{ik}}^{(t-1)}\mathbf{E}_{\times\mathbf{kj}}\mathbf{W}_{\times\mathbf{jj}}$ sums up all the similarities of walks of length t from node $\mathbf{i}$ to node $\mathbf{j}$. $\square$

Proposition 2. *Provided that the kernel $k_{\mathbf{e}}$ is positive semi-definite, the generalized random walk kernel $\mathcal{K}$ is also positive semi-definite.*

Proof. From Eq. (2), $\mathcal{K}$ is the sum of products involving $k_{\mathbf{e}}$. When $k_{\mathbf{e}}$ is positive semi-definite, the generalized random walk kernel $\mathcal{K}$ will also be positive semi-definite resulting from the closure of the positive definiteness with respect to the sum and the product of kernels. $\square$

Remarks. Note that when $g(t) = \lambda^t$, and $T \to \infty$ the sum in Eq. (1) reduces to

$$\mathbf{K}_\times = \sum_{t=0}^{\infty} (\lambda\mathbf{W}_\times\mathbf{E}_\times)^t\mathbf{W}_\times = (\mathbf{I} - \lambda\mathbf{W}_\times\mathbf{E}_\times)^{-1}\mathbf{W}_\times. \tag{3}$$

In contrast to undirected graphs[3], applying the generalized random walk graph kernel $\mathcal{K}$ on directed acyclic graphs (i.e., video DAGs constructed as shown in section 2.2) has several advantages:

- For any pair of DAGs $\mathcal{G}, \mathcal{G}'$, and for finite values of $g(t)$, the kernel $\mathcal{K}(\mathcal{G},\mathcal{G}')$ is always finite, and this results from the finiteness of walk lengths.
- The parameter λ can be chosen to be in favor of long (or short) walks without altering convergence. Indeed, walk patterns model dependencies between components, and larger (resp. smaller) values of λ make the

[3]For undirected graphs, due to the existence of loops or tottering, walks may have length up to infinity. In order to get finite kernel values, the parameter λ needs to be carefully chosen. In many problems, a very small $\lambda \ll 1$ (for instance 0.001) is often used in order to guarantee the convergence of the sum in Eq. (1) (when $T \to \infty$) or invertibility of the closed form matrix solution in Eq. (3). For undirected graphs (in contrast to DAGs), this condition highly limits the usefulness of random walk kernels in order to handle long walks and hence long term actions.

generalized random walk kernel $\mathcal{K}$ more suitable for long (resp. short) term actions (see Fig. 5).

4. Experiments

In this section, we evaluate the performance of action classification using the challenging UCF Sport database [20]. This dataset includes 150 video shots taken from various TV channels and sport events with 10 categories of actions. These videos have diverse contents and were taken under extremely challenging and uncontrolled conditions, with many viewpoint changes (see samples of video frames in Fig. 2). Each video sequence is processed in order to extract its underlying directed acyclic graph (as discussed in section 2); the maximum number of nodes in these DAGs is limited to 100 so noisy action components are ignored. Fig. 3 shows examples of these action components taken from different categories.

4.1. Setting & Evaluation Protocol

The purpose of our evaluation is to show the performance of our generalized random walk graph kernel (GRWK) compared to standard random walk kernels as well as other baseline graph kernels. We also extend the comparison of action classification against reported results in related work.

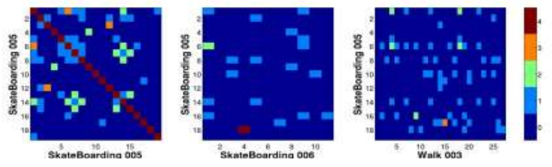
We plugged the generalized random walk kernel into support vector classifiers in order to evaluate their performances. Again, the targeted task is action classification also known as "activity recognition"; given a video shot described with a DAG, the goal is to predict which activity (class) is present into that shot. For this purpose, we trained a "one-versus-all" SVM classifier for each class; we use the train-test split evaluation protocol suggested in [13] in order to alleviate learning from background.

We repeat this training and testing process through different classes and we take the average accuracy over all classes.

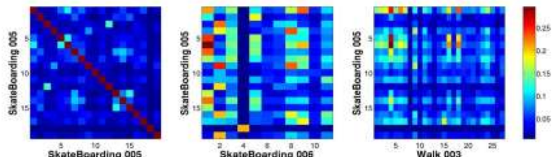
4.2. Performance & Comparison

Baselines. We first show a comparison of action recognition performance, using SVM + GRWK, against two baselines.

– *SVM + Convolution Kernel (CK).* This kernel is defined as $\mathcal{K}(\mathcal{G},\mathcal{G}') = \sum_{(v,v')\in\mathcal{V}\times\mathcal{V}'} k_{\mathbf{e}}(v,v')$. Note that this baseline, applied to mid-level components, is more appropriate, for comparison, than Bag-of-Words as the number of mid-level components is very limited for each video and this makes the underlying BOW histogram statistically not meaningful. This kernel is used in order to show the performances when using only the intrinsic visual features of nodes (components) into videos and without taking into account their relationships. Early observations, reported in Fig. 3, show that high kernel values in $\{k_{\mathbf{e}}(v,v')\}$ do not always correspond to actual matches between components. From these preliminary observations, the convolution kernel is not sufficiently discriminative as also corroborated in Table 1. The results show a clear gain when utilizing the graph structure rather than treating components independently.



(a) Label consistency between nodes



(b) Kernel values between nodes

Figure 4. This figure shows the node kernel matrices for SRWK (top) and GRWK (bottom) between videos belonging to the same (columns 1 and 2) and different action categories (column 3). For each node, we assign four labels based on pregenerated dictionaries. (a) shows the number of consistent labels between pairs of nodes. These examples show that labels for videos belonging to the same category are not always more consistent than labels for videos belonging to different categories.

– *SVM + Standard Random Walk Kernel (SRWK).* Following Def. 2, in order to evaluate this kernel, we first label nodes in $\{\mathcal{G}_i\}$. For that purpose, we build a dictionary of d centroids[4] by clustering features in $\{\psi(v) : v \in \cup_i \mathcal{V}_i\}$ belonging to training data and by assigning nodes in $\{\mathcal{V}_i\}_i$ to centroids that minimize their distances. Early observations in Fig. 4 (a), show that labels are not always consistent through different shots, and this also affects the performance of SVM + SRWK in action classification (again see Table 1).

Notice that for the above two baselines as well as our SVM + GRWK methods, components in videos used for comparison are exactly the same, and they are distributed from human body as well as background.

Comparison w.r.t related work. we also compared classification performance of our method against related work including [13] and [19]. The latter is based on graphical

[4]We generate a dictionary of d words (d is set to 200 in the experiments) for each of the four types of features (Velocity, HOG, HOF and MBH).

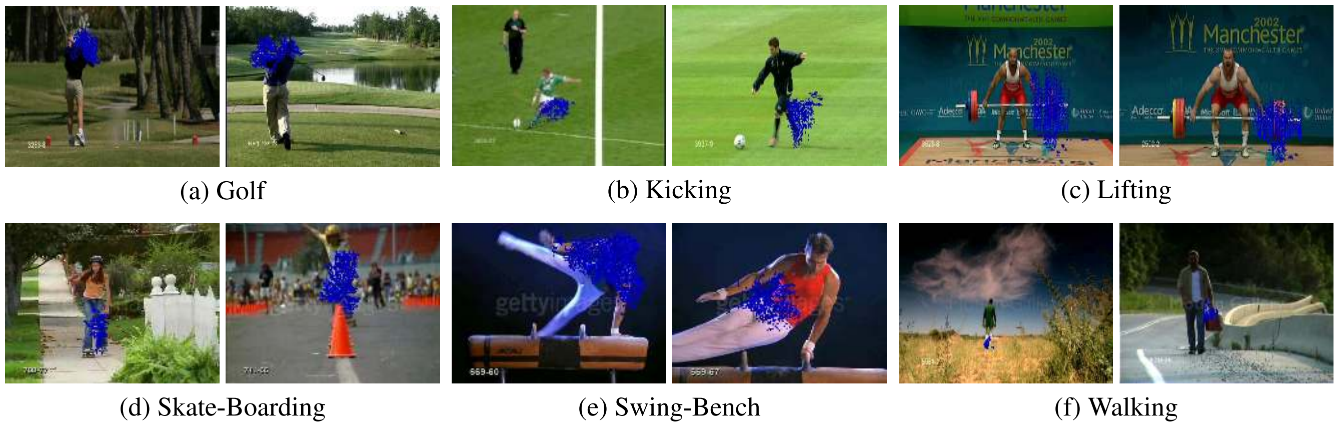


Figure 3. This figure shows example of components taken from different shots and categories. These components correspond to samples of pairs in $\{(v, v')\}$ with the highest kernel values in $\{k_e(v, v')\}$.

	Diving	Golf	Kicking	Lifting	Horse Riding	Running	Skate Boarding	Swing Bench	Swing Side	Walking	Average
#Train video	10	12	14	4	8	9	8	14	9	15	10.3
#Test video	4	6	6	2	4	4	4	6	4	7	4.7
Lan et al.[13]	1	0.5	1	1	1	0.5	0.5	0.67	1	0.14	0.731
Raptis et al.[19]	1	0.5	0.83	1	1	0.75	0	1	1	0.86	0.794
SVM + CK	1	0.33	0.33	1	0.5	0.5	0.5	1	1	0.71	0.688
SVM + SRWK	1	0.5	0.83	1	1	0.25	0	0.67	1	1	0.725
SVM + GRWK	1	0.67	1	1	1	0.5	0.5	1	1	0.86	**0.852**

Table 1. This table shows classification accuracy on the UCF Sport dataset and comparison against baselines and related work. Our experiments do not use any bounding box information in either training or testing process. Results in this table show the effectiveness of walk patterns built on trajectory components. Generalized Random Walk Kernel (GRWK) brings a significant gain over Standard Random Walk Kernel (SRWK) .

models (with three latent variables) and exploits bounding boxes of objects during training. In contrast, our method does not use any bounding box information as the similarity in GRWK concentrates mainly on common components, characterized by common walks in DAGs (that emphasize on the moving objects but not background). From results in Table 1, SVM + GRWK brings a clear gain of at least 7%.

Finally, Fig. 5, shows the evolution of the performance of action recognition (using SVM + GRWK), class by class, with respect to different and increasing values of λ. As already discussed in section 3.2, this parameter λ controls the importance of walk lengths. Following these results, large values of λ are more dedicated to long term actions (including "kicking" and "running") while small values are more suitable for short term actions (such as "diving"). Thus, GRWK is able to distinguish between confusing action classes (such as "running to kick a ball" and "running for jogging".

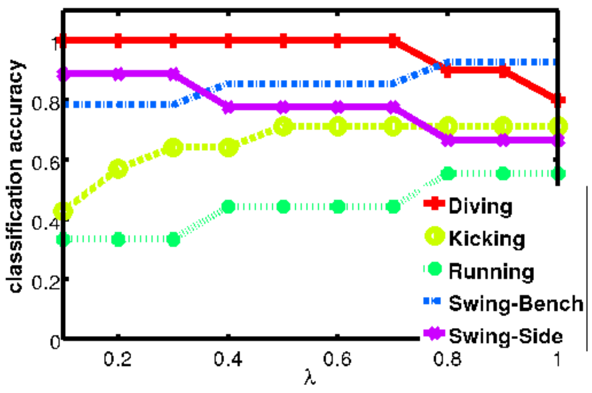


Figure 5. This figure shows the evolution of the accuracy of action recognition, class by class, with respect to the parameter λ in GRWK. For some long term actions (e.g. Kicking, Running), large values of λ increase the classification accuracy while small values are more suitable for short term actions (e.g. Diving).

5. Conclusion

We introduced in this paper a novel action classification method based on a new extension of random walk graph kernels. The strength of this method resides in its ability to (i) extend random walk graph kernels to unlabeled graphs (by making them label insensitive) and also its ability to (ii) exploit the acyclic properties of DAGs in order to guarantee the convergence of GRWK while ensuring its effectiveness. Using a challenging evaluation set, the method was able to exploit spatio-temporal causal relationship between action components in order to precisely characterize moving body parts and their dependencies. Thereby, the method was able to bring a substantial gain, in action recognition performances, when compared to different baselines as well as closely related work.

Acknowledgements

This work was supported in part by a grant from the Research Agency ANR (Agence Nationale de la Recherche) under the MLVIS project and a grant from DIGITEO under the RELIR and VISUNET projects.

References

[1] J. Aggarwal and M. Ryoo. Human Activity Analysis: A Review. *ACM Computing Surveys*, 43(3):16:1–16:43, 2011.

[2] F. R. Bach. Graph Kernels between Point Clouds. In *ICML*, 2008.

[3] W. Brendel and S. Todorovic. Learning Spatiotemporal Graphs of Human Activities. In *ICCV*, 2011.

[4] A. Gaidon, Z. Harchaoui, and C. Schmid. A time series kernel for action recognition. In *BMVC*, 2011.

[5] A. Gaidon, Z. Harchaoui, and C. Schmid. Recognizing activities with cluster-trees of tracklets. In *BMVC*, 2012.

[6] T. Gärtner, P. A. Flach, and S. Wrobel. On Graph Kernels: Hardness Results and Efficient Alternatives. In *Proceedings of the 16th Annual Conference on Computational Learning Theory and the 7th Kernel Workshop*, 2003.

[7] A. Gilbert, J. Illingworth, and R. Bowden. Action Recognition Using Mined Hierarchical Compound Features. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):883–897, 2011.

[8] Z. Harchaoui and F. Bach. Image Classification with Segmentation Graph Kernels. In *CVPR*, 2007.

[9] Y. Jung, H. Park, D.-Z. Du, and B. L. Drake. A Decision Criterion for the Optimal Number of Clusters in Hierarchical Clustering. *J. of Global Optimization*, 25(1):91–111, 2003.

[10] U. Kang, H. Tong, and J. Sun. Fast Random Walk Graph Kernel. In *SDM*, 2012.

[11] A. Kovashka and K. Grauman. Learning a Hierarchy of Discriminative Space-Time Neighborhood Features for Human Action Recognition. In *CVPR*, 2010.

[12] N. Kriege and P. Mutzel. Subgraph Matching Kernels for Attributed Graphs. In *ICML*, 2012.

[13] T. Lan, Y. Wang, and G. Mori. Discriminative Figure-Centric Models for Joint Action Localization and Recognition. In *ICCV*, 2011.

[14] I. Laptev, M. Marszałek, C. Schmid, and B. Rozenfeld. Learning Realistic Human Actions from Movies. In *CVPR*, 2008.

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

[16] J. Liu, Y. Yang, I. Saleemi, and M. Shah. Learning semantic features for action recognition via diffusion maps. *Computer Vision and Image Understanding*, 116(3):361–377, 2012.

[17] J. Liu, Y. Yang, and M. Shah. Learning Semantic Visual Vocabularies Using Diffusion Distance. In *CVPR*, 2009.

[18] R. Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010.

[19] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *CVPR*, 2012.

[20] M. D. Rodriguez, J. Ahmed, and M. Shah. Action MACH: A Spatio-temporal Maximum Average Correlation Height Filter for Action Recognition. In *CVPR*, 2008.

[21] C. Schuldt, I. Laptev, and B. Caputo. Recognizing Human Actions: A Local SVM Approach. In *ICPR*, 2004.

[22] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. Borgwardt. Efficient graphlet kernels for large graph comparison. In *AISTATS*, 2009.

[23] S. Todorovic. Human Activities as Stochastic Kronecker Graphs. In *ECCV*, 2012.

[24] S. V. N. Vishwanathan, N. N. Schraudolph, R. I. Kondor, and K. M. Borgwardt. Graph Kernels. *J. Mach. Learn. Res.*, 11:1201–1242, 2010.

[25] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action Recognition by Dense Trajectories. In *CVPR*, 2011.

[26] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009.

[27] F. Yuan, G.-S. Xia, H. Sahbi, and V. Prinet. Mid-level features and spatio-temporal context for activity recognition. *Pattern Recognition*, 45(12):4182–4191, 2012.

[28] W. Zhang, X. Wang, D. Zhao, and X. Tang. Graph Degree Linkage: Agglomerative Clustering on a Directed Graph. In *ECCV*, 2012.