Int J Comput Vis
DOI 10.1007/s11263-010-0393-z

Mining Layered Grammar Rules for Action Recognition

Liang Wang · Yizhou Wang · Wen Gao

Received: 14 October 2009 / Accepted: 24 September 2010

Abstract We propose a layered-grammar model to represent actions. Using this model, an action is represented by a set of grammar rules. The bottom layer of an action instance's parse tree contains action primitives such as *spatiotemporal* (*ST*) *interest points*. At each layer above, we iteratively mine grammar rules and "super rules" that account for the high-order compositional feature structures. The grammar rules are categorized into three classes according to three different *ST-relations* of their action components, namely the *strong relation*, *weak relation* and *stochastic relation*. These ST-relations characterize different action styles (degree of stiffness), and they are pursued in terms of grammar rules for the purpose of action recognition. By adopting the *Emerging Pattern* (*EP*) *mining* algorithm for relation pursuit, the learned production rules are statistically significant and discriminative. Using the learned rules, the parse tree of an action video is constructed by combining a bottom-up rule detection step and a top-down ambiguous rule pruning step. An action instance is recognized based on the discriminative configurations generated by the production rules of its parse tree. Experiments confirm that by incorporating the high-order feature statistics, the proposed method largely improves the recognition performance over the bag-of-words models.

Keywords Action recognition · Layered-grammar model · Action parse tree · Emerging pattern mining

1 Introduction

Action recognition is receiving more and more attention in the computer vision community. This is largely due to its critical role in many important applications, e.g. visual surveillance, content-based video retrieval, and human computer interaction. Human actions, oriented for certain purposes and performed by coordinating the body parts, can be considered as a type of spatiotemporal (ST) pattern possessing two important properties: (1) They are highly hierarchical and compositional, i.e. an action can be divided into a number of sub-actions, and the sub-actions can be further decomposed into more basic ones. For example, a basketball shooting action consists of squatting, body stretching, jumping, and wrist flipping to launch the ball. In turn, the squatting can be decomposed into knee bending and ankle bending, and so on. (2) There are large variations in actions, making action recognition a particularly challenging problem. The large variations can be caused by different factors such as varied action styles, environment/context constraints (e.g. avoiding an obstacle when reaching something, and interacting with other objects), and different viewing directions. However, regardless of these factors, the action variation is finally reflected by different organizations or spatiotemporal configurations of the hierarchical and compositional action components.

L. Wang
School of Computer Science and Technology, Harbin Institute of Technology, Harbin, Heilongjiang Province, China
e-mail: wangliang@jdl.ac.cn

L. Wang
Nat'l Engineering Lab for Video Technology, Peking University, Beijing, China

Y. Wang (✉) · W. Gao
Nat'l Engineering Lab for Video Technology and Key Lab. of Machine Perception (MoE), School of Electronics Engineering and Computer Science, Peking University, Beijing, China
e-mail: Yizhou.Wang@pku.edu.cn

W. Gao
e-mail: wgao@pku.edu.cn

Published online: 21 October 2010

We observe that different actions organize their components with diverse degrees of rigidity in order to serve for different purposes. For example, as shown in Fig. 1, in sign language the performer needs to follow strict rules in order to convey accurate gesture meaning; whereas, at the other extreme, the disco dancer tries to throw her body parts towards any random directions to enjoy the rhythm; There are also actions in between the two extremes, e.g. when drinking, fetching a cup of water always happens before the water is poured into a mouth; no matter how to fetch, and whether it is a sipping or guzzling. However, most actions organize their components in a mixed way. This poses a challenging problem of finding a unified representation and an efficient computation method to explain and recognize the mixed style actions in our daily life.

Fig. 1 Example actions that organize their components with different degrees of rigidity. From sign-language to drinking and to disco dancing, the randomness of the action patterns is in an increasing order

In this paper, we propose a solution to recognizing actions by leveraging their hierarchical and compositional properties. We adopt a layered attribute grammar model to encode the hierarchical and compositional spatiotemporal configurations of action components. As shown in Fig. 2, an instance of an action can be explained by building a parse tree in terms of the action grammar. The components at the leaf-nodes are *spatiotemporal interest points* (*STIPs*) (shown as colored balls in Fig. 2(a4)), they capture the changes of local appearance and motions in the action video (Laptev and Lindeberg 2003). These ST interest points constitute the bottom layer (Layer-0) of the action model, namely, the *STIP-map*. Moreover, the interest points form a number of small configurations (shown as colored regions in Fig. 2(a3)). Each configuration can be generated by an attribute grammar rule (or a number of rules) at the layer above (Layer-1). Similarly, across the layers above, the configurations at a lower layer are generated by "super" rules at the layer above. Thus, the configurations at a lower layer are called the *components* of a bigger configurations at the layer above. The instantiations of the rules at each layer constitute a *rule-map* of the video.

At each layer, the grammar rules can be slotted into three categories to account for the three *ST-relations* between the action components in a configuration, namely the *strong relation*, *weak relation* and *stochastic relation* (as shown in Fig. 2(a1)–(a3)). These ST-relations character-

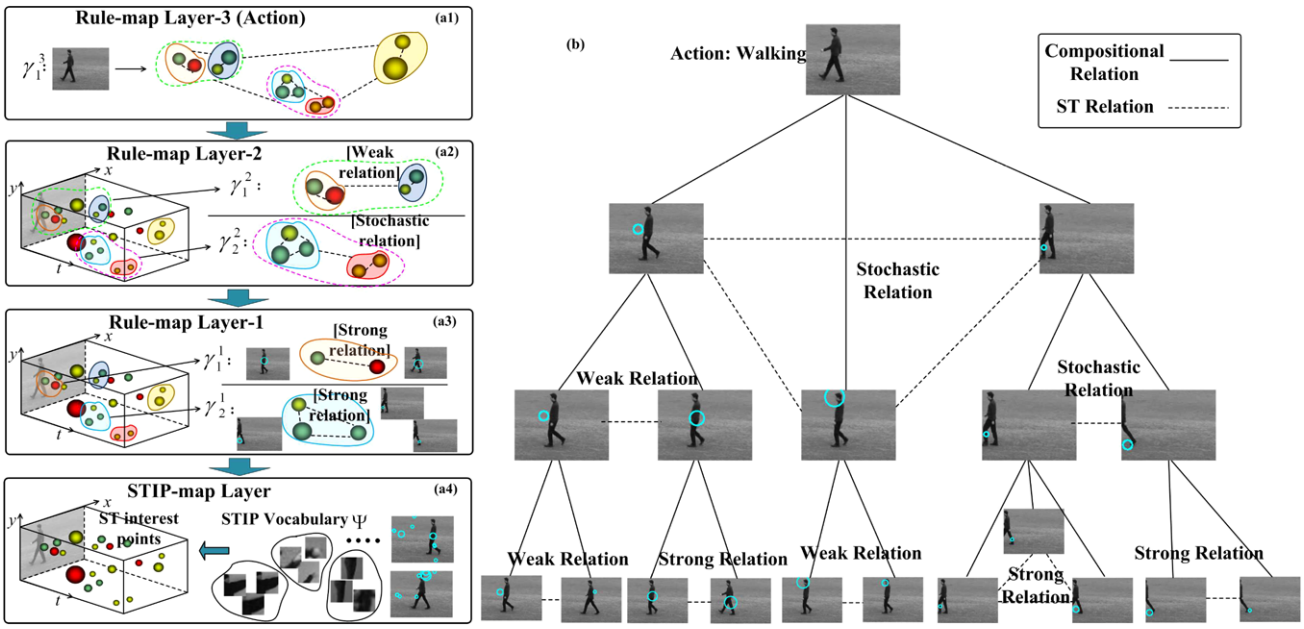

Fig. 2 The proposed hierarchical action-grammar model. The bottom layer is the ST interest point layer. Each layer above contains grammar rules and "super" rules that generate the configurations below. The colored balls in (**a4**) are the detected ST interest points of the input video. On the right side of (**a4**), the detected interest points are shown in *cyan circles* on sampled frames. In the middle of (**a4**), we show some interest point ST patches of three clusters. The right side of (**a3**) shows two mined rule instances γ_1^1 and γ_2^1. Their corresponding configurations in ST volume are shown on the left of (**a3**). Each rule instance is enclosed by a colored curve (different colors denote different rule types). (**a1**) and (**a2**) show the rule-maps at higher levels. (**b**) The parse tree of a "walking" instance

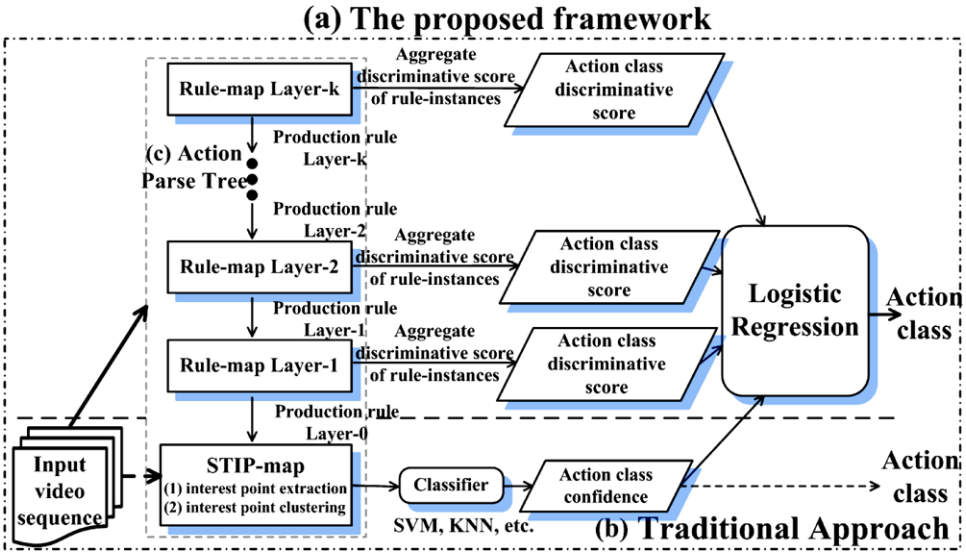

Fig. 3 The action recognition frameworks of the proposed approach and traditional interest point based methods. (**a**) The proposed approach adopts a logistic regression classifier to integrate the discriminative scores computed from each layer of an action parse tree (shown in (**c**)). (**b**) A general conventional framework for interest point based action recognition methods. It first extracts and clusters ST interest points, and then recognizes actions using the statistics of the ST interest points by classifiers, e.g. SVM, KNN

ize the three action styles introduced above. More specifically, (1) the strong relation enforces the most strict ST-relation between the components in terms of quantized ST distances (similar to a Gaussian model). (2) The weak relation adopts *Allen Algebra* to specify the ST order of the components, e.g. "ABOVE", "BELOW", "BEFORE", "AFTER". Sometimes, the partial order is crucial and sufficient to differentiate actions, e.g. picking up and putting down an object. (3) The stochastic relation assumes that the components are orderless, and it only constrains them to co-occur.

We adopt a weakly supervised learning method to learn the hierarchical structure of the production rules from action videos. Here "weakly supervised" means that we do not annotate the bounding-box of the actions, neither align the video sequences. As our primary goal is to recognize actions, we aim to learn production rules that are capable of explaining the ST-relations of action components in the local configurations, as well as to differentiate the actions from other types. This rule learning step is also called *relation pursuit*. To cope with the expensive cost of exploring the large combinatorial configuration space, we adopt an efficient data mining technique, Emerging Pattern (EP) mining (Dong and Li 2004). In this way, we mine production rules as emerging patterns such that the learned rules are statistically significant and discriminative.

Using the learned production rules, a parse tree of an action video is built in two steps, *a bottom-up multi-hypothesis rule detection step* and *a top-down ambiguous rule pruning step*. Since the learned production rules are probabilistic, a configuration can be ambiguous when it is interpreted by a number of rules in a layer. In the bottom-up step, at each layer, and for each configuration, we detect a number of production rules that best explain the configuration and keep them as the candidate rules for the final parse tree. Then, we execute the top-down step to remove the ambiguous rules. The constructed nonoverlapping hierarchical discriminative structures in term of grammar rules maximize the total discriminative scores of action parse trees. An example parse tree of a *walking* instance is shown on the right in Fig. 2.

As illustrated in Fig. 3, to recognize an action instance, we first compute a parse tree for each action class. Each parse tree has multiple layers, and for each layer we compute a discriminative score by aggregating the discriminative scores of the production rules detected in that layer. Then, we can feed these layer scores as the "regressors" into a logistic regression model already trained for that action class, and obtain a probability that this action belongs to the class. We label the action with the class of the highest probability.

In summary, this paper has the following contributions: (1) We propose a unified representation for actions of different acting styles. (2) We propose a relation pursuit method (implemented by Emerging Patterning (EP) mining algorithm) to learn different types of the grammar rules efficiently. These learned rules demonstrate strong discriminative power in recognizing different types of actions. (3) We learn a discriminative model out of a grammatical representation. Thus, the whole model leverages the grammatical representation's strong expressive power and the discriminative model's potent differentiation power plus computational efficiency.

The rest of the paper is organized as follows: Sect. 2 discusses the related work. In Sect. 3, we introduce the proposed action model, followed by presenting the model learning and action recognition methods in Sect. 4. We show experiment results in Sect. 5, and conclude the paper in Sect. 6.

2 Related Work

In this section, we briefly review the existing literature that is closely related to our work.

2.1 Spatiotemporal Interest Point Based Approaches

The conventional spatiotemporal interest point based methods model actions as a bag-of-video-words (Schuldt et al. 2004; Dollár et al. 2005). As shown in Fig. 3, this group of methods generally contain three major steps: (1) extracting ST interest points from action videos, (2) clustering the interest points based on their ST features (e.g. histogram of oriented gradient (Dalal and Triggs 2005), SIFT (Lowe 2004)), and (3) classifying/recognizing actions using the occurrence frequencies of the clustered ST interest points. Researchers have focused on each step to improve the final action recognition performance: (1) To extract robust and informative interest points, various criteria are proposed, e.g. cornerness (Laptev and Lindeberg 2003), periodicity (Dollár et al. 2005), saliency (Rapantzikos et al. 2009). (2) To cluster STIPs, K-means is used in the feature space of the interest points. Recently, semantic based clustering strategies are proposed to resolve the difficulties in selecting a proper K value for the K-means algorithm and the disagreement between appearance similarity and semantic consistency (Quelhas et al. 2007). Based on Dollár's ST interest point detector, Niebles et al. model actions using a bag-of-word model, and cluster the interest points by the underlying "topics" (Niebles et al. 2008). Liu and Shah propose to cluster ST interest points by exploring the correlation between interest points and actions types (Liu and Shah 2008). (3) To recognize actions, people use classifiers (e.g. SVM—Schuldt et al. 2004, or K-nearest neighbor—Dollár et al. 2005), or the Bayesian framework of a generative action model (Niebles et al. 2008) for action recognition.

Besides the proposed method, in the literature, there are methods that also exploit the local configurations of interest points. For example, Sivic and Zisserman (2004) propose a method to find significant objects in a video sequence by measuring the re-occurrence of spatial configurations of Harris corners (Harris and Stephens 1988) in video frames. Gilbert et al. (2008) apply the Harris corner detector in all $x-y$, $y-t$ and $x-t$ planes of action videos to detect a set of dense interest points, then discover the frequent STIP configurations by the Apriori algorithm (Agrawal and Srikant 1994; Quack et al. 2007). These frequent configurations constitute the descriptive itemsets of the action, and they are used for action recognition.

Despite of the development in the above work, there is lack of work on exploring high-order ST configurations of local features. Although, in the literature, people also use other kinds of hierarchical features extracted from down sampled images for object detection, e.g. Schnitzspan et al. (2009), the feature hierarchy is not built from the interest points. Most of the traditional interest point based methods try to find recognition cues from the low-order statistics of interest point maps. Table 1 compares the feature hierarchies exploited in the proposed approach to those of some existing approaches. As shown in the table, our approach makes full use of the ST interest point configurations at multiple layers, and we learn high-order ST interest point distributions in terms of layered grammar rules. It is worth noting that our model is a general framework to enhance the performance of existing interest point based approaches by incorporating high-order statistics of feature configurations.

Table 1 Comparison of feature hierarchies exploited in the proposed grammar model and other existing models

Approaches	Level of hierarchy	STIP-map	Low-order configuration	High-order configuration
Bag-of-words approaches (Dollár et al. 2005, Schuldt et al. 2004)	1	√		
Local configuration based approaches (Sivic and Zisserman 2004, Gilbert et al. 2008)	2		√	
Our approach	4	√	√	√

2.2 Grammar-Based Approaches

The grammatical model is a powerful representation for describing generative processes. In the literature, grammar models have been widely applied to event analysis (Lin et al. 2009; Joo and Chellappa 2006). Ivanov and Bobick (2000) adopt stochastic context free grammars (SCFG) to represent events. They first apply atomic action detectors to label possible atomic actions in an input video, then feed the labels as a string to the following SCFG parsing module for event analysis. Joo and Chellappa (2006) propose to use attribute grammar and incorporate attributes such as the location and the actor identity to facilitate event recognition. In their work, these attributes serve as constraints for a parser to analyze events. Ryoo and Aggarwal (2009) design a system to parse activities with interactions based on a robust detection and tracking system. However, the data is collected from well controlled environment. The grammar-based methods usually depend on pre-designed production rules, and they are based on explicit tracking and detection of individual object motions. It is known that the reliability of low level video processing methods (e.g. detection and tracking) are still far from satisfactory, and using predefined

rules is not practical in real scenarios. Therefore, a robust action recognition module and an automatic production rule learning method are highly demanded.

3 The Action Model

In this section, we introduce the layered grammar model for action representation and the probabilistic formulation of the attribute grammar rules.

3.1 The Attribute Grammar Model

Our action model is a 5-tuple attribute graph grammar $\mathcal{G} = \{\mathcal{L}, V_T, \Gamma, V_N, P\}$.

$\mathcal{L}$ is the starting root node. In the context of action video parsing, the root node refers to the action class.

V_T is the terminal node set, which corresponds to the ST interest point set of action videos.

V_N denotes the non-terminal node set, which corresponds to the instantiations of the production rules.

Γ is the production rule set. A production rule accounts for the compositional structure of a local configuration, including its components and the ST-relation between the components.

The attribute grammar is a context free grammar (CFG), i.e. each production rule takes the form of $\gamma : A \to \alpha$, where $A \in V_N$ is a non-terminal node, and $\alpha \in (V_N)^+ \cup (V_T)^+$ is a string of terminal or non-terminal nodes. It is different from the conventional CFG definition because α in our representation is only allowed to be either a string of terminal nodes or a string of non-terminal ones, and no mixture of them is allowed.

Both a terminal node a and a non-terminal node A are associated with a set of attributes describing certain properties of a configuration, e.g. scale and ST position. We denote their attribute functions by $X(a)$ and $X(A)$, respectively. Considering a rule γ, e.g. of the form $\gamma : A \to A_1 A_2$, it enforces constraints in the form $g_{\gamma,i}(X(A)) = f_{\gamma,i}(X(A_1), X(A_2))$ for respective attributes of the left- and right-hand sides of γ. $g_{\gamma,i}$ and $f_{\gamma,i}$ are functions of the node attributes. n_γ is the number of constraints.

P is the probability model of the grammar.

Using the grammar rules, an action instance is represented as a parse tree. Let $\{\gamma_1, \gamma_2, \ldots, \gamma_{n_O^{\mathcal{L}}}\} \subseteq \Gamma_{\mathcal{L}}$ be the production rule set used to expand the root node $\mathcal{L}$ to a parse tree $\mathbf{pt}_{\mathcal{L}}^{O}$ for an action video O. The parse tree is denoted as

$$\mathbf{pt}_{\mathcal{L}}^{O} = \left(\gamma_1, \gamma_2, \ldots, \gamma_{n_O^{\mathcal{L}}}\right) \tag{1}$$

where $n_O^{\mathcal{L}}$ is the number of production rules used in parsing O.

3.2 The Terminal Nodes: ST Interest Points

The terminal nodes of parse trees are ST interest points of the videos, denoted as

$$V_T = \left\{(a, X(a)) : a \in \Psi\right\} \tag{2}$$

where Ψ is the ST interest point set extracted from all training action videos. $X(a) = (l(a), s(a), u(a))$ is the attribute vector of an ST interest point a, where $l(a)$ denotes a's cluster label, $s(a)$ is a's scale, and $u(a)$ is the center ST location of a in the corresponding action video. The cluster label of an interest point is obtained by clustering all the detected interest points of the training videos. The scale and the center position of an interest point are obtained by the interest point detector.

3.3 The Non-Terminal Nodes: Instantiations of the Rules

The non-terminal node set V_N of parse trees is denoted as

$$V_N = \{(A, X(A)) : A \in R\} \tag{3}$$

where R denotes the set of possible instantiations of the production rules in action videos. $X(A)$ is the attribute set of A that can be instantiated.

$$X(A) = \left(\mathbf{id}(A), \mathrm{rt}(A), \mathbf{s}(A), \mathbf{u}(A), \Theta(A)\right) \tag{4}$$

$\mathbf{s}(A)$ and $\mathbf{u}(A)$ are the scale and ST position of A respectively. $\mathrm{id}(A)$ is the index of the production rule which instantiates A. $\mathrm{rt}(A) \in \{\text{strong, weak, stochastic}\}$ is the component relation type of the production rule. $\Theta(A)$ denotes the component relation parameter set of A (defined in (7), (16) and (26)).

3.4 The Production Rules

As discussed in Sect. 1, we identified three major action styles which are characterized by the three types of relations between action components. Correspondingly, we define three types of production rules in our action grammar to account for the ST-relation of the components in configurations.

The general form of a production rule is

$$\gamma : A \to \left(\alpha_1 \alpha_2 \cdots \alpha_{n_\gamma}\right) \quad [\mathrm{rt}, \varepsilon, \rho, \Theta] \tag{5}$$

where $A \in V_N$. $\mathrm{rt} \in (s, w, st)$ indicates the rule's relation type. $\alpha_i \subseteq (V_T)^+ \cup (V_N)^+$ is a string of (non-) terminal nodes. n_γ is the length of the string. 'ε' and 'ρ' denote the *support ratio* and *growth ratio* of the rule respectively. The support ratio refers to the occurrence frequency of the rule in the positive video set (the action videos containing the to-be-modeled type of actions), and the growth ratio tells the

discriminative power of the production rule, which is computed as the ratio of γ's occurrence frequencies in positive video set over that in negative video set. Θ is the component relation parameter set specifying the relations between γ's components, e.g. their relative scales, relative ST distances.

The prior probability of γ is computed as

$$p(\gamma) = \frac{\#(\gamma)}{\sum_{\gamma'} \#(\gamma')} \tag{6}$$

where $\#(\gamma)$ is the number of γ's occurrence in the training video sequences.

In the following sections, we introduce the detailed formulation of the three types of rules including their attributes, constraint equations, and probability models.

3.4.1 The Strong Rules

A strong rule γ_s enforces the most strict pairwise ST-relation between configuration components. Let's denote a configuration by $\mathbf{c} = \{c_i\}_{i=1}^n$. When $\mathbf{c}$ is instantiated by γ_s, its component relation parameter set is

$$\Theta(\mathbf{c}) = \{\mathbf{s_c}, \mathbf{id_c}, \mathbf{u_c}\} \tag{7}$$

where $\mathbf{s_c} = (\mathbf{s}_{c_1}, \ldots, \mathbf{s}_{c_n})$ and $\mathbf{u_c} = (\mathbf{u}_{c_1}, \ldots, \mathbf{u}_{c_n})$ denote the relative scale vector and the relative ST distance vector, respectively. More specifically, the relative scale $\mathbf{s}_{c_i}$ of a component c_i is defined as $\mathbf{s}(c_i)/\mathbf{s}(c_1)$. ($c_1$ is the center component of the configuration.) The relative distance is obtained similarly. $\mathbf{id_c} = (\mathrm{id}(c_1), \ldots, \mathrm{id}(c_n))$ is the rule index vector.

The constraint functions of γ_s are

$$\mathbf{s_c} = f_{s_{\gamma_s}}\big(\mathbf{s}(c_1), \ldots, \mathbf{s}(c_n)\big) \tag{8}$$

$$\mathbf{u_c} = f_{u_{\gamma_s}}\big(\mathbf{u}(c_1), \ldots, \mathbf{u}(c_n)\big) \tag{9}$$

$$\mathbf{id_c} = \big[\mathrm{id}(c_1), \ldots, \mathrm{id}(c_n)\big]^T \tag{10}$$

$$\mathbf{s}(\mathbf{c}) = \mathbf{s}(c_1) \tag{11}$$

$$\mathbf{u}(\mathbf{c}) = \mathbf{u}(c_1) \tag{12}$$

$$\mathrm{rt}(\mathbf{c}) = \mathrm{strong} \tag{13}$$

$f_{s_{\gamma_s}}$ and $f_{u_{\gamma_s}}$ compute the relative scale vector and the relative ST distance vector of $\mathbf{c}$ from its components. (10), (11), (12) are assignment equations. For example, (11) assigns the scale of the central component c_1 to $\mathbf{c}$.

The likelihood of $\mathbf{c}$ under rule γ_s is

$$p(\mathbf{c}|\gamma_s) = \frac{1}{Z_{\gamma_s}} q(\mathbf{c}|\gamma_s) \tag{14}$$

where

$$q(\mathbf{c}|\gamma_s) = \frac{1}{n}\sum_{j=1}^{n} \delta\big(\mathbf{s}_{c_j}, \mathbf{s}_{\gamma_s,j}\big) \times \delta\big(\mathbf{u}_{c_j}, \mathbf{u}_{\gamma_s,j}\big) \cdot \delta\big(\mathrm{id}(c_j), \mathbf{id}_{\gamma_s,j}\big) \tag{15}$$

computes the confidence score. $\mathbf{s}_{\gamma_s}$, $\mathbf{u}_{\gamma_s}$ and $\mathbf{id}_{\gamma_s}$ are the rule's canonical settings of the component relation parameters. $\delta(\cdot)$ is a Delta function. Thus, the confidence score tells the portion of a configuration being consistent with a rule's canonical setting. The normalization term $Z(\gamma_s) = \sum_{\mathbf{c}} q(\mathbf{c}|\gamma_s)$ is computed by summing over all the configurations of γ_s in the limited discretized configuration space. Figure 4 shows how to compute $\mathbf{s_c}$ and $\mathbf{u_c}$ in a three-component configuration.

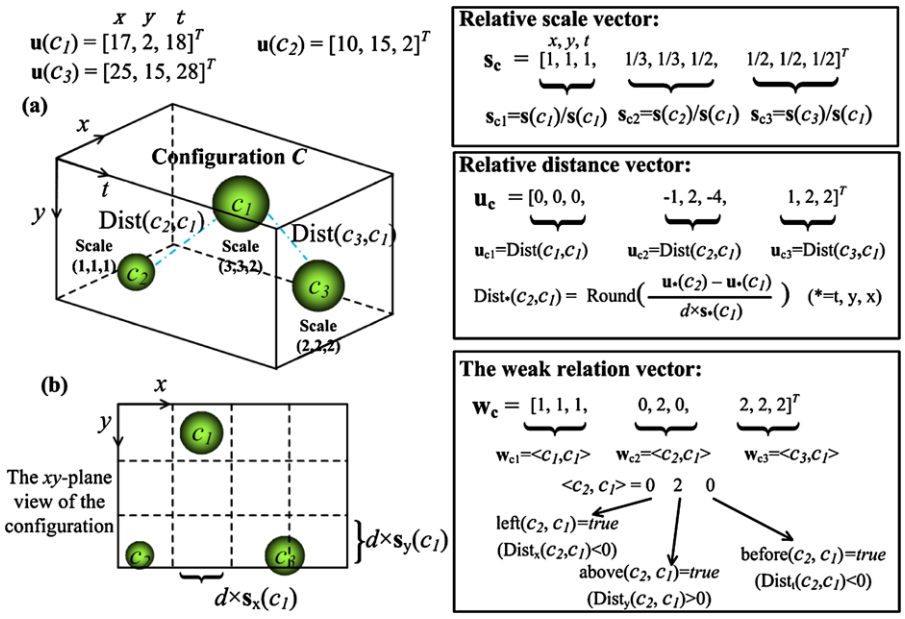

Fig. 4 An illustration of how to estimate the relative scale vector, the relative distance vector and the weak relation vector from a three-component configuration. (**a**) A configuration composed of three components, c_1, c_2 and c_3, where c_1 is the central component. (**b**) The ST space is quantized into a grid with the cell size d times of the scale of c_1. The relative distance between each component and c_1 is quantized as the number of cells between them in x, y, t. The function Round(z) rounds z to its nearest integer

3.4.2 The Weak Rules

A weak rule γ_w specifies spatiotemporal orders of its components. When using the weak rule to instantiate the configuration $\mathbf{c}$, $\mathbf{c}$'s component relation parameter set is

$$\Theta(\mathbf{c}) = \{\mathbf{s_c}, \mathbf{id_c}, \mathbf{w_c}\} \tag{16}$$

where $\mathbf{s_c}$ and $\mathbf{id_c}$ have the same meaning as the definition in the strong rules. $\mathbf{w_c} = (\mathbf{w}_{c_1}, \ldots, \mathbf{w}_{c_n})$ is the weak relation vector which describes the relative weak relations between $\{c_i\}_{i=1}^n$ and the central component c_1. $\mathbf{w}_{c_i}$ $(i = 1, 2, \ldots, n)$ specifies weak relation between c_i and c_1.

To describe the weak relation (ST partial order) between two components, we employ the *Allen temporal predicates* (Allen and Ferguson 1994) and extend it with some more spatial relation terms. The augmented set of predicates are {'before', 'after', 'above', 'below', 'left', 'right', 'overlap$_x$', 'overlap$_y$', 'overlap$_t$'}, which are defined as follows

$$\begin{aligned}
&\text{left}(a,b) \Longleftrightarrow \text{Dist}(a,b).x < 0\\
&\text{right}(a,b) \Longleftrightarrow \text{Dist}(a,b).x > 0\\
&\text{above}(a,b) \Longleftrightarrow \text{Dist}(a,b).y < 0\\
&\text{below}(a,b) \Longleftrightarrow \text{Dist}(a,b).y > 0\\
&\text{before}(a,b) \Longleftrightarrow \text{Dist}(a,b).t < 0\\
&\text{after}(a,b) \Longleftrightarrow \text{Dist}(a,b).t > 0\\
&\text{overlap}_x(a,b) \Longleftrightarrow \text{Dist}(a,b).x = 0\\
&\text{overlap}_y(a,b) \Longleftrightarrow \text{Dist}(a,b).y = 0\\
&\text{overlap}_t(a,b) \Longleftrightarrow \text{Dist}(a,b).t = 0
\end{aligned} \tag{17}$$

where a and b are a pair of components, and $\text{Dist}(a,b)$ measures the quantized distance between them. Figure 4 is an illustration about how to compute $\mathbf{w_c}$ in a configuration. We use a 3-digit number to represent the weak relations between a and b. Each digit encodes the relation in one of the x, y and t dimensions. For example, $t = 0$ when $\text{before}(a,b) = \textit{true}$; $t = 1$ when $\text{overlap}_t(a,b) = \textit{true}$; and $t = 2$ when $\text{after}(a,b) = \textit{true}$. It is similarly defined in the x and y dimensions.

The constraint functions are listed as follows

$$\mathbf{s_c} = f_{s_{\gamma_w}}\big(\mathbf{s}(c_1), \ldots, \mathbf{s}(c_n)\big) \tag{18}$$

$$\mathbf{w_c} = f_{w_{\gamma_w}}\big(\mathbf{u}(c_1), \mathbf{s}(c_1), \ldots, \mathbf{u}(c_n), \mathbf{s}(c_n)\big) \tag{19}$$

$$\mathbf{id_c} = \big[\text{id}(c_1), \ldots, \text{id}(c_n)\big]^T \tag{20}$$

$$\mathbf{s}(\mathbf{c}) = \mathbf{s}(c_1) \tag{21}$$

$$\mathbf{u}(\mathbf{c}) = \mathbf{u}(c_1) \tag{22}$$

$$\text{rt}(\mathbf{c}) = \text{weak} \tag{23}$$

Equation (18), (20), (21) and (22) are defined in the same way as the constraint functions of the strong rules. In (19) the function $f_{w_{\gamma_w}}(\cdot)$ returns a weak relation vector depicting the pairwise ST partial orders between the components $\{c_i\}_{i=1}^n$ and c_1.

The probability that $\mathbf{c}$ is generated by γ_w is

$$p(\mathbf{c}|\gamma_w) = \frac{1}{Z_{\gamma_w}} q(\mathbf{c}|\gamma_w) \tag{24}$$

where

$$\begin{aligned}
q(\mathbf{c}|\gamma_w) = \frac{1}{n}\sum_{j=1}^{n} &\delta\big(\mathbf{s}_{c_j}, \mathbf{s}_{\gamma_w,j}\big)\\
&\times \delta\big(\mathbf{w}_{c_j}, \mathbf{w}_{\gamma_w,j}\big) \cdot \delta\big(\text{id}(c_j), \mathbf{id}_{\gamma_w,j}\big)
\end{aligned} \tag{25}$$

computes the confidence score. $\mathbf{s}_{\gamma_w}$, $\mathbf{w}_{\gamma_w}$ and $\mathbf{id}_{\gamma_w}$ are the rule's canonical settings of the component relation parameters. $Z(\gamma_w) = \sum_{\mathbf{c}} q(\mathbf{c}|\gamma_w)$ is the normalization term.

3.4.3 The Stochastic Rules

A stochastic rule γ_{st} assumes that its components are orderless but co-occur in its rule instances. When configuration $\mathbf{c}$ is an instantiation of a stochastic rule, its component relation parameter set is

$$\Theta(\mathbf{c}) = \{\mathbf{s_c}, \mathbf{id_c}\} \tag{26}$$

where $\mathbf{s_c}$ and $\mathbf{id_c}$ have the same meaning as defined in both the strong and the weak rules.

The constraint equations of the stochastic rule are

$$\mathbf{s_c} = f_{s_{\gamma_{st}}}\big(\mathbf{s}(c_1), \ldots, \mathbf{s}(c_n)\big) \tag{27}$$

$$\mathbf{id_c} = \big[\text{id}(c_1), \ldots, \text{id}(c_n)\big]^T \tag{28}$$

$$\mathbf{s}(\mathbf{c}) = \mathbf{s}(c_1) \tag{29}$$

$$\mathbf{u}(\mathbf{c}) = \mathbf{u}(c_1) \tag{30}$$

$$\text{rt}(\mathbf{c}) = \text{stochastic} \tag{31}$$

These equations are the same as those in the strong and weak rules.

The probability that $\mathbf{c}$ is generated by γ_{st} is

$$p(\mathbf{c}|\gamma_{st}) = \frac{1}{Z_{\gamma_{st}}} q(\mathbf{c}|\gamma_{st}) \tag{32}$$

where the confidence term $q(\mathbf{c}|\gamma_{st})$ is

$$q(\mathbf{c}|\gamma_{st}) = \frac{1}{n}\sum_{j=1}^{n} \delta\big(\mathbf{s}_{c_j}, \mathbf{s}_{\gamma_{st},j}\big) \cdot \delta\big(\text{id}(c_j), \mathbf{id}_{\gamma_{st},j}\big) \tag{33}$$

$\mathbf{s}_{\gamma_{st}}$ and $\mathbf{id}_{\gamma_{st}}$ are the rule's canonical settings of the component relation parameters. The normalization term $Z(\gamma_{st})$ is computed as $Z(\gamma_{st}) = \sum_C q(\mathbf{c}|\gamma_{st})$.

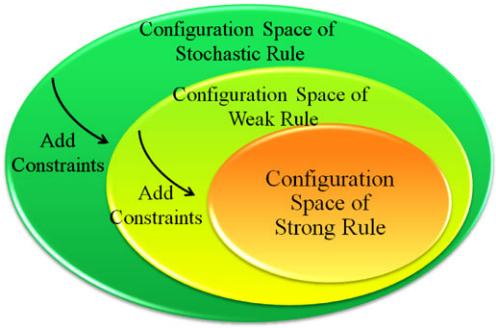

Fig. 5 The relation of the configuration spaces governed by the three types of production rules. The stricter the type of rules are, the more constrained of the configuration space is

3.4.4 Discussion about Three Types of Rules

Figure 5 shows the relation of the three types of grammar rules. From the stochastic rules to the weak rules and to the strong rules, they constrain the ST-relation of a configuration components in an increasing order of strictness. The stochastic rules provide the least structure information of a configuration; so the configurations under this type of rules span the largest configuration space. Whereas, the strong rules use the most strict constraints to confine the configuration space, and they govern the smallest configuration space. Note that, if a configuration has a high probability of being interpreted by a strong rule, it also can be explained by a weak/stochastic rule. However, without adding more specific constraints, some action types are hard to tell from each other. For instance, for 'walking' and 'running', although using some stochastic rules is easy to differentiate the two actions from the other types (e.g. drinking), in order to differentiate between these two similar actions we have to further specify/constrain the spatiotemporal relation of their action components. Thus, in principle we adopt two parameters in the relation pursuit (rule learning) process (Sect. 4.1), the *support ratio* and *growth ratio*, to learn the production rules. A relation is picked only according to its discriminative power derived from the two parameters. For a configuration, when multiple relations pass the threshold simultaneously, we select the one with the maximum likelihood to explain it. In this way, we exploit the rules' descriptive power as well as their discriminative power in recognizing actions. The detailed learning and recognition methods are introduced as follows.

4 Learning and Recognition

In this section, we present methods of learning the action grammar models and recognizing actions using the learned models.

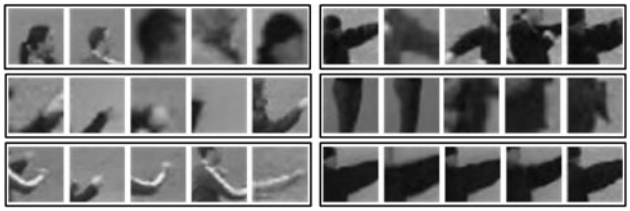

Fig. 6 Samples of ST interest points from six clusters of the KTH human action dataset (Schuldt et al. 2004) detected by Laptev STIP detector. Each ST interest point is represented by its center patch in the ST volume

4.1 Learning the Layered Grammar Rules

We adopt a weakly supervised learning scheme to learn the production rules from action videos layer by layer. Here "weakly supervised" means that we only tag the training videos with the action types, but neither bounding-box nor align the actions. We first build the STIP-maps of the training videos as follows: (1) We detect ST interest points from all the training videos using the method in Laptev and Lindeberg (2003) or Dollár et al. (2005) at multiple spatiotemporal scales. (2) These interest points are clustered into a pre-defined number of clusters according to their appearance feature vectors. The features are extracted from the spatiotemporal cuboids centered at these STIPs. Then, the ST interest points in the STIP-maps are represented by their locations, scales, and indices of the clusters they belong to. Examples of the ST interest point clusters are shown in Fig. 6. It can be seen that the extracted interest points capture the local motion and appearance changes of the video sequences. (3) We discover a set of production rules that generate the configurations on the STIP-maps. These rules form the Layer-1 rule-map of the model. Consequently, we learn a set of 'super rules' from the Layer-1 rule maps and they form the Layer-2 rule maps. This process iterates until there is no rule can be discovered or the layer number reaches a preset value.

Because the model is learned for action recognition, we want to mine production rules that are discriminative between action classes. Because the learned models are representations for each class, the rules for one action class should have high occurrence frequency in the videos of that class, but rarely appear in the others. Moreover, since there are a large number of ST interest points distributed in action videos (about 400 interest points in a 120 frame simple background 180×120 resolution action video in KTH dataset), an efficient rule discovering algorithm is required to explore such a large configuration space. A recently developed data mining technique, Emerging Pattern (EP) mining, is quite suitable for this demanding task.

4.1.1 Emerging Pattern (EP) Mining

In this section, we briefly introduce the Emerging Pattern mining method (Dong and Li 2004), and present how to apply it in our framework to mine production rules. Intuitively, EP mining finds the itemsets whose support ratios vary a lot from one dataset to another. We follow the notation in (Dong and Li 2004) to introduce the mathematical definition of EP mining. Let $I = \{i_1, i_2, \ldots, i_N\}$ be a set of N items. A *transaction* is a subset T of I. A *dataset* D is a set of transactions. A subset X is called a *k-itemset* if $k = \|X\|$. If $X \subseteq T$, we say the transaction T contains the itemset X. The *support* of an itemset X in a dataset D is defined as $\varepsilon_D(X) = \frac{\text{count}_D(X)}{\|D\|}$, where $\text{count}_D(X)$ is the number of transactions in D containing X. Given an itemset X and a pair of datasets D_1 and D_2, the *growth rate* of an itemset X from D_1 to D_2 is computed as

$$\rho(X) = \begin{cases} 0, & \text{if } \varepsilon_{D_1}(X) = 0 \text{ and } \varepsilon_{D_2}(X) = 0 \\ \infty, & \text{if } \varepsilon_{D_1}(X) = 0 \text{ and } \varepsilon_{D_2}(X) \neq 0 \\ \frac{\varepsilon_{D_2}(X)}{\varepsilon_{D_1}(X)}, & \text{otherwise} \end{cases}$$

A pattern is said to be a ν-*emerging pattern* from D_1 to D_2 if $\rho(X) > \nu$. Different from frequent itemset mining techniques like Apriori (Agrawal and Srikant 1994), EP mining is more interested in itemset's degree of changes in support. Recently, it has been used to build classifiers, and according to Dong et al. (1999), Alhammady and Ramamohanarao (2006), it performs very well in many classification tasks.

We use EP mining in this project as follows: (1) At each layer, a large number of local neighbor sets are randomly sampled from both the positive and negative datasets. The positive dataset contains the video sequences of the to-be-modeled action type, whereas the negative dataset consists of video clips belonging to other types of actions. The components of a local neighbor set can be either the ST interest points at Layer-0 or rule instances at higher layers. Each local neighbor set is transformed to a transaction (an itemset) using a quantization method. (2) Then, we mine production rules as emerging patterns from the negative transactions to the positive transactions using EP mining. We define two parameters in mining emerging patterns, i.e. the *support* in positive samples and the *growth ratio*. The *support* parameter describes the descriptive power of the mined rules of the positive data, whereas the *growth ratio* describes the discriminative power of the mined rules.

4.1.2 Mining Production Rules by Relation Pursuit

To pursue the layered grammar rules, we first construct STIP-maps of training action videos using the methods introduced at the beginning of Sect. 4.1. Then, we mine production rules by the relation pursuit utilizing EP mining algorithm layer by layer.

To mine production rules at Layer-i, we sample a big number of *local neighbor sets* from the ith layer of all the training videos (STIP-maps for $i = 0$, rule-maps for $i > 0$). Each local neighbor set is composed of a number of components within a local spatiotemporal range. The ST range is determined such that each local neighbor set contains at least 12 components. An example of a local neighbor set is shown in Fig. 7, where the central component is pt1 and pt2–pt12 are its 11 nearest-neighbors. As EP mining only deals with numerical itemsets as inputs, an important issue of rule mining is to turn each local neighbor set into a digital itemset. Because there are three types of ST-relations for the components, we design three corresponding quantization methods to transform each component of a local neighbor set into a digital item according to its attributes (scale, ST location, label, etc.) and their ST-relation with the central component. The three different quantization methods are introduced as follows.

For the strong rules, a local neighbor set is equally divided into a $5 \times 5 \times 5$ grid in spatiotemporal domain. Fig. 7(a) shows a slice of the grid of a local neighbor set at $t = 2$. Each component within the local neighbor set is assigned a vector, of which the digits are the component's label index (the cluster index of a interest point or the index of a production rule), the quantized ST distance to the central component pt1, the relative scale w.r.t. pt1, and the duplication index. Figure 7(a) illustrates how to compute the vectors for the components. For instance, pt5's label index is 1, its position in the grid is $(0, 3, 2)$ (where the upper-left cell's coordinate is $(0, 0, 2)$), and its relative ST scale w.r.t. pt1 is $(1, 1.3, 1.3)$, which is the 7th relative scale (please see the figure caption for detailed explanation). Because there is no other component possessing the same attributes as pt5, its duplicate index is 0. Thus, by concatenating these digits, the vector for pt5 is (103270). The duplicate index is used to differentiate the components with the same preceding digits in the vector. For example, pt8 and pt9 are in the same cell, and they share the same scale and label index. Their duplicate indices are set to 0 and 1 respectively. Note that the order of the duplicate index does not matter and can be assigned randomly to the components.

For the weak rule, the ST-relation between pt2–12 and pt1 is defined by their ST partial order in terms of Allen algebra (Allen and Ferguson 1994). The definition of the ST partial order between two components is introduced in Sect. 3.4.2. Figure 7(b) shows how to compute the weak relation itemsets for mining weak production rules. Taking pt6 as an example, its label is 7, the weak relation vector w.r.t. pt1 is $(0, 0, 1)$ (i.e. left(pt6, pt1) = *true* in x-dimension, below(pt6, pt1) = *true* in y-dimension, and overlap_t(pt6, pt1) = *true* in t-dimension), its relative scale index w.r.t. pt1 is 3, and its duplicate index is 0. Thus, pt6's vector is (700130).

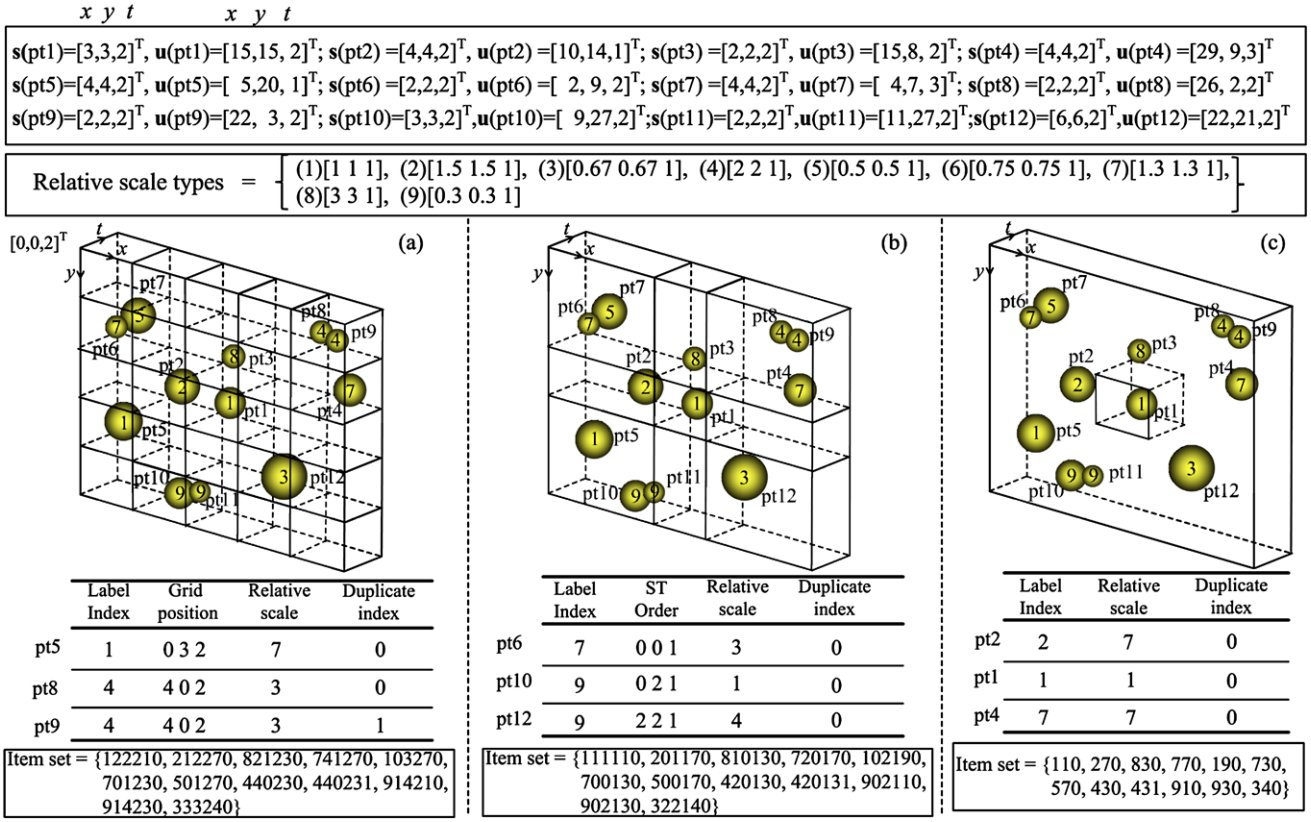

Fig. 7 An illustration of how to transform a twelve-component local neighbor set into an itemset for the strong relation, the weak relation and the stochastic relation. Each *yellow ball* represents a component (an ST interest point at Layer-0 or a configuration at Layer-i ($i > 0$)). The number at the center of each ball denotes its label. The *top row* lists every component's actual scale and location vectors. The *2nd row* lists the nine distinct relative scales. In this configuration, there are totally four types of actual scales, $[2, 2, 2]^T$, $[3, 3, 2]^T$, $[4, 4, 2]^T$ and $[6, 6, 2]^T$. The relative scales are at most $P_4^2 = 3 \times 4 = 12$. Removing the redundant ones, there are nine distinct relative scales. (**a**) For the strong rules, the grid model is used to turn the local neighbor set into a numerical itemset. (**b**) The method to generate numerical itemsets for the weak rule. (**c**) The method to generate numerical itemsets for the stochastic rule

For the stochastic rule, the vector for each component is simply determined by its label index, the relative scale w.r.t. pt1 and the duplicate index. The method to compute the numerical items for pt1–pt12 is shown in Fig. 7(c).

Using the above methods, each local neighbor set is transformed into three itemsets corresponding to the three different types of rules (as shown in the bottom of Fig. 7). Then, for each rule type, we apply EP mining to the itemsets from the negative class to the positive class to obtain a set of emerging patterns. These emerging patterns are subsequently converted to the production rules. The canonical terms of the production rules, $\mathbf{s}_{\gamma_{\{s,w,st\}}}$, $\mathbf{u}_{\gamma_s}$, $\mathbf{id}_{\gamma_{\{s,w,st\}}}$ and $\mathbf{w}_{\gamma_w}$ in (14), (24), and (32), are obtained by the inverse process of the quantization methods introduced above. That is, the quantization methods encode the components of a local neighbor set into digits; to compute the canonical terms, we decode the digits of the EPs back to their configuration space. All the production rules (irrespective of the types or the action classes) are collected together as the production rule set for the current layer.

To learn the hierarchy of the layered grammar rules, the relation pursuit procedure iterates layer by layer until no rules can be mined, or a preset threshold of the maximum number of layers is reached. As a result, the rules at a higher layer capture a larger structure of actions in space and time. This is one advantage of the proposed method compared with most of the conventional action models.

4.2 Constructing a Parse Tree For an Action Class

We adopt a *bottom-up multi-hypothesis rule detection step* followed by a *top-down ambiguous rule pruning step* to build the action parse tree for an action class of an input video sequence.

Because the learned production rules are probabilistic and have overlap in themselves, a local configuration can be interpreted by a number of rules in a layer. In the bottom-up multi-hypothesis rule detection step, at each layer, for each local configuration, we find a number of rules that explain it with likelihood larger than a threshold. These explanations

are kept as candidate instantiations of the production rules for the final parse tree. Then, we introduce the top-down step to remove the ambiguous rule instances and noise layer by layer.

The other side of removing redundant rule instances from a parse tree is to select a maximum subset of the rule instances that have the maximum class differentiation power, but their descendant components do not overlap with each other, i.e. one configuration is interpreted by a single rule. This problem can be formulated as quadratic Boolean optimization problem (Leonardis et al. 1995) as follows.

Let m be the number of the over-complete rule instances. Q is an $m \times m$ matrix encompassing the overlap of rule instances as follows:

$$Q(i,j) = \begin{cases} \tau_i(c), & \text{if } i = j \\ -\infty, & \text{if rule instance } i \text{ and } j \\ & \quad \text{overlap in their descendants} \\ 0, & \text{otherwise} \end{cases}$$

where $\tau_i(c)$ is the discriminative score of c explained by γ_i (to be introduced in the following paragraphs). Let ς denote an indicator vector, where if rule i is selected, $\varsigma(i) = 1$; otherwise, $\varsigma(i) = 0$.

The maximum subset of rules selection for a layer of a parse tree can be achieved by finding the optimal set of ς for the following quadratic Boolean optimization problem

$$\varsigma^* = \arg\max_{\varsigma} \varsigma^T Q \varsigma \tag{34}$$

We adopt the method introduced in Leibe et al. (2008) to solve this optimization problem. Please refer to Leibe et al. (2008) for details. The algorithm is summarized in Algorithm 1.

Algorithm 1 Algorithm of ambiguous rule pruning

$\varsigma \leftarrow (0, 0, \ldots, 0)^T$ Repeat the following steps

1. For each element ς_i of ς that equals to 0
   $\tilde{\varsigma} \leftarrow \varsigma, \tilde{\varsigma}_i \leftarrow 1$
   $S_i \leftarrow \tilde{\varsigma}^T Q \tilde{\varsigma} - \varsigma^T Q \varsigma$
2. $k \leftarrow \arg\max_i S_i$
3. if $S_k > 0$, $\varsigma_k \leftarrow 1$; Otherwise, algorithm end.

Output ς.

The Discriminative Score $\tau(c)$ According to Dong et al. (1999), the discriminative score $\tau(e)$ of an emerging pattern e is computed as

$$\tau(e) = \left(\frac{\rho_e}{\rho_e + 1} \cdot \varepsilon_e \right) \tag{35}$$

where ρ_e and ε_e are the growth ratio and support ratio of e. However, unlike the deterministic itemsets in Dong et al. (1999), here the configurations are generated by production rules probabilistically, so we weight the discriminative score with a posterior probability $p(\gamma|c)$, where c is the configuration and γ is the interpreting rule. Then the discriminative score of c is

$$\tau_\gamma(c) = \left(\frac{\rho_\gamma}{\rho_\gamma + 1} \cdot \varepsilon_\gamma \right) \cdot p(\gamma|c) \tag{36}$$

where $p(\gamma|c) \propto p(\gamma) \cdot p(c|\gamma)$. $p(\gamma)$ is the prior defined in (6) and $p(c|\gamma)$ is the likelihood defined in (14), (24), and (32). ρ_γ and ε_γ are the support ratio and growth ratio of the production rule γ respectively. They are automatically determined by the EP mining algorithm.

This top-down step iterates from the top layer to the bottom to prune the redundant rules from a parse tree. The rule instances that have no ancestor nodes are directly linked to the root node. Figure 2 shows an example of a parse tree for a walking sequence. The top-down step results in a compact parse tree for an action video. By comparing Fig. 2(a4) with (b), it can be seen that the noisy rule instances (e.g. those from background) are removed such that the signal-to-noise ratio is improved.

In the following section, we introduce the action classifier training and recognition method based on obtained parse trees.

4.3 Recognizing Actions

Base on a parse tree representation, a conventional way to compute the action class label $\mathcal{L}$ of an input video O is

$$\begin{aligned} \mathcal{L}^* &= \arg\max_{\mathcal{L}} p(\mathcal{L}|O) = \arg\max_{\mathcal{L}} \sum_{\mathbf{pt}} p(\mathcal{L}, \mathbf{pt}|O) \\ &\propto \arg\max_{\mathcal{L}} \sum_{\mathbf{pt}} p(O|\mathbf{pt}) \cdot p(\mathbf{pt}, \mathcal{L}), \end{aligned} \tag{37}$$

which means we need to sum over all possible parse trees. This is computationally intractable. Additionally, in this project the production rules are mined/learned from the discriminative configurations of different classes of actions for the purpose of action recognition. Thus, the learned parse trees are not fully generative, they only contain discriminative action component structures. It is not appropriated to evaluate action videos in a pure generative manner. Therefore, we propose a method that leverages the grammatical representation's strong expressive power and the discriminative model's potent differentiation power and computational efficiency to recognize actions. The details of the recognition method are introduced as follows.

The recognition model exploits the hierarchical structure of an action parse tree. The production rules at different layers of a parse tree represent discriminative action structures

(features) at different spatiotemporal scales. At each layer, a discriminative score for each action class is derived from the production rules of that layer. These scores should be weighted differently in recognizing actions because different compositional structures at different scales have different discriminative powers. Consequently, we adopt a meta-learning strategy (Vilalta and Drissi 2002) to combine the discriminative power at each layer of the parse tree to recognize actions. More specifically, the discriminative scores derived from the production rules at each layer of a parse tree is considered as base-learners, and we adopt a logistic regression model as the *meta-learner* to combine the base-learners. Thus, based on a parse tree of an action type $\mathcal{L}$ built from a video O, the probability of the action belonging to $\mathcal{L}$ is

$$y_{\mathcal{L}}(O) = \frac{1}{1+\exp(-\sum_{l=0}^{L} \omega_l^{\mathcal{L}} \cdot \tau(h^l_{\mathbf{pt}^{\mathcal{L}}_O}))} \tag{38}$$

where $\tau(h^l_{\mathbf{pt}^{\mathcal{L}}_O})$ is the total discriminative score of Layer-l of the parse tree, and $\omega_l^{\mathcal{L}}$ is the regression coefficient.

When $l > 0$, $\tau(h^l_{\mathbf{pt}^{\mathcal{L}}_O})$ is computed by

$$\tau\left(h^l_{\mathbf{pt}^{\mathcal{L}}_O}\right) = \sum_i \tau(\gamma^l_{i,\mathbf{pt}^{\mathcal{L}}_O}) \tag{39}$$

where $\gamma^l_{i,\mathbf{pt}^{\mathcal{L}}_O}$ is the rule instance corresponding to the ith production rule at Layer-l of the parse tree. The discriminative score of a production rule is computed using (36).

$\tau(h^0_{\mathbf{pt}^{\mathcal{L}}_O})$ returns the discriminative score of the STIP-map layer output by a SVM classifier. For each action class $\mathcal{L}$, we train a SVM classifier on the occurrence histograms of the ST interest point in the training videos, and use the SVM classifier output probability as the discriminative score.

The regression coefficients $\{\omega_l^{\mathcal{L}}\}_{l=0}^{L}$ are learned with labeled data, i.e. we set $y_{\mathcal{L}}(O) = 1$, if O contains an action of type $\mathcal{L}$; Otherwise, $y_{\mathcal{L}}(O) = 0$. The best-fit coefficient set is computed by the Newton-Raphson method.

Finally, the class label of the action in an input video is determined by

$$\mathcal{L}^* = \arg\max_{\mathcal{L}} y_{\mathcal{L}}(O). \tag{40}$$

5 Experiment Results

To test the proposed approach, we do extensive experiments on three public datasets, the KTH human action dataset (Schuldt et al. 2004), the Hollywood human action (HOHA) dataset (Laptev et al. 2008), and the UCF Sport action dataset (Rodriguez et al. 2008). The proposed method aims to discover high-order statistics of the low-level features by mining hierarchical feature compositions for the purpose of action recognition. The existing bag-of-words method can be seen as the base layer of the proposed approach. In the experiments, we show that by generally adding high-order compositional features, the recognition performance can be largely improved. Therefore, we compare with the baseline approaches Schuldt et al. (2004) and Dollár et al. (2005) by adopting their own features into our framework. The two baseline methods use SVM (Schuldt et al. 2004 and Dollár et al. 2005) and KNN (Dollár et al. 2005) to classify actions. As observed from Dollár et al. (2005), the results by SVM classifier are slightly better than those using KNN classifier. Hence, we adopt the SVM classifier with χ^2-kernel (Laptev et al. 2008) in both the baseline methods for implementation convenience. The χ^2-kernel is defined as

$$K(H_i, H_j) = \exp\left(-\frac{1}{A}\sum_k \frac{(H_{i,k}-H_{j,k})^2}{H_{i,k}+H_{j,k}}\right) \tag{41}$$

where H_i and H_j are two feature vectors, $H_{i,k}$ and $H_{j,k}$ are the kth entry of the two feature vectors. A is the mean value of pairwise χ^2-distances between all training feature vectors. In the remaining of this section, we first introduce the ST features used in our experiment, followed by a study of the action recognition performance on the three action datasets. Then, we present the quantitative evaluations of the parameters used in our approach. At last, the computational complexity and some extensions of the action grammar model are shown.

5.1 Spatiotemporal Interest Points

In our experiments, we adopt two types of popular spatiotemporal interest point detectors, Dollár's periodical point detector (Dollár et al. 2005) and Laptev's ST corner detector (Schuldt et al. 2004), to construct the STIP-maps of the action videos. In our experiments, we use their STIP detectors released online.[1, 2]

The periodical point detector extracts ST interest point features at four spatial scales ($\sigma = 2, 4, 6, 8$) and two temporal scales ($\nu = 4, 8$). And the ST corner detector extracts features at six spatial scales ($\sigma = 4, 8, 16, 32, 64, 128$) and two temporal scales ($\nu = 2, 4$). We utilize the K-means method to cluster the extracted feature vectors into 4000 clusters.

5.2 Experiments on Action Recognition

In this section, we study the action recognition performance of the proposed model on the three datasets.

[1] http://vision.ucsd.edu/~pdollar/toolbox/doc/index.html.

[2] http://www.irisa.fr/vista/Equipe/People/Laptev/download.html#stip.

Fig. 8 Sample frames of (**a**) the KTH human motion dataset. (**b**) The INRIA Hollywood human action dataset. (**c**) The UCF Sport action dataset

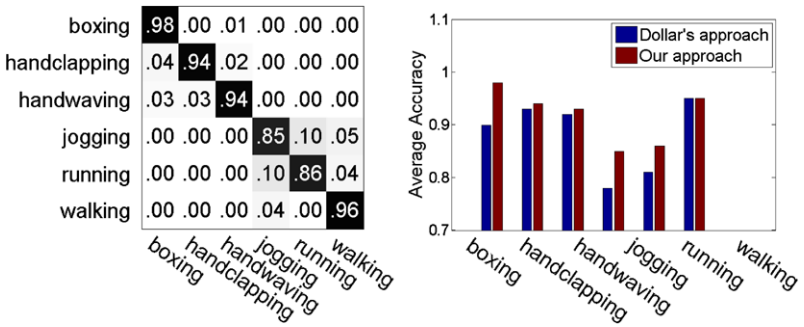

Fig. 9 Experiment results on the KTH dataset. *The left* is the confusion matrix of the recognition results by our approach based on Dollár's method (Dollár et al. 2005). *The right* figure compares the recognition accuracies of our method and Dollár's method for each action type

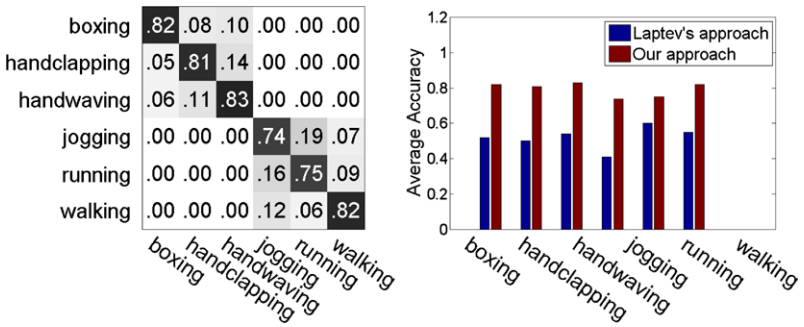

Fig. 10 Experiment results on the KTH dataset. *The left* is the confusion matrix of the recognition results by our approach based on Laptev's method (Schuldt et al. 2004). *The right* figure compares the recognition accuracies of our method and Laptev's method for each action type

5.2.1 Results on the KTH Human Action dataset

The KTH human action dataset contains twenty-five people performing six types of actions, "boxing", "waving", "handclapping", "jogging", "running" and "walking". For each person the actions were captured under four different environments with variations in scales, illuminations and camera motions, but all the videos were shot with simple backgrounds. Each type of actions contains about 96–99 available sequences. Some examples of the dataset are shown in Fig. 8(a).

In the experiment, we use the cross-validation scheme to train the models and test their performance. The video sequences are divided w.r.t the subjects into a training set and a testing set. In our experiment, each time, we randomly select 16 people' video sequences from each action type as the training set and use the rest 9 people' videos for testing. The height of the parse tree is set to 4. Twenty rounds of cross validation are performed and the average testing accuracy is shown in Figs. 9 and 10. In both figures, the left figure shows the action recognition confusion matrix of our approach, and the right one compares the recognition accuracy of the proposed approach with one of the baseline approaches. As can be seen, using Dollár's interest point detector and compared with Dollár's approach (Dollár et al. 2005), our approach increases the average recognition accuracy from 88.2% to 92.5%. Using the Laptev ST corner detector and compared with Laptev's approach (Schuldt et al. 2004), the

Table 2 Comparison of action recognition accuracies between different approaches on the three datasets

Method	Feature type	Recognition method	Accuracy (%)		
			KTH	HOHA	UCF
Dollár et al. (2005)	Periodical STIP	SVM	88.2	18.6	56.4
Niebles et al. (2008)	Periodical STIP	LDA	81.5	–	–
Nowozin et al. (2007)	Periodical STIP	Subsequence boosting	84.7	–	–
Liu et al. (2009)	PMI-based periodical STIP	SVM	92.3	–	–
Schuldt et al. (2004)	3D Harris STIP	SVM	52.2	19.5	46.7
Laptev et al. (2008)	3D Harris STIP	SVM	91.8	38.4	–
Wong et al. (Wong and Cipolla 2007)	STIP detected by NNMF	SVM	80.1	–	–
Rapantzikos et al. (2009)	Saliency-based STIP	KNN	88.3	33.6	–
Rodriguez et al. (2008)	Action MACH filters	Template matching	88.7	–	69.1
Yao et al. (Yao and Zhu 2009)	Gabor filters, optical flow	Template matching	87.8	–	–
Wang et al. (Wang and Mori 2009)	Optical flow	MMHCRF	92.5	–	–
Schindler et al. (Schindler and Gool 2008)	Gabor filters, optical flow	SVM	90.9	–	–
Sun et al. (2009)	SIFT-based trajectory	Multi-kernel SVM	–	47.1	–
Our method	STIP	Aggregated emerging pattern score	92.5	39.5	68.3

average recognition accuracy increases about 20.5%. The results confirm that our approach enhances the performance of the bag-of-video-words methods significantly. However, it can be seen that the misclassification rates between the pair of "hand-clapping" and "hand-waving", and the triple of "walking", "jogging" and "running" are still high because of their similarity. It is encouraging that by mining emerging patterns between them, we still get considerable improvement over the baseline approaches.

We also list the action recognition performances of many state-of-the-art methods in Table 2, where the results in the table are copied from their papers except Dollár's method (Dollár et al. 2005) and Schuldt's method (Schuldt et al. 2004) which are implemented by ourselves. Note that some methods use different representations and different classification engines. For example, the method in Liu et al. (2009) characterizes STIPs using MMI features and clusters the STIPs by exploring the local geometric structure of STIP feature manifold. Niebles and Li (Niebles et al. 2008) cluster the STIPs using the latent topic model. These clustering approaches associate the STIP clusters with semantic (action type) information. However, such information is not used in the interest point clustering step of the proposed method. Also, some methods use other types of representations, e.g. the methods in Yao and Zhu (2009) and Rodriguez et al. (2008) use action templates trained from annotated action videos for action detection. Wang and Mori (2009) adopt optical flow patches to represent actions, and these patches are sampled from the tracked human body. The method by Sun et al. (2009) utilize SIFT-based trajectories as the basic features. On the contrary, the proposed approach requests neither tracking, nor elaborated annotation in the training data. But, as Table 2 shows, our method still achieves a comparable performance to the other existing approaches.

5.2.2 *Results on Hollywood Human Action (HOHA) Dataset*

The Hollywood Human Action dataset (Laptev et al. 2008) is collected from thirty-two movies, e.g. "American Beauty", "Big Fish", "Forest Gump". It contains eight types of actions, "AnswerPhone", "GetOutCar", "HandShake", "HugPerson", "Kiss", "SitDown", "SitUp", and "StandUp". Some sample frames are show in Fig. 8(b). The dataset provides two types of annotations, the "automatic annotation" and the "clean annotation". The "automatic annotation" labels the clips using the scripts of the movie and the "clean annotation" labels the clips manually. In the experiments, we use data from the "clean annotation", which contains 228 training clips and 225 testing clips (we remove some short clips less than 5 frames). The layer number of the action grammar model is set to 4. The recognition results are shown in Figs. 11 and 12. The average recognition accuracies of the baseline methods using Dollár's and Laptev's ST interest point detector are 18.6% and 19.5% respectively. Using the corresponding types of interest points, the average recognition accuracies of the proposed approach are improved to 37.6% and 30.0% respectively.

5.2.3 *Results on UCF Sport Action Dataset*

The UCF sport action dataset (Rodriguez et al. 2008) is collected from broadcast television channels such as the

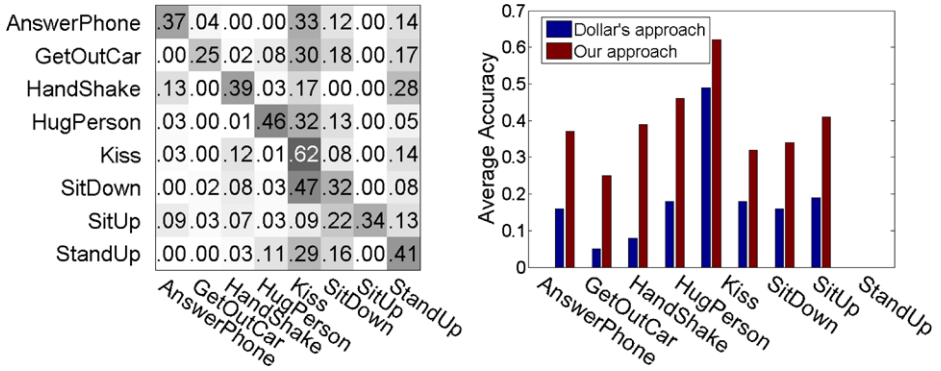

Fig. 11 Experiment results on the INRIA Hollywood human action dataset. *The left* is the confusion matrix of the recognition results by our approach based on Dollár's method (Dollár et al. 2005). *The right* figure compares the recognition accuracies of our method and Dollár's method for each action type

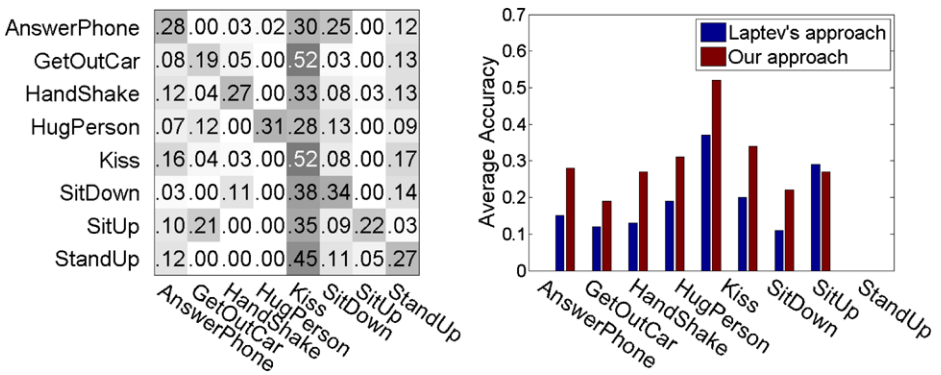

Fig. 12 Experiment results on the INRIA Hollywood human action dataset. *The left* is the confusion matrix of the recognition results by our approach based on Laptev's method (Schuldt et al. 2004). *The right* figure compares the recognition accuracies of our method and Laptev's method for each action type

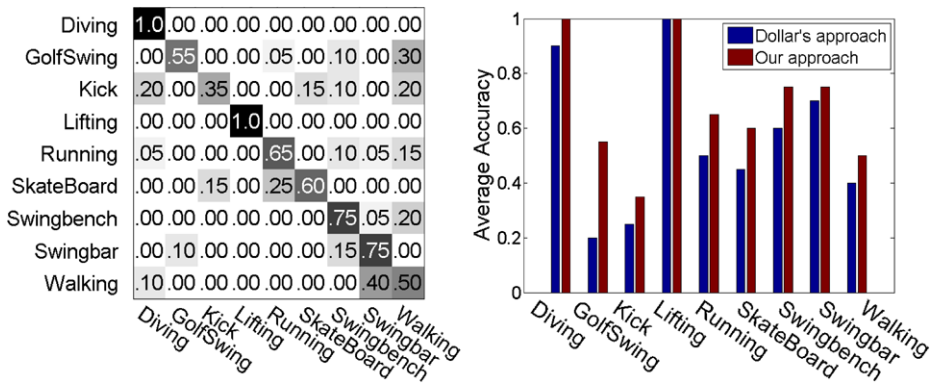

Fig. 13 Experiment results on the UCF sport action dataset. *The left* is the confusion matrix of the recognition results by our approach based on Dollár's method (Dollár et al. 2005). *The right* figure compares the recognition accuracies of our method and Dollár's method for each action type

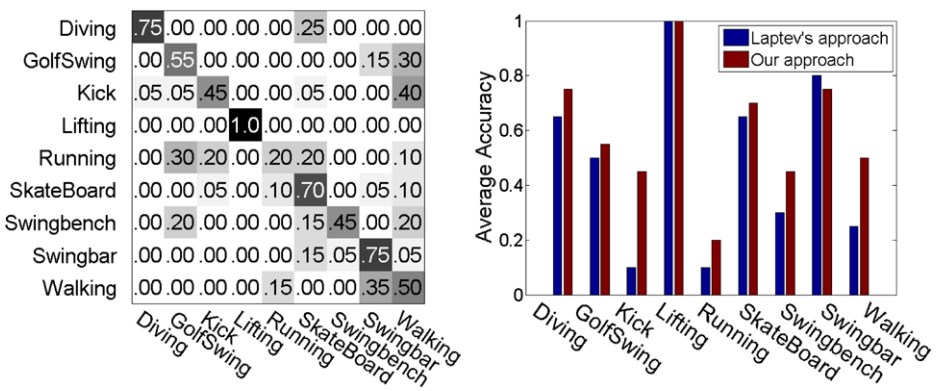

Fig. 14 Experiment results on the UCF sport action dataset. *The left* is the confusion matrix of the recognition results by our approach based on Laptev's method (Schuldt et al. 2004). *The right* figure compares the recognition accuracies of our method and Laptev's method for each action type

BBC and ESPN. The dataset contains 150 video sequences covering nine types of actions including "diving", "golf swinging", "kicking", "lifting", "horseback riding", "running", "skating", "swinging", and "walking". These videos exhibit large variations in illumination, backgrounds, scales and viewpoints. Some examples of the dataset are shown in Fig. 8(c).

Table 3 Average recognition accuracy changes with the variations of the layer number of the proposed hierarchical action grammar model. (Layer number = 1 corresponding to the baseline method.) The experiments are conducted on the three datasets based on Dollár's STIP detector

Dataset	Layer numbers			
	1	2	3	4
KTH	0.882	0.897	0.919	0.925
HOHA	0.186	0.266	0.328	0.376
UCF	0.564	0.623	0.654	0.683

Table 4 Distributions of the three types of production rules in the learned rule set and parse trees of the KTH and HOHA datasets

Dataset		Strong relation	Weak relation	Stochastic relation
KTH	In learned rule set	1%	2%	97%
	In parse tree	12%	8%	80%
HOHA	In learned rule set	3%	3%	94%
	Parse tree	18%	15%	67%

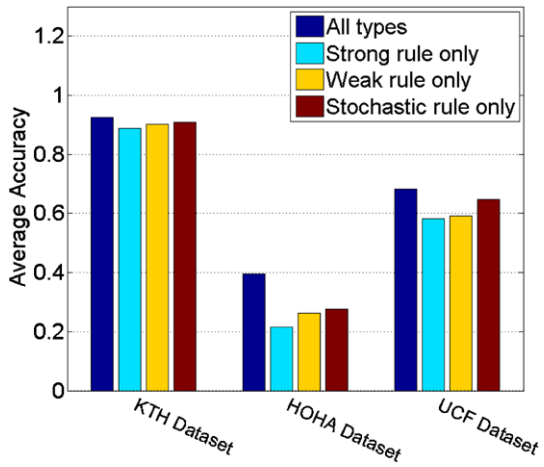

Fig. 16 Comparison of recognition accuracies under different combinations of production rule types used in our method. The interest points are detected using Dollár's STIP detector

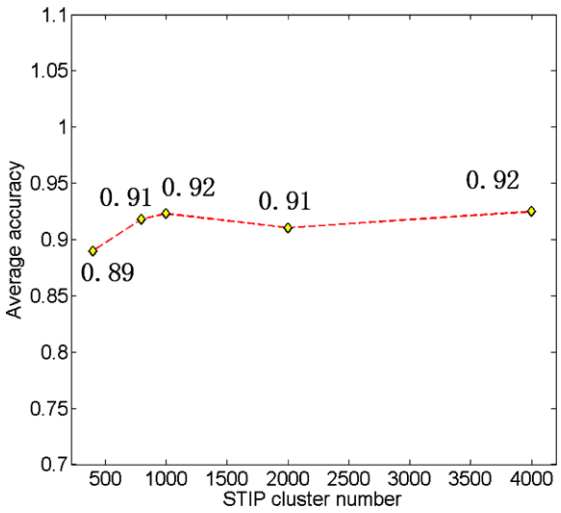

Fig. 17 Comparison of recognition accuracies with different numbers of STIP clusters. The interest points are detected using Dollár's STIP detector

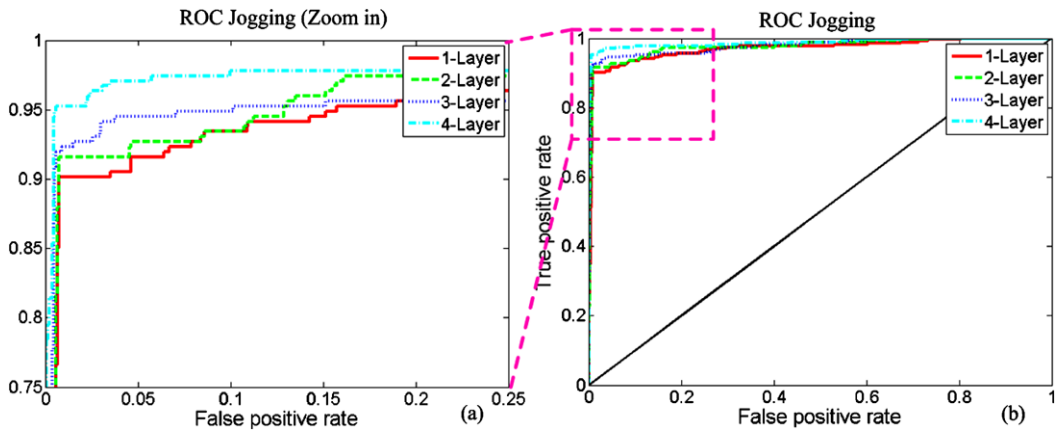

Fig. 15 ROC curves illustrating the action recognition performance change with different number of layers. The ROC curves are obtained by thresholding on the action probability defined in (38), and are plotted for action 'jogging' (in KTH dataset) in (**b**). Its zoomed-in version is shown in (**a**)

The number of videos in this dataset is very limited. There are only about fourteen sequences for each action. Each video sequence contains about 90 frames. So we use a 3-Nearest-Neighbor classifier to recognize actions in the baseline methods. The distance measure between feature vectors is χ^2 distance. And we cluster the STIPs into 200 clusters which gives the best result in our experiments. We use the cross-validation method to train the models and test their performance. Each time, for each action we leave one sequence out for testing and use the rest for training. Twenty rounds of cross-validations are performed, and the average testing results are shown in Figs. 13 and 14. It can be seen that our method outperforms the baseline approaches in most cases. Using Dollár's interest point detector, the average recognition accuracy of the corresponding baseline approach is 56.4%, and the proposed method's is 68.3%. Using Laptev's ST corner detector, the average recognition accuracies of the corresponding baseline method and our method are 46.7% and 59.4% respectively.

5.3 Evaluation of Model Parameters

In this section, we study the performance variation under different parameter settings of the action model.

The Number of Layers We do experiments on different layer numbers of the action parse tree to demonstrate its influence on the recognition performance. The results are shown in Table 3 and Fig. 15. In Fig. 15, we show the ROC curve changes for the action 'jogging' in KTH dataset. (The other actions exhibit similar results.) It can be seen

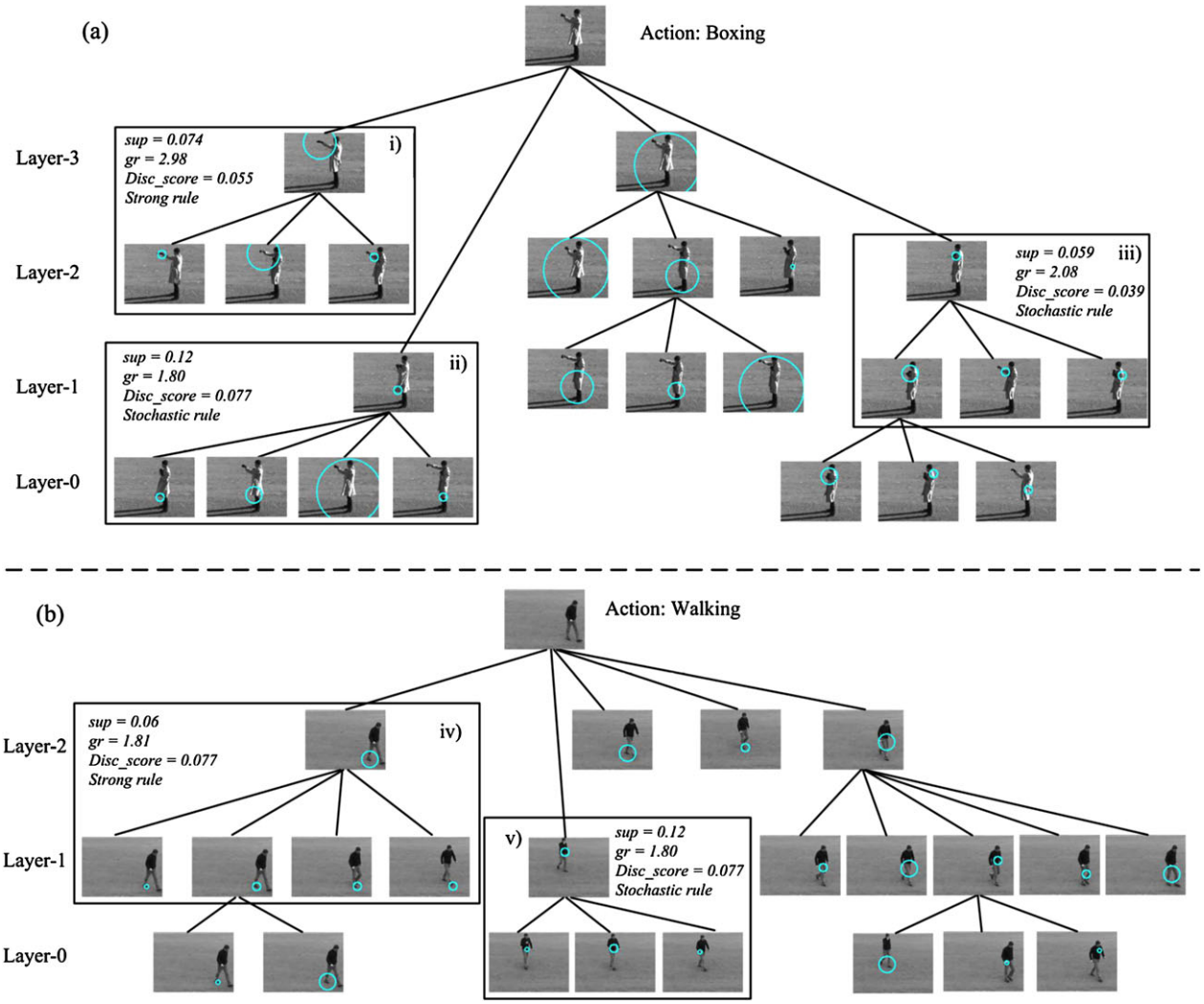

Fig. 18 Computed parse trees of the actions "boxing" and "walking" from KTH dataset. Each node-splitting corresponds to a production rule. The *cyan circles* represent the centers of the components and their sizes correspond to the scales of the components. (**i**)–(**v**) show the attributes of some production rules including their rule types, support rates, growth ratios and discriminative scores

Fig. 19 Computed parse trees of the actions "AnswerPhone" and "GetOutCar" from HOHA dataset. Each node-splitting corresponds to a production rule. The *cyan circles* represent the centers of the components. (**i**)–(**iv**) show the attributes of some production rules including their rule types, support rates, growth ratios and discriminative scores

that as the layer number increases, both the average recognition accuracy and the ROC curves improve. This is mainly because that configurations at a higher layer correspond to action structures at a larger scale, which are more discriminative than the lower ones. However, the parse tree layer number cannot increase without limitation. This is because up to certain level there exist no more common configurations among the same type of action instances. As a result, we cannot summarize/mine any more production rules.

The Number of Rule Types In Fig. 16 the experimental result shows that using all the three types of rules, the action recognition accuracy outperforms the one only using a single type of rules. In Table 4 we also list the occurrence proportions of rules of different relation types in the learned rule sets and the parse trees of the training videos. It can be seen that, in the production rule learning phase, most of the learned rules are stochastic rules. However, when building the action parse trees, the portions of strong rule and weak rule increase significantly. This is largely because the rules with more constraints usually have higher discriminative scores computed by (36), so more of them are kept in the top-down ambiguous rule pruning step when building parse trees.

The Number of STIP Clusters As shown in Fig. 17, the results on KTH Dataset demonstrate that for different numbers of the interest point clusters, the average recognition accuracies do not vary too much, i.e. our method is robust to the variation of the number of interest point clusters.

Table 5 The numbers of production rules at each layer of the parses trees in Figs. 18 and 19

Action type	Layer-0	Layer-1	Layer-2	Layer-3
Boxing	145	48	10	2
Walking	133	42	5	0
AnswerPhone	2118	717	113	33
GetOutCar	632	203	12	3

Table 6 Distributions of the three types of production rules learned to differentiate the action pairs "boxing" vs. "walking" and "walking" vs. "running"

Action types	Strong rule	Weak rule	Stochastic rule
Boxing vs. walking	0%	0%	100%
Running vs. walking	17%	12%	61%

5.4 Analysis of the Action Parse Trees

Figure 18 and 19 show the constructed parse trees of action instances of "Boxing", "Walking", "AnswerPhone" and "GetOutCar" from the KTH human action dataset and HOHA dataset. Since the full-size parse trees are too big to fit in the page with a satisfied resolution, we only show part of them. The numbers of production rules in each layer of the parse trees are shown in Table 5. When the layer number becomes higher, the number of detected rule instances decreases. Note that sometimes if an action appears in the video very shortly or its action style (or similar viewpoint) is not contained in the training data, our method will not build a full-size action parse tree. Figure 18(b) shows an example without Layer-3. We also show some rule attributes in Figs. 18 and 19, including their layer numbers, support/growth ratios and discriminative scores. From these highlighted rule instances, we can see that the mined production rules capture some meaningful configurations of the actions.

To further demonstrate the importance/necessity of identifying the three types of rules, we learn production rules from action pairs, "boxing" vs. "walking" and "walking" vs. "running". The training set contains only these related types of action videos from the KTH dataset. Table 6 shows the proportions of the three types of production rules used in constructing the parse trees. It is interesting to observe that for action pair "boxing" vs. "walking", only using stochastic rules is enough to differentiate them, as the two actions are quite different from each other. Whereas, for the action pair "walking" vs. "running", since they are similar in action components, production rules with more ST constraints are automatically mined to differentiate them.

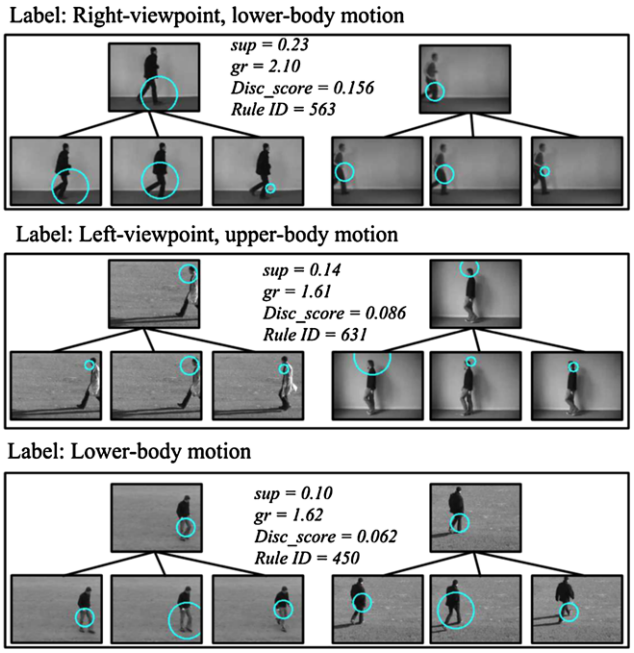

Fig. 20 Identified production rules and their corresponding semantic structures

5.5 Results Beyond Recognition

To illustrate the competence of the proposed grammar model, besides recognition, we also apply it to the semantic structure discovery and foreground localization.

Semantic Structure Discovery In this experiment, we highlight the rule instances of parse trees occurred in action videos and discover the semantic information they convey of the actions. Figure 20 shows some identified typical production rule instances and their associated semantic labels, e.g. body parts and view points.

Foreground Localization We also use the action parse tree to localize the foreground object. The reason behind is that after the top-down pruning of the parse trees, the rule instances left are mostly related to the motion of the foreground objects. In Fig. 21, at each frame, we use a rectangle to enclose all the leaf nodes (the ST interest points) of an action parse tree. It can be seen that the bounding boxes of the leaf nodes well localize the foreground objects.

5.6 Computational Expense

Our experiment is conducted on an Intel Core2 Duo 3.20 GHz CPU, 3.0G RAM PC, Windows XP system. The code is implemented in C++ using Visual Studio 2005. Table 7 lists the computational expense for production rule learning and recognition on the three datasets. Because the three datasets contain different numbers of video sequences,

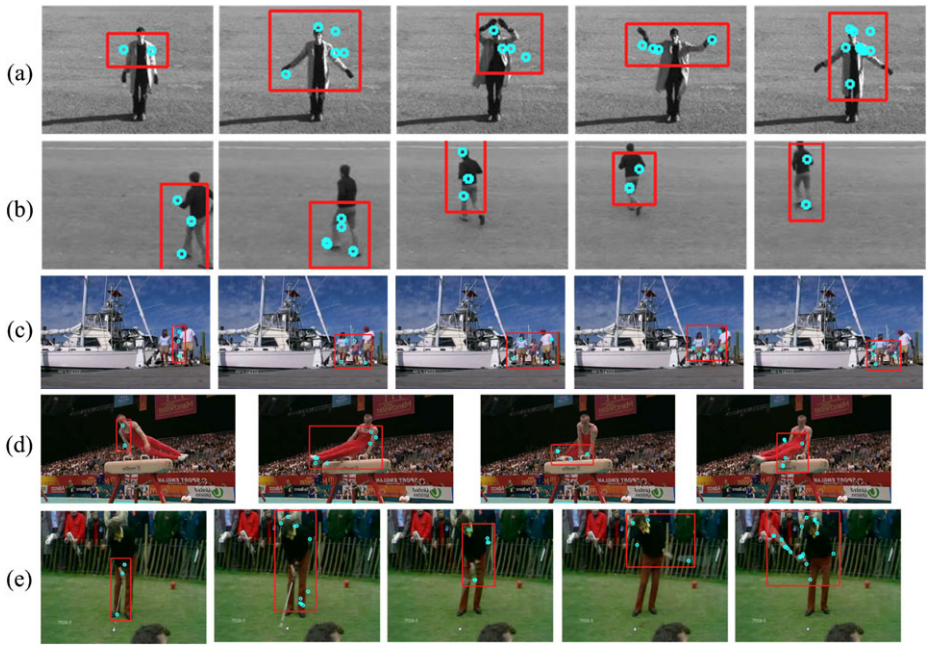

Fig. 21 Localize the foreground objects using the STIPs associated with the leaf nodes (represented as *cyan circles*) of the video's action parse trees. Note that the *cyan circles* are shown with the same size ignoring their corresponding STIPs' scales to better illustrate their locations. (**a**), (**b**) are video frames from KTH dataset. (**c**), (**d**), (**e**) are video frames from UCF sport action dataset

Table 7 Computational expense of production rule learning and recognition on the three datasets

Dataset	Frame number	The number of interest point	Learning time	Recognition time
KTH	24–251	17–708	40 min	3 sec
UCF	50–110	100–900	30 min	3 sec
INRIA	14–880	10–20620	55 min	10 sec

and even the video sequences from the same action type exhibit large variations, the learning and recognition times differ largely, and we report the average computation time here. The interest points are detected by Laptev's detector.

6 Conclusion and Future Work

This paper proposes a layered-grammar model to represent actions for recognition. Given the fact that actions are compositional, we identified three major action styles which are characterized by different rigidities in organizing the action components. The three action styles are encoded into a unified representation in a form of attributed grammar rules. We propose the *relation pursuit* to learn grammar rules from action videos. The learned rules are statistically significant and discriminative. Because it can be computationally expensive to build a global optimal parse tree that maximizes the posterior probability in (37), instead, we take an alternative approach to constructing action parse trees by combining a bottom-up multi-hypothesis rule detection step and a top-down ambiguous rule pruning step. Although this tree construction method maximizes the total discriminative scores of action parse trees by selecting non-overlapping hierarchical discriminative structures at the tree construction procedure, it does not generate the global optimal generative parse trees that maximize (37). The resulting parse trees are compact and capable of identifying the discriminative structures of the actions. Based on the parse tree representation, we design a discriminative action recognition method which are comparable to existing method.

In the future, we shall investigate the following issues based on the proposed framework. (1) The current proposed approach uses ST interest points as the bottom layer representation. We shall investigate other types of low level features, e.g. the feature introduced in Ke et al. (2005), Yao and Zhu (2009), and test their influence on the final recognition results. (2) We shall study more complex scenarios, e.g. complex actions and multi-action interactions so as to extend our model from action recognition to long-temporal-range event analysis.

Acknowledgements This work was supported by NSFC grants 60872077 and National Basic Research Program of China (973 Program) 2009CB320904.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proc. int'l conf. very large data bases* (pp. 487–499).

Alhammady, H., & Ramamohanarao, K. (2006). Using emerging patterns to construct weighted decision trees. *IEEE Transactions on Knowledge and Data Engineering*, *18*(7), 865–876.

Allen, J. F., & Ferguson, G. (1994). Actions and events in interval temporal logic. *Journal of Logic and Computation*, *4*(5), 531–579.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proc. IEEE conf. computer vision and pattern recognition* (Vol. 1, pp. 886–893).

Dollár, P., Rabaud, V., Cottrell, G., & Belongie, S. (2005). Behavior recognition via sparse spatio-temporal features. In *Proc. IEEE int'l workshop on PETS* (pp. 65–72).

Dong, G., & Li, J. (2004). Efficient mining of emerging patterns: discovering trends and differences. In *Proc. ACM SIGKDD int'l conf. knowledge discovery and data mining* (pp. 43–52).

Dong, G., Zhang, X., Wong, L., & Li, J. (1999). CAEP: classification by aggregating emerging patterns. *Discovery Science*, *1721*, 737–747.

Gilbert, A., Illingworth, J., & Bowden, R. (2008). Scale invariant action recognition using compound features mined from dense spatio-temporal corners. In *Proc. European conf. computer vision* (pp. 222–233).

Harris, C., & Stephens, M. (1988). A combined corner and edge detector. In *Proc. Alvey vision conference* (pp. 147–152).

Ivanov, Y. A., & Bobick, A. F. (2000). Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(8), 852–872.

Joo, S. W., & Chellappa, R. (2006). Recognition of multi-object events using attribute grammars. In *Proc. int'l conf. image processing* (pp. 2897–2900).

Ke, Y., Sukthankar, R., & Hebert, M. (2005). Efficient visual event detection using volumetric features. In *Proc. int'l conf. computer vision* (pp. 166–173).

Laptev, I., & Lindeberg, T. (2003). Space-time interest points. In *Proc. int'l conf. computer vision* (pp. 432–439).

Laptev, I., Marszalek, M., Schmid, C., & Rozeneld, B. (2008). Learning realistic human actions from movies. In *Proc. int'l conf. computer vision and pattern recognition*.

Leibe, B., Leonardis, A., & Schiele, B. (2008). Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision*, *77*, 259–289.

Leonardis, A., Gupta, A., & Bajcsy, R. (1995). Segmentation of range images as the search for geometric parametric models. *International Journal of Computer Vision*, *14*, 253–277.

Lin, L., Gong, H., Li, L., & Wang, L. (2009). Semantic event representation and recognition using syntactic attribute graph grammar. *Pattern Recognition Letters*, *30*, 180–186.

Liu, J., & Shah, M. (2008). Learning human actions via information maximization. In *Proc. int'l conf. computer vision and pattern recognition*.

Liu, J., Yang, Y., & Shah, M. (2009). Learning semantic visual vocabularies using diffusion distance. In *Proc. IEEE int'l conf. computer vision and pattern recognition*.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110.

Niebles, J. C., Wang, H., & Fei-Fei, L. (2008). Unsupervised learning of human action categories using spatial-temporal words. *International Journal of Computer Vision*, *79*(3), 299–318.

Nowozin, S., Bakir, G., & Tsuda, K. (2007). Discriminative subsequence mining for action recognition. In *Proc. int'l conf. computer vision*.

Quack, T., Ferrari, V., Leibe, B., & Gool, L. V. (2007). Efficient mining of frequent and distinctive feature configurations. In *Proc. ICCV*.

Quelhas, P., Monay, F., Odobez, J., Perez, D., & Tuytelaars, T. (2007). A thousand words in a scene. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(9), 1575–1589.

Rapantzikos, K., Avrithis, Y., & Kollias, S. (2009). Dense saliency-based spatiotemporal feature points for action recognition. In *Proc. IEEE int'l conf. computer vision and pattern recognition* (pp. 1–8).

Rodriguez, M. D., Ahmed, J., & Shah, M. (2008). Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *Proc. int'l conf. computer vision and pattern recognition*.

Ryoo, M. S., & Aggarwal, J. K. (2009). Semantic representation and recognition of continued and recursive human activities. *International Journal of Computer Vision*, *82*, 1–24.

Schindler, K., & Gool, L. (2008). Action snippets: how many frames does human action recognition require? In *Proc. IEEE conf. computer vision and pattern recognition*.

Schnitzspan, P., Fritz, M., Roth, S., & Schiele, B. (2009). Discriminative structure learning of hierarchical representations for object detection. In *Proc. IEEE conf. computer vision and pattern recognition* (pp. 1–8).

Schuldt, C., Laptev, I., & Caputo, B. (2004). Recognizing human actions: a local SVM approach. In *Proc. int'l conf. pattern recognition* (pp. 32–36).

Sivic, J., & Zisserman, A. (2004). Video data mining using configurations of viewpoint invariant regions. In *Proc. int'l conf. computer vision and pattern recognition*.

Sun, J., Wu, X., Yan, S., Cheong, L., Chua, T., & Li, J. (2009). Hierarchical spatio-temporal context modeling for action recognition. In *Proc. IEEE conf. computer vision and pattern recognition* (pp. 1–8).

Vilalta, R., & Drissi, Y. (2002). A perspective view and survey of meta-learning. *Artificial Intelligence Review*, *18*, 77–95.

Wang, Y., & Mori, G. (2009). Max-margin hidden conditional random fields for human action recognition. In *Proc. IEEE conf. computer vision and pattern recognition*.

Wong, S. F., & Cipolla, R. (2007). Extracting spatiotemporal interest points using global information. In *Proc. IEEE int'l conf. computer vision*.

Yao, B., & Zhu, S. (2009). Learning deformable action templates from cluttered videos. In *Proc. int'l conf. computer vision*.