Unsupervised Learning of Event AND-OR Grammar and Semantics from Video

Zhangzhang Si[a], Mingtao Pei[b], Benjamin Yao[a], Song-Chun Zhu[a]
[a]Department of Statistics, University of California, Los Angeles
[b]Lab of Intelligent Info. Tech., Beijing Institute of Technology
zzsi@stat.ucla.edu, peimt@bit.edu.cn, {zyyao,sczhu}@stat.ucla.edu

Abstract

We study the problem of automatically learning event AND-OR grammar from videos of a certain environment, e.g. *an office where students conduct daily activities. We propose to learn the event grammar under the information projection and minimum description length principles in a coherent probabilistic framework, without manual supervision about what events happen and when they happen. Firstly a predefined set of unary and binary relations are detected for each video frame:* e.g. *agent's position, pose and interaction with environment. Then their co-occurrences are clustered into a dictionary of simple and transient atomic actions. Recursively these actions are grouped into longer and complexer events, resulting in a stochastic event grammar. By modeling time constraints of successive events, the learned grammar becomes context-sensitive. We introduce a new dataset of surveillance-style video in office, and present a prototype system for video analysis integrating bottom-up detection, grammatical learning and parsing. On this dataset, the learning algorithm is able to automatically discover important events and construct a stochastic grammar, which can be used to accurately parse newly observed video. The learned grammar can be used as a prior to improve the noisy bottom-up detection of atomic actions. It can also be used to infer semantics of the scene. In general, the event grammar is an efficient way for common knowledge acquisition from video.*

1. Introduction

We are interested in building a closed loop unsupervised learning framework towards the following goals: 1) unsupervised learning of the event AND-OR grammar from video, and 2) inferring scene semantics of the environment by event analysis. Both are motivated by a general goal of automatic knowledge acquisition from video data. The acquired common sense knowledge provides a visually grounded representation for goal-based cognitive reasoning, which usually operates on abstract logical formulas.

Event analysis has gone a long way from modeling transient or periodic action [10, 3], to longer events [13, 6, 8], achieving promising recognition performance. However, most of the above work train event models for a predefined set of event classes. In contrast, an unsupervised learning algorithm automatically generates richer event classes and also reduces tedious manual labeling, thus providing more scalability for knowledge acquisition systems.

Our work is also inspired by recent progress in unsupervised learning and data mining [14, 5] as well as grammatical learning and inference [2, 4, 15] on video data. [2, 4] address the problem of event recognition using a predefined stochastic context free grammar, but do not show how to learn the grammar. For grammar leaning, our strategy is most similar to Zhang *et al.* [15], which learns a stochastic context free grammar automatically. However, it is mainly applied to trajectory analysis of multiple agents. In contrast, we adopt a richer feature representation including interactions between agents and the environment. Also, we append a Markov model of time constraints for adjacent events, resulting in a stochastic context sensitive grammar, which was first introduced into computer vision by Zhu and Mumford in [17]. The stochastic event grammar provides an efficient representation for knowledge extracted from video.

Table 1. A list of daily activities in office.

Working on computer	Reading or writing
Making a call	Fetching water
Litering	Watching soccer match
Entering and leaving	Walking in the passageway

In our work, we deal with surveillance type of videos taken in offices (Fig.1) where students conduct daily activities (Table 1) repeatedly. As the environment is almost static except for moving chairs and doors, one can perform foreground segmentation and object tracking easily. This enables us to focus on higher level event modeling. However, even in such a controlled environment as an office, automatically learning event grammar is not easy. Firstly, due to occlusion, shadow and scale change, bottom up detection for

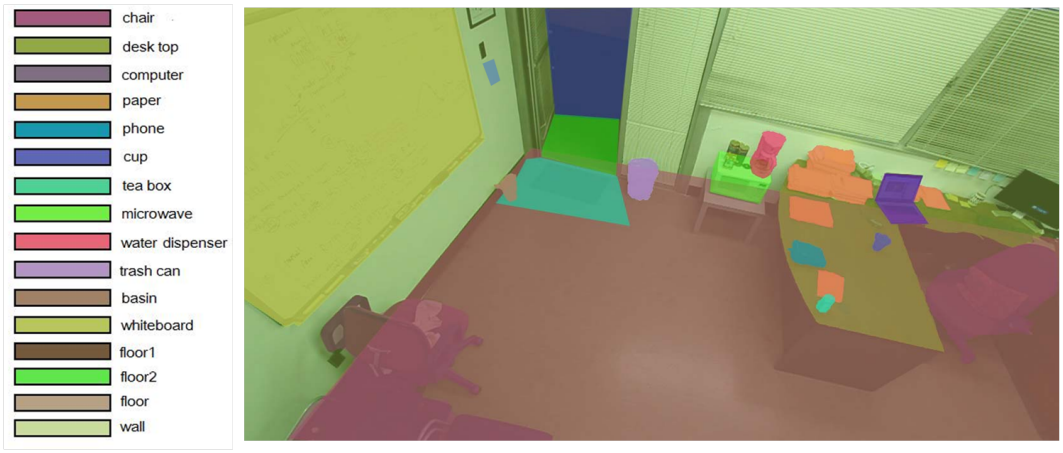

Figure 1. Semantic label map of the office scene. Manually annotated as context for agent actions.

visual attributes (*e.g.* the pose of agent) is often ambiguous. Secondly, there is large temporal deformation between instances of similar events. For example, in a "fetching water" event, the agent may choose to stay at the water dispenser for five seconds or two minutes, before he/she leaves. Simple agglomerate grouping would perform poorly under such temporal variation.

To the goal of automatic knowledge acquisition from video, we build a prototype system for unsupervised event learning with the following characteristics: 1) atomic actions are defined by spatial-temporal configurations of unary (*e.g.* agent) and binary (*e.g.* agent-environment) relations grounded in the video data; 2) longer events are defined by composition of events as production rules in a stochastic grammar; 3) scene semantics can in turn be inferred through recognition of agent actions and events. For example, small objects in a scene, such as mug and phone, can be revealed through contextual actions such as drinking and making a phone call. The unsupervised event learning takes as input the video data together with a semantic label map of the actionable regions (*e.g.* desk, chair, floor), and carries out three steps: i) grounded relations (*e.g.* agent position and pose) are detected. ii) The transient spatial-temporal patterns of grounded relations are clustered as an alphabet of transient atomic actions. The video is then encoded as a symbol sequence by atomic actions. iii) Starting from the atomic actions, shorter events are grouped into longer ones recursively under information projection and minimum description length principles. And as a result, production rules of the stochastic grammar are learned.

Dataset. We collect surveillance style video for students' daily activities in the office. Student actors are asked to repeatedly perform the daily activities listed in Table 1. In total we collect about 3 hours of video, with 2.7×10^5 frames at the resolution of 1280*720 pixels. The data is available at http://www.stat.ucla.edu/~zzsi/officelifevideo.html.

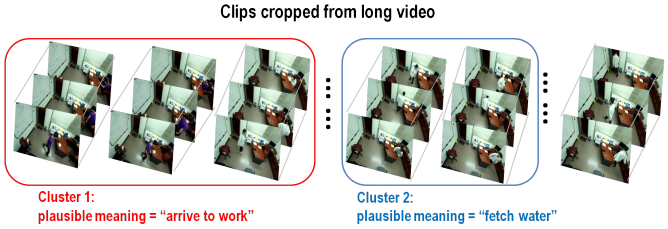

Figure 2. Clustering from clips.

Table 2. The grounded unary and binary relations of event grammar: relations directly detectable from video.

Name	Definition	Description
r_1	absent(agent)	not found in the frame
r_2	near(agent, desk)	near the desk
r_3	near(agent, board)	near the white board
r_4	near(agent, door)	near the door
r_5	near(agent, water)	near the water dispenser
r_6	near(agent, trashcan)	near the trash can
r_7	near(agent, passage)	at the passageway
r_8	touch(agent, keyboard)	typing on keyboard
r_9	touch(agent, mug)	grabbing a mug
r_{10}	touch(agent, phone)	grabbing the phone
r_{11}	bend(agent)	bending the body downwards to pick up something
r_{12}	sit(agent)	sitting on something
r_{13}	celebrate(agent)	celebrating something with two arms stretching upwards
r_{14}	stand(agent)	standing straight
r_{15}	occlude(soccer match, screen)	soccer match appearing on the computer screen

2. Detecting grounded relations

For an event grammar to work, one needs a set of grounded relations directly detectable from video. A

grounded relation is of the form $r_j(A)$ (a unary relation) or $r_j(A, B)$ (a binary relation), where r_j is a logical-valued relation (*e.g.* near, touch, appear) indexed by j, and A, B are entities (*e.g.* agent, phone). Table 2 specifies the 15 unary and binary relations (in [7] a richer set of relations are used). There are four types of relations: agent location ($r_1 \sim r_7$), agent-environment interaction ($r_8 \sim r_{10}$), agent pose ($r_{11} \sim r_{14}$) and background event (r_{15}).

Different grounded relations are detected using different methods. Firstly we use a standard background subtraction algorithm to segment moving agent and fluent changes of objects, and use a commercial surveillance system to track the detected agent. We track the bottom point of the bounding box for the agent, and cluster these two dimensional locations using k-means. We set the number of clusters k to be much larger than the number of semantically distinctive regions. Then we determine which clusters are equivalent depends on the scene label map. As a result, we obtain spatial models for regions of interest (*e.g.* door, desk, board) in Fig.4. For each newly detected agent location, we output the posterior probabilities for this location to belong to the regions of interest.

The location of the agent is computed by combining foreground segmentation and skin color detection that locates the head and hands of the agent. The real valued location is then quantized into a categorical variable by finding its nearest region of interest (*e.g.* desk, door). The agent pose is inferred by a nearest neighbor classifier using both pixels and foreground segmentation map within the estimated bounding box for the agent. An illustration of four poses using segmented foreground mask is shown in Fig. 3. The binary relations `touch(agent, keyboard)` and `touch(agent, phone)` are detected by checking whether there is enough skin color within the designated area for the laptop and phone, which are static objects in the office environment. The relation `touch(agent, mug)` is also detected using skin color, and the unique color and shape of the mug. The background relation `occlude(soccer match, screen)` is determined by checking whether there is large amount of green color occluding the laptop. Using the techniques described above, we detect grounded relations for every video frame. The detection result is organized as a spatial temporal table where each row corresponds to a time frame. Each column corresponds to a grounded relation.

Figure 3. Standing, bending, sitting and celebrating poses.

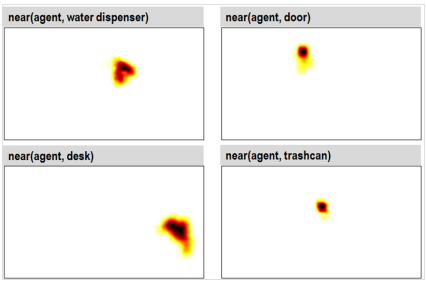

Figure 4. The spatial models for the locational binary relations. Darker region indicates a higher probability.

Table 3. Learned atomic actions.

symbolic definition	frequency	semantic
$a_1 = (r_1)$	.0392	absent
$a_2 = (r_4, r_{14})$	.0719	standing at door
$a_3 = (r_4, r_{11})$	.0065	bending at door
$a_4 = (r_7, r_{14})$	.098	walking in passageway
$a_5 = (r_2, r_{12})$	.1765	sitting at desk
$a_6 = (r_2, r_8, r_{12})$	.0915	sitting at desk, typing
$a_7 = (r_2, r_8, r_9, r_{12})$	.0065	sitting at desk, typing, grabbing mug
$a_8 = (r_2, r_9, r_{12})$	.0327	sitting at desk, grabbing mug
$a_9 = (r_2, r_8, r_{14})$	.0523	standing at desk, grabbing mug
$a_{10} = (r_7, r_{14})$	.0654	walking in passageway, grabbing mug
$a_{11} = (r_5, r_9, r_{14})$	.0523	standing at water dispenser, holding mug
$a_{12} = (r_5, r_9, r_{11})$	.0261	bending at water dispenser, holding mug
$a_{13} = (r_2, r_{10}, r_{12})$	.0131	sitting at desk, picking up phone
$a_{14} = (r_2, r_8, r_{12}, r_{15})$	.0131	sitting at desk, typing, soccer match on screen
$a_{15} = (r_2, r_{12}, r_{15})$	.0327	sitting at desk, soccer match on screen
$a_{16} = (r_2, r_{13}, r_{15})$	.0261	celebrating at desk, soccer match on screen
$a_{17} = (r_2, r_{14})$	.1242	standing at desk
$a_{18} = (r_2, r_{11})$	.0065	bending at desk
$a_{19} = (r_6, r_{14})$	.0131	standing at transhcan
$a_{20} = (r_6, r_{11})$	.0065	bending at trashcan
$a_{21} = (r_6, r_9, r_{11})$	.0261	standing at trashcan, holding mug
$a_{22} = (r_6, r_9, r_{14})$	.0131	bending at trashcan, holding mug
$a_{23} = (r_3, r_{14})$	.0065	standing at white board

3. Learning events and event grammar

3.1. Information projection

The unsupervised learning of stochastic event grammar is conducted under the information projection and minimum description length principle. In general, for vectored

data $\{\mathbf{x}_{1:J}^{(1)}, ..., \mathbf{x}_{1:J}^{(N)}\}$ we can organize them into a data matrix with N rows and J columns. N is the number of data instances, and J is the length of the vectors indicating the span in space and/or time. The goal is to find a probabilistic model $p(\mathbf{x})$ to maximally explain the data matrix. Initially, the data matrix is explained by a reference (or background) distribution $q(\mathbf{x})$ which is a simple i.i.d. Bernoulli distribution. It does not capture any spatial or temporal correlation between events. The description length using q to encode the data matrix, defined as $\sum_{i=1}^{N} -\log q(\mathbf{x}_{1:J}^{(i)})$, is often very large.

Starting from the reference distribution q we pursue a series of models by information projection to approximate the target distribution f that generates the observed data:

$$q = p_0 \rightarrow p_1 \rightarrow p_2, ..., \rightarrow p_K = p \approx f \tag{1}$$

so that the Kullback-Leibler divergence $\mathcal{K}(f, p_k)$ between the target distribution f and the model p_k decreases monotonically and the log-likelihood increases monotonically.

The model update in Eq.1 is achieved by pursuing a series of homogeneous rectangular blocks $\{\mathcal{B}_1, ..., \mathcal{B}_K\}$ in the data matrix (as illustrated in Fig.5) and representing each block by a probabilistic model. We refer to [9] for a detailed derivation of the block pursuit procedure. A rectangular block in the data matrix is a set of common components (columns) shared by a set of examples (rows). The shared components form a *template* of the block members, capturing the co-occurrence in space or time. The area of the block indicates its significance and thus the information gain, or reduction in description length. Similar approaches have also been adopted in the grammar learning of textual data [12].

The final model p_K after K steps of pursuit encodes the original sequence with a much better model capturing the co-occurrences of simpler events. p_K has a higher likelihood on the observed sequence, thus reducing the description length for data. The information gain compared with the reference model q is a three-fold summation:

$$\begin{aligned} \text{info. gain} &= \sum_{i=1}^{N} \log \frac{p_K(\mathbf{x}^{(i)}; \mathcal{B}_1, \cdots \mathcal{B}_K)}{q(\mathbf{x}^{(i)})} \\ &= \sum_{k=1}^{K} \sum_{i \in \text{rows}(\mathcal{B}_k)} \sum_{j \in \text{columns}(\mathcal{B}_k)} \log \frac{p_K(\mathbf{x}_j^{(i)} | \mathcal{B}_k)}{q(\mathbf{x}_j^{(i)})} \end{aligned} \tag{2}$$

To penalize the model complexity, we apply a constant penalty for each additional block learned. This is equivalent to imposing a Laplacian prior on the number of blocks, or the dictionary size of the learned grammar.

The above learning method can be implemented either by clustering/bi-clustering, which produces multiple blocks at the same time, or by stepwise pursuit, which produces one block at a time. The learning of event grammar is carried out into two stages. (1) Learn a set of terminal nodes as blocks on the data matrix of grounded relations. These terminal nodes account for *atomic actions* which directly specify spatial temporal configurations of grounded relations. This is done by clustering. (2) Learn non-terminal nodes as blocks on the data matrix of atomic actions, to account for longer events composed of atomic actions. This is done by step-wise pursuit.

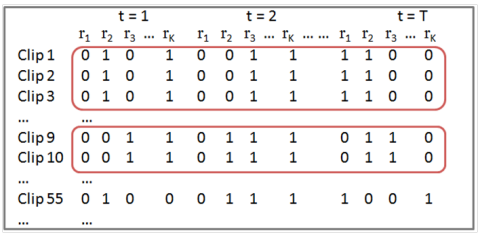

Figure 5. Pursuing homogeneous blocks from the data matrix.

3.2. Learning atomic actions

We define *atomic actions* to be simple and transient events composed spatially and temporally by grounded relations. To learn an alphabet of atomic actions, we use a temporal scanning window spanning 5 frames to collect a large number of small clips. Each 5-frame clip is described by a binary vector of detected relations:

$$\{(\mathbf{r}_{1,1}, ..., \mathbf{r}_{1,D}, ..., \mathbf{r}_{5,1}, ..., \mathbf{r}_{5,D})\}$$

where $D = 15$ is the number of grounded relations detected per frame. A k-centroids clustering is then performed on the grounded relation vectors of these 5-frame clips, using the simple Hamming distance as the metric. And a centroid of a cluster is simply determined as the grounded relation vector that has minimal distance to all the cluster members. As the timespan is very small, we can assume that the grounded relations (*e.g.* agent location, pose) stay constant during the short period. So we constrain the centroids to be stationery, *i.e.* $\mathbf{r}_{1,d} = \mathbf{r}_{2,d} = ...\mathbf{r}_{5,d}, \forall d = 1, ...D$. For each cluster, we estimate the symbol probabilities $p(\mathbf{r}_1)$, ..., $p(\mathbf{r}_{15})$ by counting the member sub-sequences of the cluster. And we represent this stochastic model by its mode (the most likely sub-sequence) denoted as $\mathbf{r}_{1:15}^{(k)}$ for brevity. Each cluster corresponds to a block pursued in the data matrix in Fig. 5.

The result of clustering is a list of 23 atomic actions shown in Table 3, together with the their relative frequencies normalized by the total number of collected clips. Each atomic action is represented by a list of grounded relations that are activated. The semantic description for atomic actions is in Table 3. And we also show corresponding video frames for a subset of atomic actions in Fig.6, together with a graphical representation to highlight the interaction between the agent and environment. The atomic actions that happen most frequently include a_5 (sitting at desk), a_{17}

(standing at desk), a_6 (sitting at desk, typing), a_4 (walking in the passageway) a_2 (standing at door) and a_9 (sitting at desk, grabbing a mug). a_5, a_6 can be considered as constituent components of a longer event "working at desk". a_{17}, a_4 probably happen between two events and serve as transitions. a_2 indicates the student is entering or leaving. The the learned atomic actions and their relative frequencies are representative and truthful to the video data.

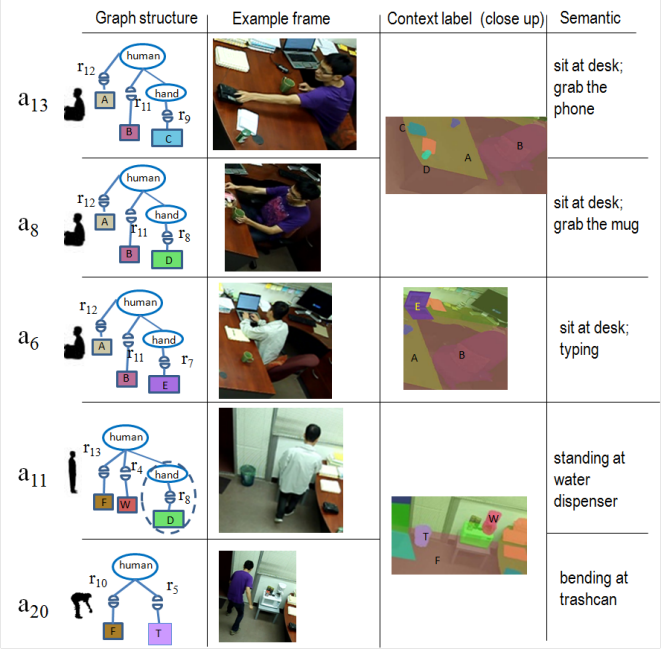

Figure 6. Illustrating a subset of the learned atomic actions.

Now the sequence of multi-dimensional relations is encoded by the alphabet of 23 atomic actions. For computational efficiency, to discover longer events we use hard assignments by computing the most likely atomic action per every 5 frames. The resulting sequence of atomic actions is

$$\mathbf{w}_{1:T} = (w_1, ..., w_T), \qquad \text{where } w_t \in \{a_1, ..., a_{23}\}$$

and T is the total number of video frames divided by 5.

3.3. Learning longer events and event grammar

Compressing the sequence of atomic actions. There is large variation in the duration of atomic actions. For example, a student may repeatedly enter the office, work for a varying time and leave the office. If we naively group atomic actions into longer ones, we get a large number of repetitive patterns of various lengths, providing little information. To deal with temporal variation, we perform a simple compression operation: every repetitive sub-sequence is summarized into one symbol (e.g. *bbbb* substituted by *b*). We may interpret this operation as learning a large number of grammar rules in the form $\tilde{N} \rightarrow NN...N$ with various lengths of repetition. We estimate a nonparametric model (Fig.8) for the length of repetition, or *duration* under maximum likelihood principle.

After compression, the original sequence of atomic actions $\mathbf{w}_{1:T}$ is transformed into a much shorter one $\mathbf{c}_{1:M}$ ($M << T$) where each symbol c_i takes value from the same domain as w_i. We then scan the sequence $\mathbf{c}_{1:M}$ to collect subsequences of length l ($l = 2$ in our system) and form a data matrix. Now the columns of this data matrix are atomic actions instead of grounded relations. A large number of homogeneous blocks (*i.e.* frequent sub-sequences) are identified from the data matrix. They are candidates for the right hand side of production rules in the event grammar. From the candidates, we select a subset of production rules in a step wise fashion according to their corresponding information gains.

A proposed candidate production rule takes the form $\alpha \rightarrow \beta\gamma$. It re-encodes the current sequence into a new sequence by replacing all occurrences of $\beta\gamma$ by α. By doing this, the information gain is computed as:

$$\text{info. gain} = \Delta_1 + \Delta_2 + \Delta_3 - \text{constant penalty} \qquad (3)$$

and,

$$\Delta_1 = n'_\alpha \cdot \left(\log \frac{n_\alpha}{n'}\right) - \log \frac{n_\beta}{n} - \log \frac{n_\gamma}{n}\Big)$$

$$\Delta_2 = n'_\beta \cdot \left(\log \frac{n'_\beta}{n'} - \log \frac{n_\beta}{n}\right) + n'_\gamma \cdot \left(\log \frac{n'_\gamma}{n'} - \log \frac{n_\gamma}{n}\right)$$

$$\Delta_3 = (n' - n'_\beta - n'_\gamma - n'_\alpha) \cdot \log \frac{n}{n'}$$

where $n'_\alpha, n'_\beta, n'_\gamma$ are the frequencies of α, β, γ in the newly encoded sequence respectively, n_β, n_γ are the corresponding frequencies in the current sequence. n is the length of the current sequence. $n' = n - n'_\alpha$ is the length of the new sequence. Eq.3 is a special case of Eq.2. We rank the candidate production rules using Eq.3 and select the largest one. This learning procedure is recursively carried out, until the information gain (or reduction of description length) is negative for any new candidate production rule. As a result, we obtain a dictionary of new production rules shown in Table 4, where to make the grammar more compact we merge shorter production rules into a longer ones that maximally reduce the description length. The learned production rules capture the interesting activities including working, entering, leaving, littering, fetching water and entertaining (watching soccer). Finally, by combining the production rules (*e.g.* $AB \cup AC \rightarrow A(B \cup C)$) we get a stochastic AND-OR grammar illustrated in Fig. 7, where for brevity we only show the graph structure and omit the branching probabilities of OR nodes. Here an AND node represents an event that is decomposed into subevents or atomic actions; an OR node represents alternative ways to realize an event. The learned AND-OR grammar contains a large amount of node sharing in the compositional hierarchy.

Table 4. Learned production rules of event grammar. For simplicity, we omit the starting symbol S and the branching probabilities that S produces the following non-terminal nodes.

production rule	description length reduction	semantic
$N_{24} \rightarrow a_{15}a_{16}$	.12	watching soccer,
$N_{32} \rightarrow N_{24}N_{24}$	.085	celebrating
$N_{29} \rightarrow a_{10}a_{11}a_{12}a_{11}a_{10}$	.381	
$N_{30} \rightarrow N_{29}a_9$	.080	
$N_{31} \rightarrow a_9N_{30}$	.067	
$N_{44} \rightarrow a_8N_{31}$	.057	
$N_{27} \rightarrow a_5a_6$	.080	working at desk: on computer or reading/writing, then leave
$N_{42} \rightarrow N_{27}a_5$	.042	
$N_{47} \rightarrow N_{42}a_{17}$	.039	
$N_{48} \rightarrow N_{47}a_4$	.022	
$N_{49} \rightarrow a_{17}N_{47}$	.025	
$N_{33} \rightarrow a_1a_2$	.073	entering the office, arrive and sit at the desk
$N_{34} \rightarrow a_2N_{33}$	.061	
$N_{38} \rightarrow a_4N_{34}$	.054	
$N_{39} \rightarrow N_{38}a_4$	.042	
$N_{40} \rightarrow a_{17}N_{39}$	.060	
$N_{41} \rightarrow N_{40}a_{17}$	.060	
$N_{46} \rightarrow N_{41}a_5$	.033	
$N_{35} \rightarrow a_{10}N_{21}$	.065	dumping waste at the trashcan
$N_{36} \rightarrow N_{35}a_{22}$	.074	
$N_{37} \rightarrow N_{36}a_{21}$	.055	
$N_{43} \rightarrow a_6a_5$	.037	full working mode
$N_{45} \rightarrow a_8a_5$	.045	Other frequent sub-events
$N_{50} \rightarrow a_4a_{17}$	.019	

3.4. Markov random field on the duration of events

A context free grammar $\mathcal{G}$ learned above can synthesize meaningful event sequences similar to observed ones. However, the durations of the synthesized events are random and uncorrelated. To capture this information, we define a Markov random field (MRF) on top of the durations of nodes in the parse graph pg produced by the event grammar, so that the grammar can model singleton, pairwise and higher-order statistics for the duration of sequential events. We use pooled histograms of duration as non-parametric potential functions of the MRF. Detailed specifications of the event parse graph and MRF are referred to [7].

3.5. Parsing with event grammar

We use an online parsing algorithm (details are referred to [7]) similar to Earley's parser [1, 11] to generate parse graphs based on the input data. Earley's algorithm reads terminal symbols sequentially, creating a set of all pending derivations (states) that is consistent with the input up to the current input terminal symbol. Given the next input symbol, the parsing algorithm iteratively performs one of three basic operations (prediction, scanning and completion) for each state in the current state set. To incorporate

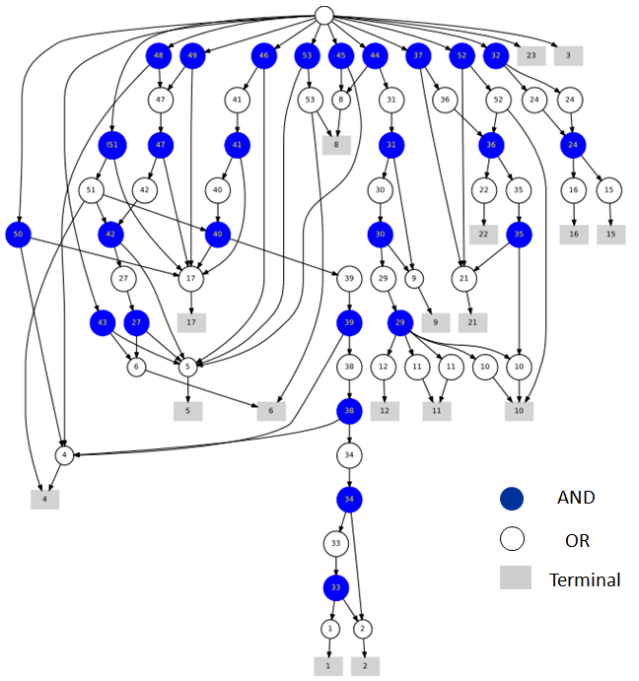

Figure 7. Graphical representation of the learned event AND-OR grammar.

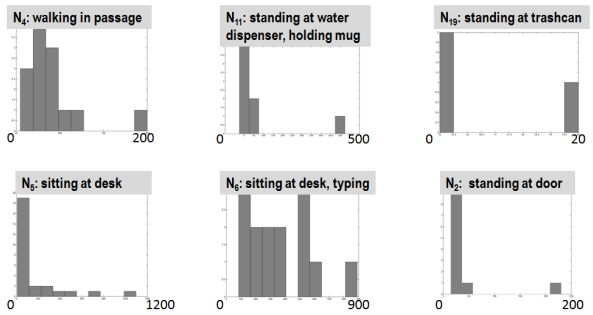

Figure 8. Singleton duration model $\phi(d_j)$ for atomic actions.

the Markov temporal model in Sec.3.4, we re-weight the proposed parse graphs by multiplying the probability computed in the Markov random field. For computational scalability, we only keep the top few candidate parses for the currently scanned sequence.

4. Evaluating the learned event grammar

4.1. Video parsing using learned event grammar

Using the learned event grammar, we parse the sequence of atomic actions extracted from a long video in Fig.9. The sequence is already compressed so that repeating subsequences are suppressed into single symbols. In the zoomed-out parts of the parse graph in Fig.9, we also show the detected bounding boxes of the agent. The semantic description for different non-terminal nodes is also illustrated.

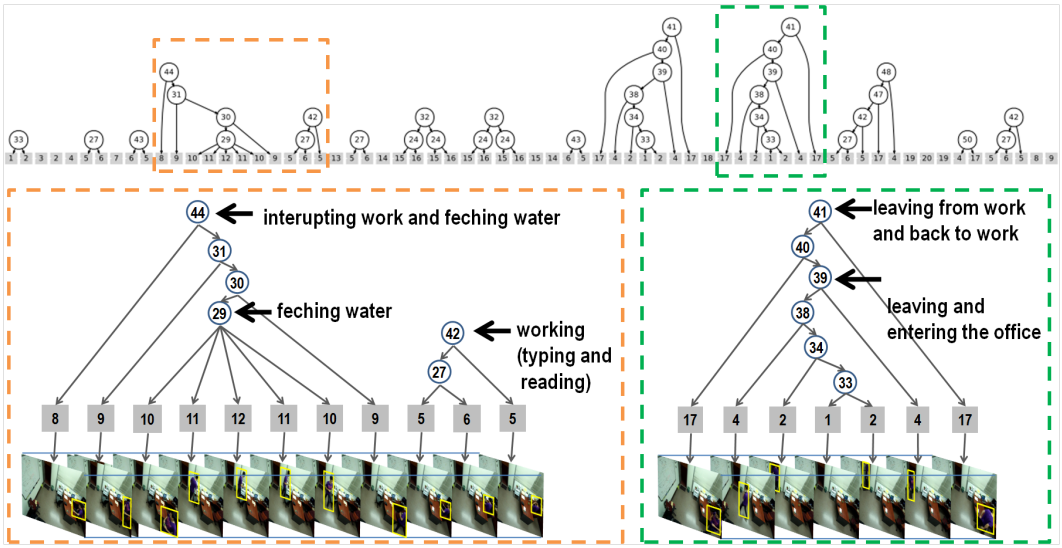

Figure 9. Video parsing result.

4.2. Event grammar helps atomic action detection

Due to the ambiguity of bottom up detection, the sequence of detected atomic actions is noisy and prone to error. We propose to use the learned event grammar to "denoise" the atomic actions sequence. With the learned spatial and temporal grammars as the prior, the detection of atomic actions follows a Bayesian maximum-a-posteriori:

$$\mathbf{a}^* = \arg\max_{\mathbf{a}} p(\mathbf{r}|\mathbf{a};\Theta)p(\mathbf{a};\mathcal{G})$$

where $\mathbf{r}$ is the sequence of grounded relations in the video. It is more robust than merely using bottom up proposals:

$$\mathbf{a}_{\text{bottom up}} = \arg\max_{\mathbf{a}} p(\mathbf{r}|\mathbf{a};\Theta)$$

where $\mathcal{G}$ is the learned grammar, and Θ are parameters of the bottom up detectors of atomic actions. We perform an experiment on a collection of 12061 frames. Table 5 shows the comparison of classification performance before and after using the learned grammar to correct bottom up detections for atomic actions $\{a_1, ..., a_{23}\}$.

Table 5. atomic actions detection

atomic action	bottom up	bottom up + event parsing
standing at door	90.7%	100%
walking	78.9%	100%
sitting at desk	84.1%	96.4%
standing at desk holding mug	55.4%	98.6%
celebrating at desk, typing, soccer match on screen	85.0%	100%
standing at water dispenser	63.3%	100%
sitting at desk, typing	88.1%	100%
sitting at desk (not typing)	84.1%	100%

4.3. Scene semantics from event recognition

In the previous sections, the learning and parsing of event grammar relies on manual labeling of scene semantics (*i.e.* the scene label map in Fig.1). Now we try to release this requirement of manual labeling, and use the event grammar to infer scene semantics automatically, thus closing the loop of unsupervised learning. For this task, we need to use techniques in interactive image segmentation (*e.g.* [16]), where the user draws " scribbles " on the image to indicate a certain pixels as foreground or background, and a computer program automatically segment the image into foreground and background regions.

Treating the scene semantics as missing variables, we can use the learned event grammar to segment and recognize objects and regions of interest in the scene. Now the only bottom-up information includes the agent's poses and trajectories, from which we can still obtain the most likely parse graph $\hat{\text{pg}}$, or sample a set of likely parse graphs ([7]) by marginalizing out the missing variables. Given the event parse graph $\hat{\text{pg}}$, we generate probabilistic "scribbles" on the scene image according to trajectories of agent's hands and feet and agent poses. For example, if the agent is in sitting pose for a long period, then a region surrounding the agent is labeled as desk with probability:

$$p(x^{\text{desk}}|\hat{\text{pg}}) = \sum_{t,x^{\text{agent}}} p(x^{\text{desk}}|t, x^{\text{agent}})p(t, x^{\text{agent}}|\hat{\text{pg}})$$

$$= \sum_{t=1}^{T} p(x^{\text{desk}}|x_t^{\text{agent}}) \frac{p(\text{sitting}|\mathbf{I}_t, \hat{\text{pg}})}{\sum_{j=1}^{T} p(\text{sitting}|\mathbf{I}_j, \hat{\text{pg}})}$$

where $\mathbf{I}_t$ denotes the video frame at time t, x_t^{agent} is the detected location of agent at time t, $p(x^{\text{desk}}|x_t^{\text{agent}})$ is a spatial model for the position of desk relative to the sitting agent, and $p(\text{sitting}|\mathbf{I}_t, \hat{\text{pg}})$ is the probability that the event

sitting is present in $\mathbf{I}_t$ according to the parse graph. Here a more principled way is to also marginalize out the parse graph, but for computational efficiency we use the most likely parse graph $\hat{\text{pg}}$. As another example, if the agent's hand reaches out quickly and stops, then the region near the inflection point is labeled as "touch-able" objects such as mug and phone, each with normalized probability. This probabilistic scribble map is then used as input to an interactive segmentation method, which minimizes a two-part energy that can be expressed with log-likelihood:

$$\min_{\mathbf{L}} -\log p(\hat{\text{pg}}|\mathbf{L}) - \log p(\mathbf{L})$$

The data term $-\log p(\hat{\text{pg}}|\mathbf{L})$ denotes the discrepancy between the label map $\mathbf{L}$ and the probabilistic scribble map produced by $\hat{\text{pg}}$. The smoothness term $-\log p(\mathbf{L})$ denotes discontinuities of neighboring sites in the label map $\mathbf{L}$.

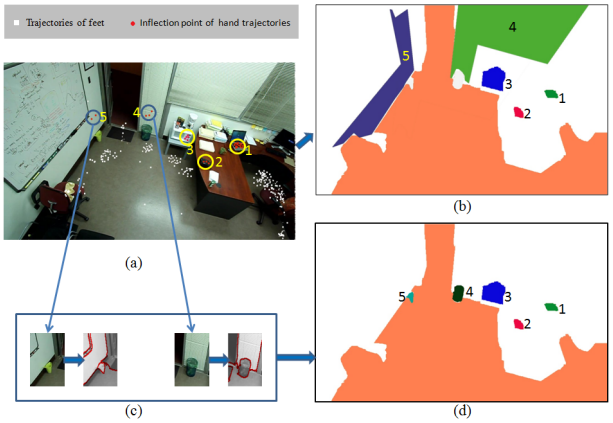

Figure 10. Scene segmentation by parsed trajectories. (a) The trajectories of the agent's hands and feet. (b) The segmentation of objects by the trajectory "scribbles". (c) The segmentation of adjacent areas of 4 and 5. (d) The final segmentation result for interesting objects.

Fig.10 (a) shows the trajectories of the agent's hands and feet. Fig.10 (b) shows the segmentation result by the trajectories. The ground is successfully segmented by the trajectories of the feet. The keyboard, phone, microwave are segmented by concentrated trajectories of hands. The segments 4 and 5 in Fig.10 (b) are too large to be interest objects, so we prune them. Fig.10 (d) shows the final segmentation result of interesting objects in the scene.

5. Conclusion

In summary, we propose a prototype system for event learning, which explores all activities that happen in a certain environment, and organizes them in a meaningful way by a hierarchical event dictionary and a stochastic event grammar. The learned event grammar can be used to parse newly observed videos to recognize events. We also show a promising application where it is used to discover scene semantics without manual labeling of the scene. We are working towards applying to more diverse datasets and obtaining richer event grammar.

Acknowledgement: The work is supported by NSF IIS 1018751 and MURI grants N00014-10-1-0933, N00014-11-c-0308.

References

[1] J. C. Earley. *An Efficient Context-Free Parsing Algorithm*. PhD thesis, Carnegie-Mellon Univ, 1968. 6

[2] Y. A. Ivanov and A. F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *TPAMI*, 22(8):1–21, 2000. 1

[3] Z. Lin, Z. Jiang, and L. S. Davis. Recognizing actions by shape-motion prototype trees. In *ICCV*, 2009. 1

[4] D. Moore and I. Essa. Recognizing multitasked activities using stochastic context-free grammar. In *AAAI*, 2001. 1

[5] F. Nater, H. Grabner, and L. V. Gool. Exploiting simple hierarchies for unsupervised human behavior analysis. In *CVPR*, 2010. 1

[6] J. C. Niebles, C.-W. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010. 1

[7] M. Pei, Y. Jia, and S.-C. Zhu. Parsing video events with goal inference and intent prediction. In *ICCV*, 2011. 3, 6, 7

[8] M. S. Ryoo and J. K. Aggarwal. Stochastic representation and recognition of high-level group activities. *IJCV*, 93(2):183–200, 2011. 1

[9] Z. Si and S.-C. Zhu. Learning and-or templates for object recognition and detection. Technical report, Statistics Dept., UCLA, 2011. 4

[10] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Conditional models for contextual human motion recognition. *CVIU*, 104:210–220, 2006. 1

[11] A. Stolcke. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics*, 1995. 6

[12] K. Tu and V. Honavar. Unsupervised learning of probabilistic context-free grammar using iterative biclustering. In *Proceedings of the 9th international colloquium on Grammatical Inference: Algorithms and Applications*, ICGI '08, pages 224–237, 2008. 4

[13] S. F. Wong, T. K. Kim, and R. Cipolla. Learning motion categories using both semantic and structural information. In *CVPR*, 2007. 1

[14] J. Yuan, Y. Wu, and M. Yang. From frequent itemsets to semantically meaningful visual patterns. In *SIGKDD*, 2007. 1

[15] Z. Zhang, K. Q. Kuang, T. N. Tan, and L. S. Wang. Trajectory series analysis based event rule induction for visual surveillance. In *CVPR*, 2007. 1

[16] Y. Zhao, S.-C. Zhu, and S. Luo. Co3 for ultra-fast and accurate interactive segmentation. *ACM Multimedia*, 2010. 7

[17] S.-C. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2(4):259–362, 2006. 1