in Proc. IEEE Computer Vision and Pattern Recognition (CVPR), Colorado Springs, CO, 2011

Probabilistic Event Logic for Interval-Based Event Recognition

William Brendel, Alan Fern, Sinisa Todorovic
Oregon State University, Corvallis, OR, USA
brendelw@onid.orst.edu, afern@eecs.oregonstate.edu, sinisa@eecs.oregonstate.edu

Abstract

This paper is about detecting and segmenting inter-related events which occur in challenging videos with motion blur, occlusions, dynamic backgrounds, and missing observations. We argue that holistic reasoning about time intervals of events, and their temporal constraints is critical in such domains to overcome the noise inherent to low-level video representations. For this purpose, our first contribution is the formulation of probabilistic event logic (PEL) for representing temporal constraints among events. A PEL knowledge base consists of confidence-weighted formulas from a temporal event logic, and specifies a joint distribution over the occurrence time intervals of all events. Our second contribution is a MAP inference algorithm for PEL that addresses the scalability issue of reasoning about an enormous number of time intervals and their constraints in a typical video. Specifically, our algorithm leverages the spanning-interval data structure for compactly representing and manipulating entire sets of time intervals without enumerating them. Our experiments on interpreting basketball videos show that PEL inference is able to jointly detect events and identify their time intervals, based on noisy input from primitive-event detectors.

1. Introduction

We study modeling and recognition of multiple video events that are inter-related in various ways. Such events arise in many applications, including sports video, where several players perform coordinated actions, like *running*, *catching*, and *passing* to achieve a goal. Recognizing such events under occlusion and amidst dynamic, cluttered background is challenging. We address these uncertainties by: (I) Jointly modeling events in terms of time intervals that they occupy in the video, and their spatiotemporal relationships; and (II) Resorting to domain knowledge that can provide useful soft and hard constraints among the events, and thus help reduce ambiguities in recognition.

Given a video, we use domain knowledge and observations to: (1) recognize every event occurrence, (2) localize the time intervals that they occupy; and (3) explain their recognition in terms of the identified spatiotemporal relationships and semantic constraints from domain knowledge.

To address (1)–(3), we introduce probabilistic event logic (PEL). PEL uses weighted logic formulas to represent arbitrary probabilistic constraints among time intervals. This generalizes much prior work that constrains time points, rather than intervals. PEL's logic-based nature facilitates injection of human prior knowledge. Further, PEL avoids the brittleness of pure logic by associating weights with formulas that represent the cost of formula violations. Thus, a video interpretation that violates a formula becomes less probable, but not impossible, as in pure logic.

To address the scalability issue of reasoning about all time intervals of a video, we develop a new MAP inference algorithm for PEL. PEL inference leverages the spanning-interval data structure for compactly representing and efficiently manipulating entire sets of time intervals. Accordingly, our algorithm's time and space complexity does not necessarily grow with the length of a video, but rather with the much smaller number of spanning intervals.

Motivation. It is worth considering how the state-of-the-art methods — specifically, graphical-modeling based methods, such as MRFs or CRFs, suited for holistic reasoning about events and their temporal context — could be used to realize our goals (1)–(3). They would, first, need to partition the video into atomic time intervals (e.g., by using spatiotemporal segmentation, or scanning windows of primitive-event detectors), and, then, associate random variables with each of the quadratically many pairs of time intervals. The variables would serve to encode observations (e.g., noisy primitive event detections) and hidden information (e.g., more abstract events) about those intervals, as well as relationships between the intervals. Standard inference mechanisms, such as belief propagation or MCMC, could then be used to assign values to the variables, yielding a holistic video interpretation in terms of (1)–(3). Unfortunately, such a hypothetical approach is intractable for realistic videos, due to an enormous number of variables and constraints that a graphical model would contain. Another issue is that it would produce poor event localization results.

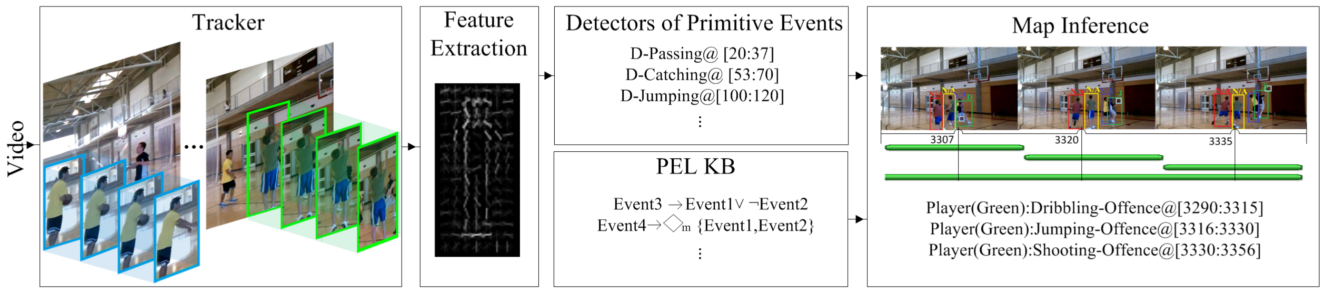
Video
Tracker
Feature Extraction
Detectors of Primitive Events
D-Passing@ [20:37]
D-Catching@ [53:70]
D-Jumping@[100:120]
⋮
PEL KB
Event3 → Event1 ∨ ¬Event2
Event4 → ◇m {Event1,Event2}
⋮
Map Inference
3307
3320
3335
Player(Green):Dribbling-Offence@[3290:3315]
Player(Green):Jumping-Offence@[3316:3330]
Player(Green):Shooting-Offence@[3330:3356]

Figure 1. An overview of our approach in the context of 2-on-2 basketball games: We use a tracker to obtain spatiotemporal tubes of the four players, the ball, and the rim. Then, we apply a scanning-window detector to each tube to localize the time intervals of primitive events. These noisy detections are combined with the PEL knowledge base (KB). A MAP inference is applied to produce a holistic video interpretation, which specifies the occurrence intervals of all observable and hidden events.

This would be particularly pronounced for more abstract events. Suppose, for example, a basketball player is just *standing* during the game, and the goal is to identify when the player is *on offense*. The event *on offense* may happen arbitrarily at any subinterval of *standing*, because it is related to the activities of the other players. Since there is no low-level segmenter, or primitive-event detector that would be able to identify this subinterval, the localization error of the event *on offense* would inherently be large. One could try to heuristically partition the video into even smaller time intervals than those initially provided; however, this would lead to the aforementioned tractability issues. Alternatively, one could begin with a small set of (e.g., most salient) intervals, and then incrementally add intervals to the model during inference. While such an approach is potentially viable, we offer a more direct approach that avoids pre- and post-processing of the intervals altogether, and gains efficiency by reasoning about entire blocks of intervals.

Overview. Fig. 1 shows an overview of our approach. PEL inference begins with noisy detectors that attempt to localize time intervals occupied by primitive events. These detections are combined with the PEL domain knowledge, including hard and soft constraints, to produce a MAP video interpretation in terms of the occurrence intervals for all observable and hidden events of interest.

2. Prior Work and Our Contributions

Spatiotemporal constraints among a set of events can be represented by: dynamic Bayesian networks [21]; context-free grammars [8, 11, 7]; AND-OR grammars [3, 6]; and conditional random fields [14, 13]. These approaches typically encode only pairwise event constraints. Our novelty is in formulating a distributed system of event definitions in terms of pairwise and higher-order probabilistic constraints, which jointly define each event. Also, these approaches typically take time points as primitives of their models. Specifically, they usually partition the video into time instances, and make the assumption that the Markovian independence holds between these time instances. Thus, they do not explicitly model event intervals, but derive them from a set of points in time. This is in light of the well-established understanding that many types of events are fundamentally interval-based, and are not accurately modeled in terms of time points [1]. By contrast, our PEL allows for explicit modeling of intervals. It specifies probabilistic constraints on properties of, and relationships among time intervals that must be satisfied by a complex system of interrelated events.

A set of inter-related events can also be modeled by combining atemporal logics with grammars and graphical models [12, 15, 17, 18, 16, 20], such as, e.g., Markov Logic Networks [20] and penalty logic [4]. However, they do not address the aforementioned limitations, because their first-order objects are time points, instead of continuous intervals. Direct extensions of these approaches to an interval-based notion of time encounters tractability issues.

The advantages of representing events by an interval-based logic have been demonstrated in [19, 5]. However, interval-based logic has been used exclusively to specify events in terms of subevents, and does not have a probabilistic mechanism for addressing uncertainty. Our PEL generalizes this work by: (i) allowing arbitrary constraints among constituent and non-constituent events, and (ii) defining a probabilistic semantics, and conducting a probabilistic inference over weighted logic formulas.

3. Syntax and Semantics of PEL

We first review pure event logic, and then extend to PEL.

3.1. Event Logic

Event logic (EL) was introduced by Siskind [19] for defining interval-based events. Its syntax defines: event symbols, interpretations, and formulas, as explained below.

Event Symbols. An *event symbol* is a character string that gives a name to an event of interest. Event symbols may have arguments, e.g., Running($P1$) is the event that player $P1$ is running. PEL distinguishes between observable (detected) events, and hidden events, similar to observed and hidden variables in generative models. Event symbols for observable events will have a prefix of "D-", for "detected", e.g., D-Shooting($P2$). Each observable event, also called primitive event, has a corresponding hidden event, e.g., Shooting($P2$). Not all hidden events have the corresponding observed events, e.g., the event on Offense($P3$). We wish to infer all hidden events from (noisy) detected events.

Interpretations. Truth values are assigned to *event occurrences*, which have the form $E@I$, for event symbol E and time interval $I = [a, b]$, where a and b are positive integers such that $a \leq b$. Asserting that $E@I$ is true means that an instance of E occurred precisely over interval I. An *interpretation* over a set of event symbols, is a set of event occurrences involving those symbols that contains all of the true event occurrences, and no others. We will denote an interpretation by (X, Y), where X is the set of observable event occurrences, and Y is the set of hidden event occurrences. In our basketball domain, X will be composed of detected event occurrences, e.g. D-Dribbling$(P3)@[10, 30]$, and Y will be composed of hidden event occurrences, such as Defense$(P3)@[20, 30]$ which we must infer based on the noisy information in X. There can be exponentially many valid interpretations for any given X, and our goal is to infer the best one.

Formulas. EL uses *formulas* to specify constraints on interpretations in terms of static and dynamic properties of time intervals, by relating the intervals via the seven Allen relations [1]: co-occur (=), strictly before (<), meet (m), overlap (o), start (s), finish (f), and during (d). For example, the interval $[2, 3]$ is before $[5, 6]$, meets $[4, 5]$, overlaps $[3, 4]$, starts $[2, 4]$, finishes $[1, 3]$, and is during $[1, 4]$. We also use inverses of relations, e.g., "mi" is inverse meets. We recursively define EL formulas as follows. A formula is either an event symbol E (a primitive formula), or one of the compound expressions $\neg\phi$, $\phi \vee \phi'$, $\phi \wedge_r \phi'$, or $\Diamond_r \phi$, where ϕ and ϕ' are formulas, and r is one of the Allen relations (we will commonly use the shorthand $\phi \to \phi'$ for $\neg\phi \vee \phi'$).

The semantics of formulas are specified by defining when a given formula ϕ is satisfied (true) along an interval I of an interpretation (X,Y), denoted by $(X, Y) \models \phi@I$. The $\models$ relation can be defined recursively as follows: for a primitive formula E, $E@I$ is satisfied if it is in (X, Y); $\neg\phi@I$ is satisfied if $\phi@I$ is not satisfied; $\phi \vee \phi'@I$ is satisfied if either $\phi@I$ or $\phi'@I$ are satisfied; $\phi \wedge_r \phi'@I$ is satisfied if ϕ and ϕ' are true along some intervals I_1 and I_2 that are related by r and span I; and finally $\Diamond_r\phi@I$ is true if ϕ is true along an interval I' that is related to I by r. Later, it will be useful to consider the set of all intervals in which ϕ is true in (X, Y), which we will denote at SAT$((X, Y), \phi) = \{I \mid (X, Y) \models \phi@I\}$.

Intuitively, by combining $\neg$, $\vee$, and primitive events it is possible to specify arbitrary constraints that must hold over an interval I. For example, the formula Dribbling$(p) \to$ HasBall(p) is true of intervals where if p is dribbling then they are also identified as having the ball. The $\wedge_r$ operator allows for specifying temporal constraints between intervals. For example, the formula PassTo$(p, q) \to$ (Pass$(p) \wedge_m$ BallMoving $\wedge_m$ Catch(q)), is true of an interval if when the passing event occurs there is a meeting sequence of events starting with the pass, the ball moving, and ending with a catch. This specifies a necessary condition for PassTo. Finally, the $\Diamond_r$ operator allows for specifying constraints on intervals related to a given interval I. For example, the formula, [HasBall$(p) \wedge$Jumping(p)] $\to$ $\Diamond_{mi}$[$\neg$HasBall$(p) \vee$Jumping(p)] encodes that a player cannot jump with the ball and then land with the ball.

Note that by including a formula in an EL KB, we indicate that any valid interpretation must satisfy the formula along all of its intervals, otherwise the interpretation is ruled out as invalid. This can be quite brittle, since even the smallest violation of a constraint renders an interpretation invalid. Below, we explain how PEL addresses this limitation.

3.2. Probabilistic Event Logic

A *PEL knowledge base (KB)* is a set of weighted event-logic formulas: $\Sigma = \{(\phi_1, w_1), \ldots, (\phi_n, w_n)\}$, where w_i is a non-negative numeric weight associated with formula ϕ_i, representing a cost of violating ϕ_i over an interval, relative to all other formulas in KB. Note that formulas with large weights relative to others will behave as hard constraints. Σ assigns a score, S, to any interpretation (X, Y)

$$S((X, Y), \Sigma) = \textstyle\sum_i w_i \cdot |\text{SAT}((X, Y), \phi_i)|, \qquad (1)$$

where $|\text{SAT}((X, Y), \phi)|$ is the number of intervals in (X, Y) satisfied by ϕ.

Given S, we specify the posterior of the hidden part of interpretations as $\Pr(Y|X, \Sigma) \propto \exp(S((X, Y), \Sigma))$. Since S can be viewed as a weighted sum of features of (X, Y) (one feature per formula), this model is a log-linear probability model, analogous to CRF. Our model can be used to answer arbitrary probabilistic queries about the hidden events in an interpretation. We here focus on solving the MAP inference problem for PEL, i.e., computing $\text{MAP}(X, \Sigma) = \arg\max_Y S((X, Y), \Sigma)$.

Given a PEL KB and a MAP inference procedure, we compute an interpretation for a video, V, as follows. First, we run a set of event detectors on V, as described in Sec. 6. This produces a set of observed event occurrences $X = \{\text{D-}E_1@I_1, ..., \text{D-}E_k@I_k\}$ where the detector asserts that observable events D-E_i occurred at each interval I_i. For example, in basketball, the detector might produce event occurrence D-Catching(P1)@[1,10]. Note that it is not necessarily the case that, in reality, the player 1 catches the ball in interval [1,10]. Rather, this provides evidence, and the actual act of catching must be inferred.

4. PEL Inference

We consider efficiently computing $S((X,Y),\Sigma)$ and MAP inference. This could be solved by compiling a PEL KB into an equivalent graphical model (e.g., as is done for Markov Logic Networks), and applying existing inference algorithms. However, such compilations would require introducing a distinct variable for every event occurrence $E@I$, where I is any subinterval of a video's time interval $[1,T]$, resulting in $O(T^2)$ variables. Instead, we develop a new inference algorithm, directly for PEL.

Spanning Intervals. We avoid enumerating over the $O(T^2)$ time intervals via the use of *spanning intervals (SI)*. SIs were introduced by Siskind [19], but have not yet been exploited for probabilistic inference, which is a key contribution of our work. An SI is denoted by $[[a,b],[c,d]]$, where a,b,c,d are non-negative integers, and is used to represent the set of intervals that begin somewhere in $[a,b]$, and end somewhere in $[c,d]$. That is, $[[a,b],[c,d]]$ represents the set $\{[p,q] \mid p \in [a,b], q \in [c,d], p \leq q\}$. Note that the SI of a temporally disjoint set of intervals is a union of SIs.

We use an SI to compactly represent the set of all event occurrences where the corresponding event formula is satisfied. Specifically, given an SI, $\mathbb{S}$, we write $E@\mathbb{S}$ to denote the set of all event occurrences, $E@I$, where $I \in \mathbb{S}$. In this way, we can compactly represent interpretations by specifying all event occurrences in terms of SIs, which can provide quadratic space savings.

Our inference performs set operations over SIs to identify time intervals where the event formulas of the PEL KB are true. Computing set operations over SIs is very efficient. For example, the intersection of two SIs is easily computed in $O(1)$ time as: $[[a_1,b_1],[c_1,d_1]] \cap [[a_2,b_2],[c_2,d_2]] = [[\max(a_1,a_2),\min(b_1,b_2)],[\max(c_1,c_2),\min(d_1,d_2)]]$. Importantly, the complexity of these operations does not depend on the temporal extent of the intervals, but rather only on the much smaller number of SIs.

Computing Scores. Equation (1) shows that to efficiently compute S we must efficiently compute $|\text{SAT}((X,Y),\phi)|$. To this end, we compute an SI representation of $\text{SAT}((X,Y),\phi)$, and then find the number of its intervals $|\text{SAT}((X,Y),\phi)|$. In particular, we compute $\text{SAT}((X,Y),\phi)$ by recursion, as follows. If ϕ is a primitive formula E, then SAT returns the SIs associated with E in (X,Y). For SAT of $\neg\phi$, we compute SIs for ϕ, and then apply the SI complement operator. The SAT of $\phi \vee \phi'$ is the union of the SIs of ϕ and ϕ'. For SAT of $\Diamond_r\phi$, we first compute the SIs for ϕ, and then apply the SI operator for the appropriate Allen relation r. For example, if ϕ is satisfied along $\mathbb{S} = [[a,b],[c,d]]$ and $r = m$ (i.e. "meets"), then we would get $[[1,T],[a-1,b-1]]$, giving the set of all intervals that meet an interval in $\mathbb{S}$. The complexity of SAT depends on the size of the SI representation of (X,Y) and ϕ. In the worst case, the SI representation can grow exponentially large in the nesting depth of ϕ. In practice, we observe that the SI representations remain vanishingly small compared to $O(T^2)$.

MAP Inference. To conduct inference, for convenience, we compile a PEL KB to an equivalent PEL conjunctive normal form (PEL-CNF), where the equivalence holds with respect to the MAP inference result. To this end, we rewrite the weighed formulas of a PEL KB as clauses, i.e., disjunctions of literals: E, $\Diamond_r E$, $E \wedge_r E'$, and their negations. The following definition and theorem formally state that this compilation can be done efficiently.

Definition. Given a set of event symbols $\mathcal{E}$, two PEL KBs Σ and Σ' are MAP equivalent with respect to $\mathcal{E}$ iff for all sets of observed events X, $\text{MAP}(X,\Sigma)$ and $\text{MAP}(X,\Sigma')$ agree on all occurrences of event symbols from $\mathcal{E}$.

Theorem. *Given any PEL KB Σ over event symbols $\mathcal{E}$, there is a MAP equivalent PEL-CNF KB Σ' with respect to $\mathcal{E}$, which can be computed in time linear in the size of Σ.*

Proof:(Sketch) For any EL formula ϕ one can create a new event symbol E_ϕ and set of clauses C_ϕ such that if the clauses are all satisfied then $\phi@I$ is true iff $E_\phi@I$ is true. This tool allows replacing non-clausal structure with weighted clauses, where the C_ϕ clauses are assigned "large enough" weights to act as hard constraints.$\square$

MAP for PEL is NP-hard since it can easily encode 3-SAT. Thus, we consider an approximate MAP approach based on stochastic local search (SLS). Our PEL-SLS (Figure 2) algorithm takes as input a PEL-CNF KB, Σ, observations, X, and a noise parameter, p. The output is a set of hidden event occurrences, Y, such that the interpretation, (X,Y), is high scoring, ideally the MAP solution. Starting with an empty set Y_0 the algorithm produces a sequence $Y_1, Y_2, \ldots$ for a desired number of iterations, and returns the highest scoring Y_i. On each iteration, Y_{i+1} is produced from Y_i, as follows. First, the algorithm computes the set of formulas in Σ that are violated somewhere in the current interpretation (X,Y_i), and randomly selects one such formula ϕ. Next the algorithm selects a random SI, $\mathbb{S}$, over which ϕ is violated in (X,Y_i). The key idea is to then identify changes to Y_i so that ϕ is satisfied along all intervals in $\mathbb{S}$. This is accomplished by the MOVES function (see below) which returns a set of such alterations to Y_i. Usually Y_{i+1} is set to the move that achieves the highest score, but with probability p, it is a random move to avoid local maxima.

It remains to describe the MOVES function. The moves for $\phi \vee \phi'$ is $\text{MOVE}(\phi,\mathbb{S},(X,Y_i)) \cup \text{MOVE}(\phi',\mathbb{S}',(X,Y_i))$ since any valid move for ϕ or ϕ' will satisfy the disjunction. Since clauses are just disjunctions of literals, it remains to define moves for each possible form of literal. A primitive literal E, produces a single move that adds $E@\mathbb{S}$ to Y_i, noting that SI set operations are used to combine $E@\mathbb{S}$ with the occurrences of E already in Y_i. The literal $\neg E$ also yields a single move that uses SI operations to delete $E@\mathbb{S}$ from

```
PEL-SLS
// PEL-CNF KB: $\Sigma = \{(\phi_1, w_1), \ldots, (\phi_n, w_n)\}$
// Observations: $X$
// Noise Parameter: $0 \leq p \leq 1$
$Y_0 \leftarrow \emptyset$; i=0;
repeat for desired iterations,
    $\Phi = \{\phi_j \mid \text{SAT}((X, Y_i), \neg\phi_j) \neq \emptyset, j = 1, \ldots, n\}$
    $\phi \leftarrow$ RandomElement($\Phi$)
    $S \leftarrow$ RandomElement (SAT($(X, Y_i), \neg\phi$))
    $\mathcal{Y} = \{Y^{(1)}, \ldots, Y^{(k)}\}$ = MOVES($\phi, S, (X, Y_i)$)
    if flip(1-p) then $Y_{i+1} \leftarrow \arg\max_{Y \in \mathcal{Y}} S((X, Y), \Sigma)$
    else $Y_{i+1} \leftarrow$ RandomElement($\mathcal{Y}$)
    $i \leftarrow i + 1$
return Highest scoring $Y_i$
```

Figure 2. PEL Stochastic Local Search

Y_i. The moves for the literal $\Diamond_r E$ correspond to adding $E@\mathbb{S}'$ to Y_i for some SI, $\mathbb{S}'$, such that for each $I \in \mathbb{S}$ there is an r-related $I' \in \mathbb{S}'$. There are typically many possible choices for $\mathbb{S}'$, and the choice of which one to select is largely heuristic, while guaranteeing completeness via appropriate randomization. As an example, consider $r = m$ and $\mathbb{S} = [[a, b], [c, d]]$. One choice for $\mathbb{S}'$ is all possible intervals that meet an interval in $\mathbb{S}$, i.e. $[[1, T], [a-1, b-1]]$. Our implemented system generates a number of possibilities, and returns one randomly as the move. Handling the other literals follows a similar pattern, and is not covered here for space reasons. All of our MOVE operators work directly on SIs avoiding the $O(T^2)$ enumeration problem.

5. Learning PEL Formula Weights

This section presents an algorithm for learning the weights of a set of EL formulas $\{\phi_1, \ldots, \phi_n\}$ using a training set of interpretations $D = \{(X_i, Y_i)\}$. Each training example is derived from a video, where the X_i are the observed event occurrences based on detectors, and the Y_i are the ground truth hidden event occurrences, provided by a human labeler. The goal is to learn weights, resulting in a PEL knowledge base $\Sigma = \{(\phi_1, w_1), \ldots, (\phi_n, w_n)\}$, such that MAP$(X_i, \Sigma)$ is (approximately) equal to $Y_i, \forall i$.

We use the PEL-SLS algorithm to approximate the MAP inference during learning. Specifically, we use a variant of Collins' generalized Perceptron algorithm [2]. The main requirement of the algorithm is that the scoring function which evaluates examples (i.e., interpretations) be representable as a linear combination of n features. From (1), this requirement can be met by defining a feature, f_i, for each formula as $f_i((X, Y)) = |\text{SAT}((X, Y), \phi_i)|$. Starting with all zero weights, the algorithm iterates through the training interpretations, and for each (X_i, Y_i) uses the current weights to compute the current MAP estimate Y, based on X_i. If $Y = Y_i$ then there is no weight update, otherwise the weights are adjusted to reduce the score of (X_i, Y), and to increase the score of the correct interpretation (X_i, Y_i). In particular, for each weight w_j the update is $w_j \leftarrow w_j + \alpha \cdot (f_j((X_i, Y_i)) - f_j((X_i, Y)))$, where $0 < \alpha \leq 1$ is a learning rate. Unlike Collin's algorithm, if the update produces a negative weight, we set it to zero. This variant of the Perceptron algorithm preserves the main convergence property of the original algorithm [4].

6. Detection of Primitive Events

This section describes the tracker and detector we use for detecting primitive events and their time intervals.

Tracking: Given a video of a 2-on-2 basketball game, the goal of tracking is to extract spatiotemporal tubes of the four players, the ball, and the rim. This is challenging, because the uncertainty about the targets may arise from a multitude of sources, including: changes in the players' scales, occlusions over relatively long time intervals, and dynamic, cluttered backgrounds. The state of the art poorly performs in the face of these challenges [22]. Therefore, we have implemented a semi-supervised tracking system based on the template matching approach of [9]. Tracking of [9] is interactively corrected by the user. First, the user delineates a bounding box around the target. Then, the target is automatically tracked by convolving the target's template with every video frame. The convolution output is expected to be highest at places where the object occurs. The template is updated at each frame by the best match found in the previous frame. On average, the user has to correct about 10 frames per minute of the video. The user edits include repositioning of the bounding box to the right location, and correcting the ID label of the bounding box.

Detection of Primitive Events: We scan each extracted tube with windows of different lengths (30:30:300 frames, shifted by 5 frames), to detect primitive events and localize their time intervals. We use the popular Bag-of-Words detector [10]. Specifically, from a tube's window, we extract 2D+t Harris corners [10], and describe them by the histogram of gradients (HoG) and the histogram of flow (HoF). Then, we map these descriptors to a codebook of visual words, and classify the resulting histogram of codewords by a linear SVM. The codebook is obtained by K-means clustering of all descriptors from the training set (K=300).

7. Results

For evaluation, we use two datasets. The first is our dataset of actual (not staged) 2-on-2 basketball games (see Fig. 5). The basketball dataset is suitable for evaluating detection and localization of multiple events characterized by rich spatiotemporal constraints. The videos show a real-world setting with the following challenges: camera motion, changes in the player's scale, motion blur of fast ac-

Events	Number of Intervals			Number of Frames		
	Groundtruth		Detection	Groundtruth		Detection
	train	test	results	train	test	results
Dribbling	50	24	18	6067	2773	2177
Jumping	86	46	33	3053	1393	976
Shooting	39	20	16	1029	494	264
Passing	72	36	38	2153	1032	1104
Catching	64	30	20	672	334	228
Bouncing	85	38	34	10380	3788	3396
NearRim	46	24	28	5067	2468	2618
BallTrajectory	62	41	27	2412	1280	842
Defense	244	108	114	17342	5346	5989
Offense	300	116	104	12123	4834	4332
HasBall	289	109	71	2604	1280	842

Table 1. The total number of frames and time intervals occupied by the 8 primitive events and 3 higher-level events in our basketball dataset. The top 5 primitive events and 3 higher-level events are performed by the 4 players. The remaining bottom 3 primitive events are associated with the ball. Note that all 4 players and the ball cannot be seen all the time. Also, the event *defense* can be associated with the players who do not perform any of the primitive events from the list (e.g., when they simply stand). The detection results are obtained by PEL inference on the test sequences.

tions, frequent inter-player occlusions, varying illumination. The four players, ball, and rim are tracked and labeled in the training and test sets with 8 primitive events, and 3 higher-level (hidden) event, listed in Tab. 1. Frames that do not contain the events from Tab. 1 have been removed from the videos. We plan to extend annotations of our basketball dataset and make them public.

The second dataset contains 50 YouTube videos for each of 16 classes of Olympic sports [13]. Each event is performed only by a single subject, and represents only a sequence of primitive actions in a *meet* relationship (e.g., *long-jump* consists of *standing still*, followed by *running*, *jumping*, *landing*, and *standing up*).

PEL formulas are specified based on our domain knowledge of basketball and Olympic sports. The formula weights are learned on training examples. We use the following evaluation metrics: (a) segmentation accuracy as the ratio of intersection and union of inferred and ground-truth time intervals of events, (b) detection error, where true positives are detected events with segmentation accuracy greater than 50%, and (c) accuracy defined as the total number of true positives and true negatives divided by the total number of event instances.

Testing on synthetic data. We design a controlled setting for evaluating different aspects of PEL inference. The ground-truth annotations of the 8 primitive events occurring in the test set of the basketball dataset are corrupted by four different types of noise. Then, these noisy annotations are input to PEL inference, as if they were obtained by running realistic detectors of the primitive events. In Fig. 3a, we start from the ground-truth time intervals, and randomly add an increasing number of new intervals of bogus primitive events (false positives). In Fig. 3b, we start from the ground-truth time intervals, and randomly remove an increasing number of them (false negatives). In Fig. 3c, we randomly change the duration of ground-truth intervals, but do not change their labels. Note that Figs. 3a-c simulate realistic noise in tracking, where some tracks might be wrongly split (or merged) into subtracks (or larger tracks), some parts of the tracks might be missing, and the track ID's might be wrongly switched. As can be seen, PEL inference gracefully degrades as tracking noise increases, due to the joint reasoning over multiple constraints in the PEL KB. This suggests that we can handle imperfect tracking. In this paper, we use a semi-supervised tracker to focus on a number of other contributions. We do not completely ignore the vision problem, as we work with noisy detectors and intervals. The experiment in Fig 3d differs from the previous cases, since we use as input to PEL inference real responses of the detector of Sec. 6, but we gradually remove an increasing number of Type 2 and Type 3 PEL formulas from the PEL KB (see Appendix). Fig 3d shows that the PEL interpretation score decreases, since it depends on the number, and type of formulas in the KB. As can be seen, PEL inference gracefully degrades as domain knowledge becomes scarce.

Quantitative results – Basketball: Tab. 1 presents the detection results obtained by PEL inference on the basketball test sequences. Fig 4 shows two confusion matrices—one contains results of the primitive detector, and the other contains detection results after PEL inference. We can see that PEL inference improves the detector's noisy results.

Quantitative results – Olympic sports: Table 7 compares our average video classification accuracy with that of [13]. We treat the Olympic sports classes as higher-level, hidden events in the PEL KB. We specify as primitive events, simple short-term actions, such as *walk*, *Run*, *jump*, *bend*, *throw*, *stand-up*, etc. Since the events are performed by a single athlete, we do not use the tracker, but directly apply the detector, described in Sec. 6, to detect these primitive events. The detector is trained on 10 short sequences for each primitive event taken from the dataset. The formulas in the PEL KB corresponding to the 16 higher-level events (e.g., *long jump*) are specified as a *meet* sequence of the primitives events. Table 7 shows that we outperform the state of the art [13].

8. Conclusion

We have formulated probabilistic event logic (PEL), which uses weighted event-logic formulas to represent arbitrary probabilistic constraints among events in terms of time intervals. An efficient MAP inference for PEL has been presented for detecting and localizing all event occurrences in a new video. The inference algorithm directly operates over special data structures, called spanning intervals. The complexity of these operations does not depend on the extent of

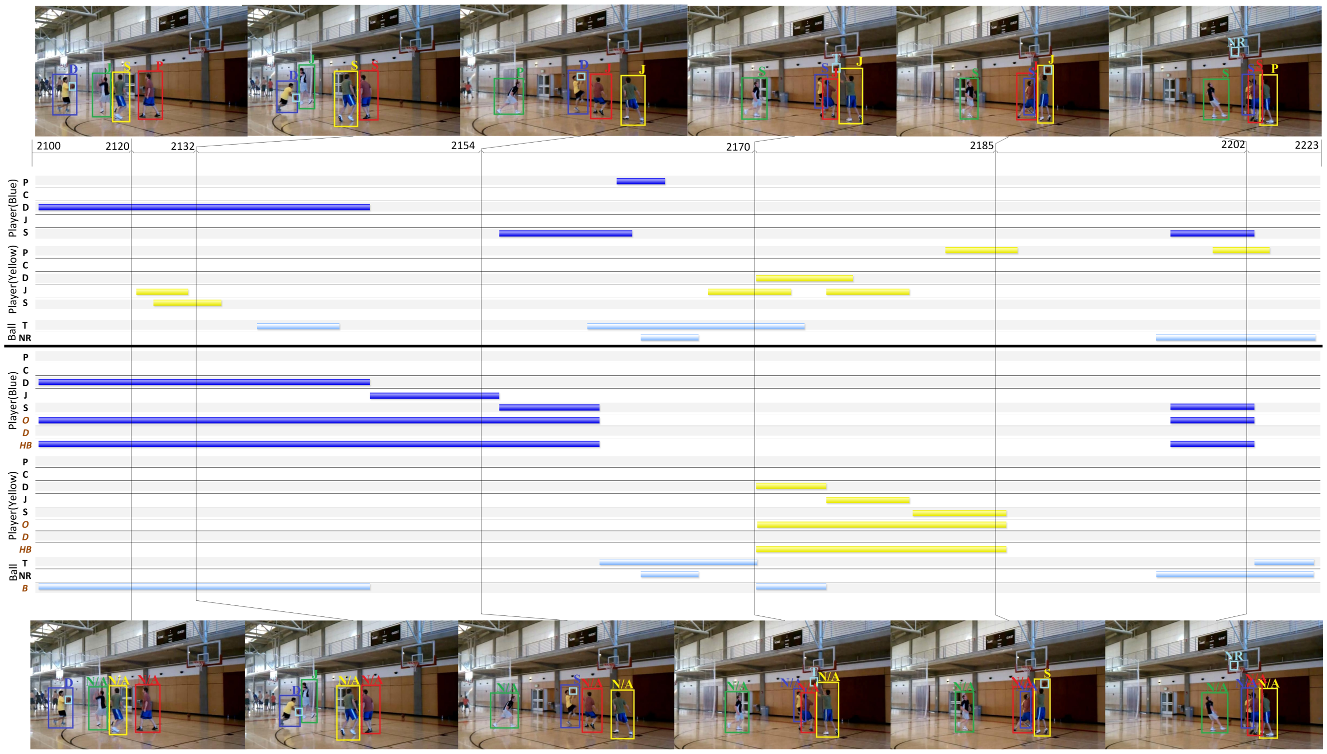

Figure 5. An example sequence from our basketball dataset: (top two rows) Only a subset of results of the tracker and primitive-event detector—each player's ID is marked with unique color, and detected primitive events are denoted with their name's first letter. (bottom two rows) Only a subset of results of PEL inference. PEL resolves ambiguities about exact occurrence and duration of each event, and improves event detection over the primitive detector, due to holistic reasoning about soft and hard constraints over time intervals in the PEL knowledge.

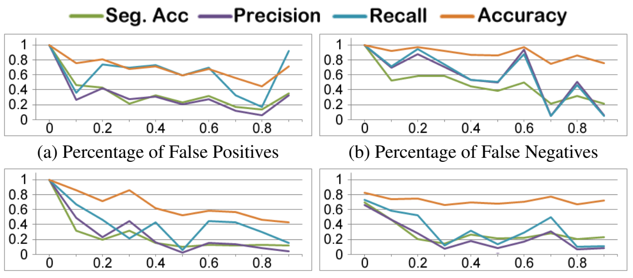

Figure 3. PEL inference under a controlled amount of noise (horizontal axis) on the basketball test videos: (a) increasing the number of false positives, (b) increasing the number of false negatives, (c) noise in durations of the event intervals, (d) removing formulas from the PEL KB. For (a), (b) and (c) the input set of observations for PEL inference is the set of ground truth event intervals corrupted by noise. For (d) the input to PEL inference are primitive-event detections from the real detector of Sec. 6. (best viewed in color)

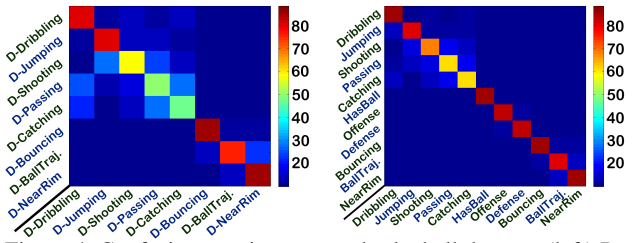

Figure 4. Confusion matrices on our basketball dataset. (left) Results of the primitive-event detector. (right) PEL inference. PEL reduces errors of the primitive-event detector.

time intervals, which are hypotheses of event occurrences during the inference, but rather only on the much smaller number of spanning intervals. We have presented successful detection and localization of inter-related events in basketball videos with severe occlusions and dynamic backgrounds. We compare favorably with the state of the art on the benchmark Olympic sports videos. PEL efficiently reasons about many events and their time intervals, and thus is highly scalable.

Appendix

Table 3 lists a subset PEL formulas that we use in our experiments for the basketball domain.

Sport class	Our	[13]	[10]
high-jump	70.1%	68.9%	52.4%
long-jump	75.3%	74.8%	66.8%
triple-jump	66.4%	52.3%	36.1%
pole-vault	85.5%	82.0%	47.8%
gymnastics-vault	87.9%	86.1%	88.6%
shot-put	65.4%	62.1%	56.2%
snatch	70.8%	69.2%	41.8%
clean-jerk	85.6%	84.1%	83.2%
javelin-throw	78.3%	74.6%	61.1%
hammer-throw	78.9%	77.5%	65.1%
discus-throw	60.4%	58.5%	37.4%
diving-platform	91.5%	87.2%	91.5%
diving-springboard	81.8%	77.2%	80.7%
basketball-layup	80.2%	77.9%	75.8%
bowling	75.8%	72.7%	66.7%
tennis-serve	62.4%	49.1%	39.6%
Average classification accuracy	76.0%	71.1%	62.0%

Table 2. Average video classification accuracy on the Olympic Sports Dataset [13]. We define primitive events, such as “Walk”, “Run”, “Jump”, “Bend”, “Throw”, etc., and specify the formulas of the 16 sports classes as a *meet* sequence of the primitives events.

Type 1:
D-Dribbling(x) $\rightarrow$ Dribbling(x)
D-Jumping(x) $\rightarrow$ Jumping(x)
D-Shooting(x) $\rightarrow$ Shooting(x)
D-Passing(x) $\rightarrow$ Passing(x)
D-Catching(x) $\rightarrow$ Catching(x)
D-Bouncing(x) $\rightarrow$ Bouncing(x)
D-BallTrajectory(x) $\rightarrow$ BallTrajectory(x)
D-NearRim(x) $\rightarrow$ NearRim(x)
ExactlyOne(Defense(x),Offense(x))
Shooting(x) $\rightarrow$ Offense(x)
HasBall(x) $\rightarrow$ ExactlyOne(Dribble(x),Shooting(x),Passing(x))
(Dribble(x) $\vee$ Shooting(x) $\vee$ Passing(x)) $\rightarrow$ HasBall(x)
HasBall(x) $\rightarrow$ $\neg$BallTrajectory
Dribbling(x) $\leftrightarrow$ Bouncing
Type 2 of the form $(E_1 \wedge \ldots E_n) \rightarrow \Diamond_r(E_1 \vee \ldots E_k)$ for $r \in \{m, mi, fi, f\}$:
Shooting(x) $\rightarrow \Diamond_{mi}$ (Shooting(x) $\vee$ BallTrajectory)
Passing(x) $\rightarrow \Diamond_{mi}$ (Passing(x) $\vee$ BallTrajectory)
Catching(x) $\rightarrow \Diamond_{mi}$ (Catching(x) $\vee$ HasBall(x))
Catching(x) $\rightarrow \Diamond_{m}$ (Catching(x) $\vee$ $\neg$HasBall(x))
(HasBall(x) $\wedge$ Jumping(x)) $\rightarrow \Diamond_{mi}$ (Jumping(x) $\vee$ ShootBall(x))
(HassBall(x) $\wedge$ Jumping(x)) $\rightarrow \Diamond_{mi}$ (Jumping(x) $\vee$ $\neg$HasBall(x))
HasBall(x) $\rightarrow \Diamond_{fi}(\Diamond_{mi}$(HasBall(x)) $\vee \Diamond_{fi}$(Passing(x) $\vee$ Shooting(x)))
Type 3 of the form $(E_1; \ldots; E_n) \rightarrow \Diamond_r(E \vee (E_1; \ldots; E_k))$ for $r \in \{m, mi\}$:
Shooting(x) $\rightarrow \Diamond_{mi}$ (Shooting(x) $\vee$ (BallTrajectory ; NearRim))
(BallTrajectory ; NearRim) $\rightarrow \Diamond_{m}$ (BallTrajectory $\vee$ Shooting(x))
(BallTrajectory ; Catching(x)) $\rightarrow \Diamond_{m}$ (BallTrajectory $\vee$ Passing(x))

Table 3. Different types of PEL formulas we use for the basketball domain. The user learning curve for entering PEL formulas in the system is similar to other languages for expressing knowledge.

Acknowledgement

The support of the National Science Foundation under grant NSF IIS 1018490 is gratefully acknowledged.

References

[1] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *J. Logic Comput.*, 4(5), 1994.

[2] M. Collins. Discriminative training methods for hidden Markov models: Theory and experiments with the perceptron algorithm. In *EMNLP*, 2002.

[3] D. Damen and D. Hogg. Recognizing linked events: Searching the space of feasible explanations. In *CVPR*, 2009.

[4] A. Fern. A penalty-logic simple-transition model for structured sequences. *Computational Intelligence*, 25(4):302–334, 2009.

[5] A. Fern, R. Givan, and J. Siskind. Specific-to-general learning for temporal events with application to video event recognition. *JAIR*, 17:379–449, 2002.

[6] A. Gupta, P. Srinivasan, J. Shi, and L. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *CVPR*, 2009.

[7] R. Hamid, S. Maddi, A. Bobick, and I. Essa. Structure from statistics: Unsupervised activity analysis using suffix trees. In *ICCV*, pages 1–8, 2007.

[8] Y. Ivanov and A. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE TPAMI*, 22(8):852–872, 2000.

[9] F. Jurie and M. Dhome. Real time robust template matching. In *BMVC*, 2002.

[10] I. Laptev. On space-time interest points. *IJCV*, 64:107–123, 2005.

[11] G. Medioni, I. Cohen, F. Bremond, S. Hongeng, and R. Nevatia. Event detection and analysis from video streams. *IEEE TPAMI*, 23(8):873–889, 2001.

[12] R. Nevatia, J. Hobbs, and B. Bolles. An ontology for video event representation. In *Detection and Recognition of Events in Video, CVPRW*, 2004.

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

[14] A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell. Hidden conditional random fields. *IEEE TPAMI*, 29:1848–1852, 2007.

[15] N. Rota and M. Thonnat. Activity recognition from video sequences using declarative models. In *ECAI*, 2000.

[16] M. S. Ryoo and J. K. Aggarwal. Spatio-temporal relationship match: Video structure comparison for recognition of complex human activities. In *ICCV*, 2009.

[17] V. D. Shet, D. Harwood, and L. S. Davis. Multivalued default logic for identity maintenance in visual surveillance. In *ECCV*, pages 119–132, 2006.

[18] V. D. Shet, J. Neumann, V. Ramesh, and L. S. Davis. Bilattice-based logical reasoning for human detection. In *CVPR*, 2007.

[19] J. Siskind. Grounding lexical semantics of verbs in visual perception using force dynamics and event logic. *JAIR*, 15:31–90, 2001.

[20] S. D. Tran and L. S. Davis. Event modeling and recognition using Markov logic networks. In *ECCV*, 2008.

[21] T. Xiang and S. Gong. Beyond tracking: Modelling activity and understanding behaviour. *IJCV*, 67(1):21–51, 2006.

[22] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4):13, 2006.