Markov Logic Networks for Scene Interpretation and Complex Event Recognition in Videos

Atul Kanaujia
ObjectVideo, Inc.
11600 Sunrise Valley Drive,
Reston,VA
atul.kanaujia@gmail.com

Ping Wang
ObjectVideo, Inc.
11600 Sunrise Valley Drive,
Reston,VA
pwang@objectvideo.com

Niels Haering
ObjectVideo, Inc.
11600 Sunrise Valley Drive,
Reston,VA
nhaering@objectvideo.com

ABSTRACT

Automatic extraction and representation of visual concepts and semantic information in scenes is a desired capability in surveillance operations. We target the problem of complex event recognition in network information environment, where lack of effective visual processing tools and incomplete domain knowledge frequently cause uncertainty in the datasets and consequently, in the visual primitives extracted from it. We employ Markov Logic Network (MLN) to address the task of reasoning under uncertainty. In this work we demonstrate use of MLN as a domain knowledge representation language that can be used for inferring complex events in real world. MLN is a knowledge representation language that combines domain knowledge, visual concepts and experience to infer simple and complex real-world events. MLN generalizes over the existing probabilistic models, including hidden Markov models, Bayesian networks, and stochastic grammars. The framework can be made scalable to support variety of entities, their activities and interactions that are typically observed in the real world. Experiments with real-world data in a variety of urban settings illustrate the mathematical soundness and wide-ranging applicability of our approach.

1. INTRODUCTION

We target the problem of visual event recognition in network information environment, where faulty sensors, lack of effective visual processing tools and incomplete domain knowledge frequently cause uncertainty in the data set and consequently, in the visual primitives extracted from it. Our framework is based on Markov Logic Network (MLN), that combines probabilistic graphical models and first order predicate logic (FOPL), to address the task of reasoning under uncertainty. MLN is a probabilistic logic framework to combine symbolic information obtained from visual processing modules, prior background domain knowledge and experience (observation history) to robustly infer simple and complex real-world events under uncertainty and ambiguous sensor data. MLN generalizes over the existing state-of-the-art probabilistic models, including hidden Markov models, Bayesian networks, and stochastic grammars. In this work we apply Markov Network based inference to a number of challenging problems in automated scene understanding. Markov Logic Networks (MLN) is a knowledge representation language well-suited for the task of complex event recognition for visual surveillance applications. The visual processing modules generate a symbolic description of what appears in the visual field, in terms of entities, scene elements and their inter-relations. MLN uses these as inputs to output a Markov Random Field (MRF)-based representation of the state of the real-world in terms of various predicates defined in its knowledge base. In this work we also propose heuristics for MLN rule formulation and hierarhcical processing to make MLN inference scalabile to longer videos and events involving multiple agents. MLN offers several advantages over other rule-based activity recognition methods. In addition to being a probabilistic framework, MLN allows ability to write more flexible rules with existential quantifiers over sets of entities. This allows greater expressive power of the domain knowledge used by MLN compared to other probabilistic rule based methods such as attribute grammars or dynamic Bayesian networks [23].

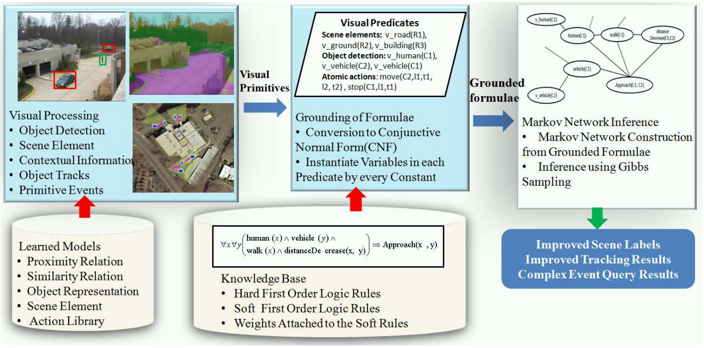

Visual Processing
• Object Detection
• Scene Element
• Contextual Informatio
• Object Tracks
• Primitive Events
Visual Primitives
Visual Predicates
Scene elements: v_road(R1), v_ground(R2), v_building(R3)
Object detection: v_human(C1), v_vehicle(C2), v_vehicle(C1)
Atomic actions: move(C2,l1,t1, l2, t2) , stop(C1,l1,t1)
Grounding of Formulae
• Conversion to Conjunctive Normal Form(CNF)
• Instantiate Variables in each Predicate by every Constant
Grounded formulae
v_human(C1)
human(C1)
walk(C1)
distance Decrease(C1,C2)
vehicle(C2)
Approach(C1, C2)
v_vehicle(C2)
Markov Network Inference
• Markov Network Construction from Grounded Formulae
• Inference using Gibbs Sampling
Learned Models
• Proximity Relation
• Similarity Relation
• Object Representation
• Scene Element
• Action Library
∀x∀y(human (x) ∧ vehicle (y) ∧ walk (x) ∧ distanceDe crease(x, y)) ⇒ Approach(x , y)
Knowledge Base
• Hard First Order Logic Rules
• Soft First Order Logic Rules
• Weights Attached to the Soft Rules
Improved Scene Labels
Improved Tracking Results
Complex Event Query Results

Figure 1: Overview of the proposed Markov Logic Network based system for complex event modeling and recognition in a video

2. RELATED WORK

Traditionally Hidden Markov Models(HMM) [2], Propagation Nets (P-Net)[21] and other forms of Dynamic Bayesian Networks(DBN)[15][14] had been widely applied to event recognition. These models are trained using a set of annotated example videos to learn parameters and are restrictive in terms of number of actors and types of activities that can be modeled due to fixed structure of the model. Notable work among rule based activity modeling techniques is the probabilistic method based on multi-agent belief network for complex action detection proposed by Intille and Bobick[8]. The method dynamically generates belief network for recognizing complex action using the pre-specified structure that represents temporal relationships between the actions of interacting agents. Perse et. al[17] developed an activity template based technique to identify the semantic labels of an observed activity in a basketball game and proposed a similarity measure based on Levenstein distance to match symbols obtained from the trajectories of the players to the template symbols. More recent research has focused on stochastic grammers based event recognition such as [1, 6, 19, 12]. Gupta et al.[6] developed a storyline model using probabilistic grammars to dynamically infer relations between component actions and also learns visual appearance models for each actions using the weakly labeled video data. Ryoo and Aggarwal [19] modeled composite actions and interactions between agents using non-probbailistic Context Free Grammar(CFG). Sridhar et. al[22] developed an unsupervised method to identify component events of a complex activity by modeling interactions between subsets of tracks of entities as a relational graph that captured qualitative spatio-temporal relationships between these agents. Our work is based on Markov Logic Networks(MLN) that combines expert domain knowledge, expressed as first-order logic rules, with probabilistic logical inference to robustly identify composite events involving multiple interacting agents. In that respect, our work is similar to the methods proposed in [23, 20, 9, 13].Tran and Davis[23] developed a visual event modeling framework based MLN that addressed a wide range of uncertainities due to detection, missing observations, inaccurate logic rules and identity maintenance. Later works [20, 9, 13] further developed the MLN based systems to infer multi-agent activities, use domain knowledge to improve scene interpretation and incorporated Allen's interval logic to improve scalabilty of MLN inference. In comparison, event recognition techniques based on stochastic attribute grammars typically use simpler rules than Markov Logic Networks(MLN). For example, it is difficult to express rule such as "a moving vehicle has atleast one person in it" using generative grammars. Grammars do not allow existential quantifiers, which are needed in case of missing observations. Also methods to perform probabilistic inference are better understood for graphical models based MLN than for probabilistic grammars.

Contributions: In this work we apply MLN inference to challenging problems in video content analysis by incorporating domain knowledge to overcome uncertainties and drawbacks of the visual processing tools. Specifically: (a) We employ MLN to perform functional and semantic labeling of image regions using hand crafted FOPL rules ; (b) We use MLN as a fusion tool to merge target tracklets based on visual similarity scores and spatio-temporal proximity of the tracks of the target ; (c) We demonstrate automatic learning of knowledge base(KB) rules that represent spatial-temporal relationship between entities observed in the scene ; (d) We propose techniques to improve scalability of inference in MLN and apply it to recognize complex events in a long video.

3. MARKOV LOGIC NETWORK

In this work we exploit Markov Logic Network(MLN) to represent, analyze and recognize spatio-temporal interactions between various entities in the scene. Markov Logic Networks [4, 18] is a principled framework that combines probabilistic reasoning with first-order predicate logic (FOPL) to bridge semantic gaps between visual events detected in the scene and the events occuring in real world. The rules defined using only FOPL provide semantic descriptions of what is typically but not always observed during an event. These rules, that can be seen as hard constraints, therefore may not generalize well to realistic scenarios where they are occasionally violated due to uncertainty in observing different constituent sub-events. Markov Logic Network is an elegant combination of FOPL and probabilistic reasoning that relaxes these constraints and allows convenient modeling of complex events as soft rules rather than as hard constraints. Violation of these rules do not make the occurrence of the events impossible but less probable. The task of finding variable states that satisfy the FOPL rules therefore get transformed as finding a truth assignment of the predicates that maximizes overall probability of the grounded Markov network. The domain knowledge is represented as a set of FOPL rules in MLN that has an associated weight,which can be thought of as the assertion strength of the rule.

Our MLN framework has following primary components:

- **Visual Processing:** These modules feed grounded predicates to the MLN in the form of constants denoting space-time locations of entities detected in the scene, scene element and entity classification and primitive events directly inferred from visual tracks of the entities. The constants are used to ground(instantiate) the variables in FOPL formulae of the MLN.

- **Knowledge Base (KB)**: KB is composed of a set of hard and soft rules modeling spatio-temporal interaction between entities and temporal structure of various complex events. The hard rules are assertions that should be strictly followed. Violation of hard rules sets the probability of the complex event to zero. On the other hand, soft rules allow uncertainty and exceptions. Violation of soft rules will make the complex event less probable but not impossible.

- **Markov Network (MN)**: MN is a Markov Random Field, generated by instantiating (referred to as grounding) the variables in KB rules from the constants obtained from visual entities and primitive (atomic) events detected in the scene. KB can be thought as template for constructing the Markov network. For every set of constants (detected visual entities and atomic events) observed in a scene, the FOPL rules involving the corresponding variables are instantiated to form the Markov network. Each node in MN represents either a grounded predicate or an inferred predicate. An edge exist between two nodes if the predicates appear in a formulae.

The rules defined in the Knowledge Base (KB) represent common sense domain knowledge of relation between etities that is typically observed in a scene. These are mostly hand crafted but can also be learned from the training data. After grounding, inference is run in MLN to compute probabilities of each of the nodes denoting predicates. Each rule with grounded atoms corresponds to a clique in MN. Since MN is a Markov Random Field (MRF), the joint distribution over all the predicates is computed as product of potentials defined over these cliques.

$$P(\mathbf{W} = x) = \frac{1}{Z} exp \left(\sum_j w_j f_j(x) \right) = \frac{1}{Z} exp \left(\sum_j w_j n_j(x) \right)$$

where $\mathbf{W}$ represents a possible truth assignment to the grounded predicates of the MLN and n_j denotes the number of cliques (instantiated formulae f_j) with truth value 1. $\mathbf{W}$ is also referred to as possible world satisfying constraints represented by the rules in KB. If the rule j with weight w_j is satisfied for a given set of constants, the world (corresponding to the given truth assignment) is $exp(w_j)$ times more probable than when the rule j is not satisfied. For every grounded formula (clique) f_j with truth value 1, we add w_j to the exponential term. The grounded MN can be queried to recognize complex events based on the probability of the corresponding event predicate. Fig. 1 summarizes all the steps involved in running probabilistic inference using MLN.

Uncertainty and missing information due to failure of the visual processing module to detect a constituent sub-event should lower the probability of a complex event. We assume following four types of uncertainties in our framework:

- **Incomplete or missing information**: A complex event A may be composed of a number of sub-events $\{B_1, B_2, \cdots, B_N\}$. The following rule takes care of the uncertainty in non-observation of any of these sub-events:

  $$\mathbf{w} : \mathbf{B}_1 \wedge \mathbf{B}_2 \wedge \mathbf{B}_3 \cdots \mathbf{B}_N \Rightarrow \mathbf{A} \quad (1)$$

  The sub-events are usually the events inferred from other rules or those that are directly observed and provided as predicates from the visual processing module.

- **Imperfect first order predicate logic rule**: Logical rules may not always be true. Weights associated to the rules signify how accurately the rule is followed. If an FOPL rule is almost always followed, it will have higher weight. Less accurate rules will have lower weight.

- **Uncertainty in visual detection of entities**: MLN makes distinction between the visual observation of an entity and a symbolic presence of it in the real world. Visual detectors provide probabilistic weight of detecting a target using classification based approach. The detected target is denoted by a predicate *v_appear(human)*. Actual appearance of the target in real world is denoted by the predicate *appear(human)*. The following two rules in the knowledge base take care of uncertainty in the visual detectors themselves:

  $$\mathbf{W}_1 : \quad P_D \quad v\_appear(human) \Rightarrow appear(human)$$
  $$\mathbf{W}_2 : \quad 1 - P_D \quad v\_appear(human) \Rightarrow appear(human)$$

  Where P_D is the calibrated detection probability obtained from the discriminative object detectors.

- **Uncertainty in maintaining identity of an entity**: Inaccurate visual processing module may cause tracking gaps between two entities. This is handled by both vision based predicate and inferred knowledge base (KB) rules. The visual predicate is defined to assess visual appearance similarity measure between any two entities across time based on image features. Additional rules are included in KB to establish identity relation between entities using spatial-temporal proximity.

  $$atLoc(A_1, Loc_1, Time_1) \wedge atLoc(A_2, Loc_2, Time_2) \wedge (Loc_1 = Loc_2) \wedge nearbyTime(Time_1, Time_2) \Rightarrow equal(A_1, A_2)$$

  This rule uses spatial-temporal proximity to establish identity relation between entities A_1 and A_2. In addition, appearance similarity cues can as well be employed to establish identity of target (see Section 5).

Alchemy[3] is an open-source system that provides a number of algorithms for statistical relational learning and probabilistic logic inference, based on the Markov logic representation. Alchemy also supports structure learning and hybrid Markov Logic Networks. In our work, we employ Alchemy for learning and inference using Markov Logic Network (MLN). In the rest of the sections, we apply MLN to perform a variety of complex tasks for video content analysis. These include semantic scene labeling (section 4), track merging (section 5) and complex event detection (section 7) in videos. In addition, section 6 discusses a novel application of MLN in discovering semantic rules denoting relations between entities in a scene.

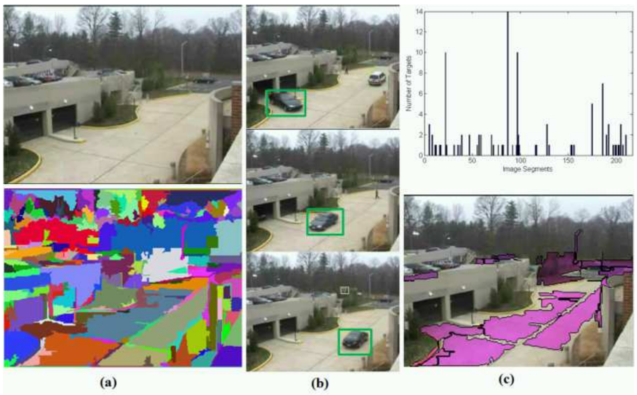


Figure 3: MLN formulation to improve ground plane labels in the image,(a) Original input image and segmentations using [5]; (b) sample images of target footprints used for improving confidence of ground image regions; (c)(top) Frequency plots of the image regions containing target footprints; (c)(bottom) Image regions having high frequency of target footprints.

4. SCENE INTERPRETATION

Markov Logic network can be directly applied to improve scene element labeling based on contextual relation between the scene elements and the entities. Rules that are true in general such as *human and vehicle footprints are more likely to be on a ground plane* and *agents can disappear only if they go out of scene or at an entrance of a building* can be used to refine the scene element classification and functional labeling of various image regions. In this scenario MLN acts as fusion tool that integrates low-level visual scene element classification results with the high level

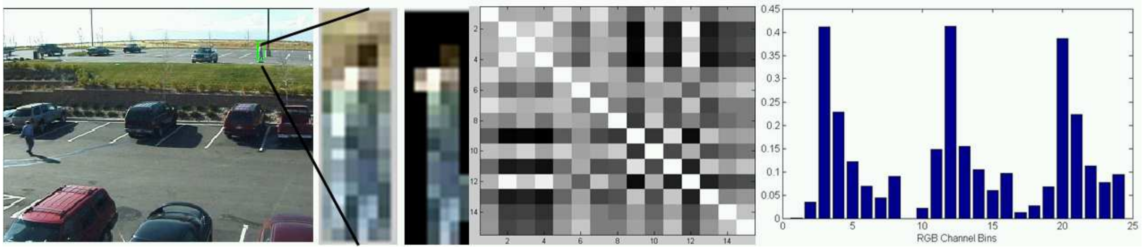


Figure 2: Appearance descriptor computation from a target tracklet. For a detected target, we first mask out the background and compute RGB histogram of the foreground pixels with 8 bins for each color channel, normalized by the number of frames. The 24 dimensional vector is used to compute similarity between the tracklets of the agents. The affinity matrix is obtained for all the target tracklets (15) detected in the video

domain knowledge rules for improved scene understanding. In this section we apply MLN inference to the problem for semantic and functional labeling of image regions. We use domain knowled to improve ground plane labeling and identify image regions corresponding to vehicle garage exits in the video. Our visual processing module uses [5] to perform image segmentation. We use the surface layout algorithm proposed by Hoiem *et.* al [7] to classify the image segments as $C = \{SKY, VERTICAL, HORIZONTAL\}$ regions. For each image segment, the algorithm provides scene element classification probability.

Groundplane Refining: The scene element classification provided by [7] are often inaccurate. We incorporate this uncertainty in the MLN framework by defining extensional predicates denoting real class of the image regions as *zoneClass(z)*. The visual predicates generated by the low-level visual processing module is denoted as *zoneVertical(z), zoneHorizontal(z)* and *zoneSky(z)*. For each of the image region z, we add three rules with weights corresponding to the probability value as provided by the scene element classifier. So if an image region z_1 is classified as $VERTICAL$ we add the rule:

$$\mathbf{W}_1 \quad zoneVertical(z_1) \Rightarrow zoneClass(z_1, VERTICAL)$$
$$\mathbf{W}_2 \quad zoneVertical(z_1) \Rightarrow zoneClass(z_1, HORIZONTAL)$$
$$\mathbf{W}_3 \quad zoneVertical(z_1) \Rightarrow zoneClass(z_1, SKY)$$

We incorporate domain knowledge by adding rules based on the known fact that a zone whose adjacent regions are classified to category C, has high probability that it also belongs to class C. We write another rule to incorporate the common sense to classify a region as groundplane if it has high footprints of moving targets like humans and vehicles. To do so we generate the predicate $zoneFootPrint(A_1, Z_1)$ if agentA_1 has footprints lying in the zone Z_1. We compute a histogram of zones that have most number of footprints. Occasional errors in visual processing generates spurious targets in the image sequence. Hence we use a threshold on number of footprints to generate predicates $zoneFootPrint(A_1, Z_1)$ for the zone. Figure 3 shows the image segments (highlighted) that have high frequency of footprints based on the histogram (also shown in the figure) for each zone. The adjacency relation between pair of zones, $zoneAdjacentZone(Z_1, Z_2)$, is computed by first computing whether the two segments lie near to each other (based on distance between the centroids). If they do, they are adjacent if together they form a single connected component. The soft rules to improve the confidence of the zone classification in MLN framework are:

$$\mathbf{W}_1 \quad zoneFootPrint(A_1, Z_1) \Rightarrow zoneClass(Z_1, HORIZONTAL)$$
$$\mathbf{W}_2 \quad zoneAdjacentZone(Z_1, Z_2) \wedge zoneFootPrint(A_1, Class) \Rightarrow zoneClass(Z_1, Class)$$

The weights are chosen based on subjective confidence in the rules and usually have $\mathbf{W}_1 > \mathbf{W}_2$.

Identifying Vehicle Garage Exits: We also apply MLN framework for semantic labeling for image regions, based on the spatio-temporal characteristics of the targets in the scene. An image zone is a garage exit if it is a vertical structure, vehicles appear in the region and it is not a boundary zone. Both $zoneVertical(\mathbf{Z}_1)$ and $vehicle(\mathbf{Z}_1)$ predicates are generated from the visual processing module. The predicates $zoneBoundary(\mathbf{Z}_1)$ are generated by classifying an image segment as boundary zone if it contains boundary pixels of the image. Similarly, the predicates $zoneAppear(\mathbf{A}_1, \mathbf{Z}_1)$ are generated for zone z1 every time the footprint of an appearing vehicle $\mathbf{A}_1$ is in that image segment. The rule for detecting a vehicle garage exit in an image is formulated as:

$$\mathbf{W} \quad vehicle(V_1) \wedge zoneAppear(A_1, Z_1) \wedge\ !zoneBoundary(Z_1) \wedge zoneClass(Z_1, VERTICAL) \Rightarrow zoneVehicleGarageExit(Z_1)$$

Figure 4 shows the image regions detected as garage exits.

5. TRACKS ASSOCIATION

In a realistic surveillance scenario, targets often undergo partial or full body occlusion and appearance changes due to varying illumination. Low-level target tracking modules with inaccurate motion models cannot handle these uncertainties and may split a single target track into multiple tracklets belonging to different target IDs. Our visual tracking module is based on Kalman filtering that uses motion dynamics and shape (aspect ratio of the bounding boxes) matching to track targets across consecutive time frames. However, target data association is using only the state of the target in the previous timestep (Markov assumption in Kalman Filtering). MLN provides a flexible framework to incorporate diverse similarity measures based on spatio-temporal prximity and low-level visual cues that cannot be not modeled in the tracking algorithm due to Markov assumption. We formulate MLN rules to apply common sense knowledge for target track association in a video sequence.

We define proximity or similarity measure over these trackets to merge multiple tracklets into single track. In our video sequences the size of the targets ranged from 10×10 pixels for the humans to 40×40 pixels for the vehicle. Sophisticated shape and appearance based similarity measures are futile at this resolution. We therefore used a simple RGB histogram (8 bins for each color channel) to compute appearance descriptor of the target bounding boxes (see figure 2). Background subtraction is used to mask out irrelevant regions from the bounding box. The histogram is computed for all the frames in the tracklets and normalized to sum to

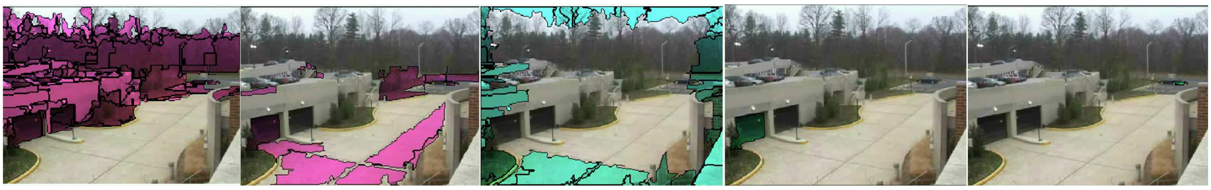


Figure 4: (a) Image regions where ever any agent appears ; (b) Vertical regions in the image obtained from visual predicate $zoneClass(\cdots)$; (c) Boundary zones of the image; (d) Image region correctly identified as vehicle garage exit ;(e) False detection

1 for each channel independently. We use the normalized histogram intersection as the similarity measure between the target tracklets. We define $v\_equal(A_1, A_2)$ predicate that represents observed appearance similarity between the two targets in the video. In addition, we define extensional predicate $equal(A_1, A_2)$ denoting whether the two agents are same in real world. We introduce four rules for each pair of the targets in the video as follows:

$$\begin{array}{ll}\mathbf{W}_1 & disappear(A_1, LocTime_1) \Lambda\ appear(A_2, LocTime_2) \\ & \Lambda\ timeLE(LocTime_1, LocTime_2) \\ & \Lambda\ nearBy(LocTime_1, LocTime_2) \Rightarrow equal(A_1, A_2) \\ \mathbf{W}_2 & class(A_1, Type_1)\ \Lambda\ class(A_2, Type_2) \\ & \Lambda\ !(Type_1 = Type_2) \Rightarrow !equal(A_1, A_2)\end{array}$$

The first rule associates two targets if they disappear and reappear shortly at image regions that is close in image. The second soft rule is for two targets belonging to the same class has a higher chance of being the same. Notice that we give a small weight to this rule ($\mathbf{W}_1 > \mathbf{W}_2$) as this rule is often violated. The affinity matrix in figure 2, denotes the similarity score, based on the normalized RGB histogram intersection, between the targets detected in the entire sequence. In our formulation we generate predicates with variable $LocTime_1 = (X_1, Y_1, Time_1)$ denoting a space-time point.

Predicate, (Observed/Inferred)	Description
noContact(A,B) (Observed)	True iff the intersection of bounding boxes A and B is empty
smallerY2Eq(A,B) (Observed)	True iff the bottom edge of bounding box A has a Y-value smaller than or equal to that of bounding box B
closeY2(A,B) (Observed)	True iff the bottom edges of bounding boxes A and B are within a few pixels of each other(greater than threshold)
occludedBy(A,B) (Inferred)	True if bounding box A is occluded by bounding box B in the image
further(A,B) (Inferred)	True if bounding box A is further from the camera than bounding box B

Table 1: Observed and inferred predicates used for structure learning

$(R1)$	$smallerY2Eq(a, b)\ \Lambda\ !close(Y2(a, b) \Rightarrow further(a, b)$
$(R2)$	$!noContact(a, b)\ \Lambda\ smallerY2Eq(a, b)\ \Lambda\ !close(Y2(a, b) \Rightarrow occludedBy(a, b)$
$(R3)$	$!noContact(a, b)\ \Lambda\ further(a, b) \Rightarrow occludedBy(a, b)$

Table 2: We generate the synthetic bounding boxes using the above ground truth rules that model the depth ordering relation between the two bounding boxes.

6. RULES MINING IN VIDEOS

Rules minimg refers to identifying a set of rules/clauses that is typically followed in a relational model. In MLN, rule mining refers to structure learning for discovering relations between the predicates or refine an existing MLN. Existing implementation of structure learning in Alchemy uses Hypergraph lifting that optimizes Weighted Pseudo Log-Likelihood (WPLL) to search in the space of plausible clauses and identify most likely rules in the knowledge base [10, 11]. Each clause has an associated weight denoting the how likely it is followed in the data. Structure learning starts with either a predfined set of clauses or an empty set, and generates most probable clauses by adding or deleting literals in all possible forms (logical connectives) that share variables with the clause.

We apply the structure learning to a specialized task of inferring rules for 3D spatial relations between objects in a scene. The 3D spatial relation (depth ordering) between any two objects is inferred using the relative location of 2D bounding boxes obtained from the object detector. Given locations of the two bounding boxes, the box with lower y co-ordinates of the bottom line is nearer and occludes the other if the two bounding boxes overlap or are in contact. However nothing can be said is they do not occoverlap. The reasoning applied to this framework assumes that the images have negligible camera tilt and line of intersection of ground plane and image plane is horizontal. We have two types of predicates: the ones that can be observed, and the ones that will be inferred based on observations. For learning, we provide observations and ground truth of $occludedBy(\cdots)$ and $further(\cdots)$ from 20 synthetic images, each containing 3 randomly positioned blocks. We have three relevant observable predicates: $noContact(\cdots)$, $smallerY2Eq(\cdots)$ and $closeY2(\cdots)$ as described in Table 1. We also add six additional irrelevant predicates on spatial relationships, $smallerY1Eq$, $closeY1$, $smallerX1Eq$, $closeX1$, $smallerX2Eq$ and $closeX2$ in order to test whether the structure learning can find the most relevant evidence for reasoning $occludedBy(\cdots)$ and $further(\cdots)$ in the presence of irrelevant data. Table 2 shows the groundtruth rules that holds true in this scenario. Figure 6 shows the three most weighted rules inferred using structure learning in MLN. For structure learning we start with an empty Knowledge Base(KB) and provide grounded predicates for both the observed and inferred predicates generated from the synthetic data. The rules obtained from structure learning are infact the reformulation of the groundtruth rules shown in table 2. Note that the weights are negative and the negation of the above clauses are the rules learned by structure learning. First inferred rule states that two bounding boxes a1 and

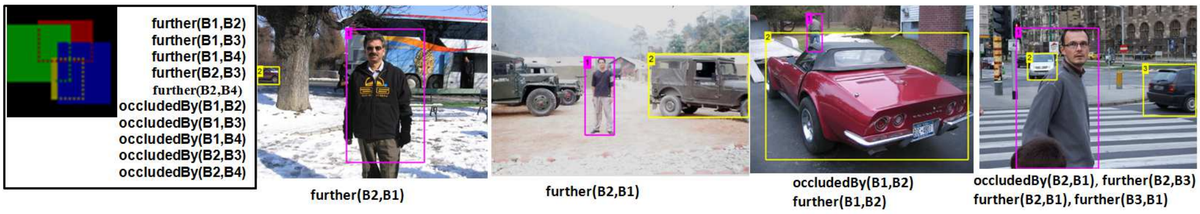


Figure 5: Depth ordering for the bounded boxes of various detected targets in synthetic and real images using the rules discovered via structure learning in Markov Logic Network. The leftmost image shows the depth ordering predicates inferred for the synthetic test image containing 4 randomly sampled bounding boxes. Here B1 is red, B2 is green, B3 is yellow and B4 is blue bounding boxes in the image. The bounding boxes in the test real images are numbered.

a2 that are not in contact with each other do not occlude. This is true in general. Second rule is that for two bounding boxes a1 and a2, with bottom edge of a1 lower than a2, a1 is not occluded by a2 and a1 is also not further from a2. Third rule states that for bounding box a1 and a2 that are close to each other along Y co-ordinates, they are neither occluded nor further from each other. Both the statements are true for visual entities detected in an image.

```
-12.7   !noContact(a1,a2) v !noContact(a2,a1) v occludedBy(a2,a1)
-15.7   smallerY2Eq(a1,a2) v !smallerY2Eq(a2,a1)
        v occludedBy(a1,a2) v further(a1,a2)
-13.7   occludedBy(a1,a2) v further(a1,a2) v !closeY2(a1,a2)
        v !closeY2(a2,a1)
```

Figure 6: Rules inferred using Structure learning in Markov Logic Network. The netgative value of the weights denotes that negation of the correspodning clause is the inferred rule

7. COMPLEX EVENT MODELING

We apply Markov Logic Networks (MLN) to the task of complex events detections in the larger videos containing multiple targets instances interacting with each other and the environment. In the current framework, we assume that the uncertainty is due to missing information or inaccuracy of the rule formulations. The visual predicates obtained from low-level and mid-level information fusion modules are assumed to be true. Figure 1 shows the architecture of MLN based event recognition framework.

Markov Logic Network (MLN) has been criticized for lack of scalability and tractability when a large number predicates and instances of the variables are present in the scene. This is primarily due to large number of atoms generated in the MN after the grounding step, causing inference to become intractable. For the inference, all the observed targets are used to instantiate all the variables in all the predicates used in the FOPL rules of the Knowledge Base(KB). Longer rules that involve a large number of variables often cause combinatorial explosion in the number of possible cliques formed in the grounded Markov Network. This also makes inference computationally expensive in time and resources. Hence longer rules should as well be avoided in MLN formulations. Rather, it is more efficient to split the rule into multiple rules evaluating different component of the event. It is therefore more tractable to apply MLN inference to applications where numerous low-level spatial-temporal relations between two or more agents has already been pre-computed or provided by the underlying visual processing modules.

MLN Rule formulation: We follow a set of guidelines for formulating MLN rules in Alchemy to improve inference in larger videos. We use hard constraints and rules to reduce combinatorial explosion during inference. An interpretation of the world violating a hard constraint has zero probability and can be readily eliminated. A predicate is evaluated for all possible instantiations of all of its variables. Hence use of predicates of high arity should be avoided in rules formulations. Instead of forming a single long rule with many predicates, we form multiple shorter rules with fewer predicates. This forms smaller cliques in the network and fewer nodes. We do not define the domain of space and time variables. This will cause a large number of instantiation of the predicates during inference. We may also treat points in spatio-temporal domain rather than separately in space and time. For example, the arity of the predicate $move(A,\ LocX1,\ LocY1,\ Time1,\ LocX2,\ LocY2,\ Time2)$ will get reduced for the predicate $move(A{,}LocX1\_Y1\_Time1,\ LocX2\_Y2\_Time2)$. We treat time interval as one unit rather than treating it with a start and end time. This further reduces the arity of the predicates $move(A,\ LocX1\_Y1,\ LocX2\_Y2,\ Int\_Time1\_Time2)$. This greatly reduces the computation complexity of the inference step. We precompute the predicates like $nearby(\cdots)$ as hard constraints precomputed out of Alchemy. For example, if we decide we use a threshold of 5 units as the distance to classify points that are nearby, we will have $nearby5(P1, P1)$ as sample grounded predicates for all possible instances of $P1$ and $P2$.

Hierarchical Inference: Most of the simpler events involving one or two agents such as approaching, meeting, embarking and disembarking vehicle, are based on only temporally local interactions between the agents. More complex events such as loading and unloading an object from the vehicle, is composed of many component simple events occurring in, typically, a partially ordered temporal relation to one another, subject to certain logical constraints. For analyzing long videos, we therefore divide a long temporal sequence into multiple smaller sub-windows. In principle, we can design a hierarchy of MLN inference for analyzing and recognizing complex events at different time intervals and different temporal scale. Events detected by MLN in these temporal windows are thresholded and used to generate predicates for inference at next higher level of the hierarchy using MLN. Since most of the simpler events exhibit strong locality constraint, they get detected in the lower level of the hierarchy. The detected predicates are passed on to higher layer for detecting events involving long term dependencies and that extending across multiple temporal windows. MLN at higher level fuse the information from the lower level inferred predicates to make decisions. Analyzing the events within these overlapping sub-windows does not affect accuracy of event detection but however requires structural modification and manual specification of what simple predicates should be passed to next higher level.

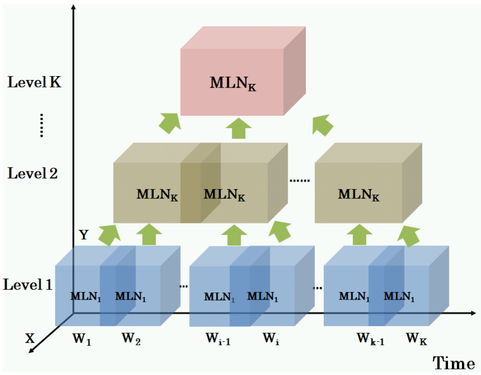


Figure 7: Hierarchical splitting of a temporal sequence into multiple overlapping windows. Each of the box shown represents an MLN that fuses information from the MLNs from lower levels. Section 7.1 discusses an example of a multi-level MLN for inferring a complex activity.

7.1 Event Modeling and Recognition

The first step in formulating an MLN based framework for complex event modeling in videos is to define the constants and ground atoms that will be generated by the visual processing module. The visual processing module detects and tracks two classes of agents in the scene - vehicles and humans. The spatio-temporal attributes of the agents - time and location, are the key variables in the event recognition. We define event predicates in terms of spatial location as a constant pair either in (X, Y) co-ordinates in image frame or on ground map if image to map homography is available. Time is represented as a constant or time intervals using two constants as starting time and end time. Visual processing module generates groundings (instances) at fixed time intervals by detecting and tracking the targets in video. These ground atoms include: (a) target footprint location and time ; (b) target classification $human(A_1)$ or $vehicle(V_1)$ and ; (c) primitive events $appear(A_1, Loc, Time)$, $disappear(A_1, Loc, Time)$, $move(A_1, LocS, TimeS, LocE, TimeE)$ and $stationary(A_1, Loc, TimeS, TimeE)$. The atomic events are detected from the tracks of the objects. In addition we define predicates for assessing space-time proximity between events in the video as $nearBy(\cdots)$,$close(\cdots)$, $shortlyAfterInt(\cdots)$ and $timeLE(\cdots)$ We list in table 3 various multi-agents events recognized by our MLN system. In order to detect if human is approaching a vehicle we check whether the distance between it and the vehicle is decreasing. If the homography is available, we compute the angle between motion direction vector and human and vehicle displacement vector to detect whether motion is towards the vehicle or not. Without homography, we compute the same in image co-ordinates and assume fronto-parallel camera viewpoint.

We applied MLN inference to detect a complex activity involving 4 agents - two humans, one object(box) and one vehicle. The activity involves an object being delivered by a human agent to another human agent using a vehicle. The first order predicate logic rule to recognize the activity is formulated using multiple extensional predicates defined to detect constituent sub-events. The composite activity of object delivery are assumed to have the constituent sub-events detected in the following partial temporal order:(a) A vehicle parks in a parking lot ; (b) A human disembarks from the vehicle; (c) Another human standing near the vehicle unloads an object from the vehicle; (d) the second human agent carries the object away from the vehicle ; (e) The first human agent embarks the vehicle ; (f) the vehicle drives away from the parking lot. The rule formulation of the object delivery activity is shown in the table 4 and uses a number of extensional predicates defined in table 3.

Event Predicate	Description about the Event
parkVehicle(A)	Identifies a vehicle arriving in the parking lot and stopping in the subsequent time intervals
driveVehicle(A)	Identifies a stationary vehicle that starts moving in the subsequent time intervals
meeting(A,B)	Two human A and B meet if both of them are stationary and are near to each other in space and at overlapping times
embark(A,B)	Happens when human A comes near vehicle B in space and disappears after which vehicle B starts moving
disembark(A,B)	Disembark event happens when a human target appears close to a stationary vehicle target
approach(A,B)	This event occurs when distance between human A and vehicle B decreases and the direction motion is along the displacement of vehicle from human
personNear-Vehicle(A,B)	Happens when human A is close in space to a vehicle B
personCarry-Object(A,B)	Happens when human A and object B move at same location to each other in space and time
loadObjectTo-Vehicle(A,B,C)	Identifies sequence of events where human A carries object B to location near vehicle C. Following this the object B disappears
unload-ObjectFrom-Vehicle(A,B,C)	Identifies sequence of events where human A is stationary near vehicle C and object B appears. Following this human A walks away from the vehicle C with the object B
personCarry-Object-Away(A,B,C)	Identifies a sequence of events where a human A carries object B and moves away from the vehicle C

Table 3: Predicates representing various inferred multi-agent events occurring in the video

Seq No.	No. of Vehicles	No. of Humans	No. of Frames	SD/ HD
*Seq*1	9	4	3600(4 min)	SD
*Seq*2	0	28	1772(1 min, 58 sec)	SD
*Seq*3	15	5	3923(4 min, 20 sec)	SD
*Seq*4	10	9	15,000(10 min)	HD
*Seq*5	3	6	20,760(12 min)	HD

Table 5: Description of videos used for evaluating our MLN based system

8. EXPERIMENTS

We ran experiments on both standard resolution and high-definition videos. Table 5 summarizes the videos used in the study.

Data Description and Visual Processing Module: The sequences *Seq*1, *Seq*2 and *Seq*3 were captured in standard definition at resolution of 320×240. We used standard background modeling techniques to detect targets in these videos.

$$parkVehicle(A_3, Loc_1, TimeInt_1) \wedge disembark(A_1, A_3, TimeInt_2, Time_1) \wedge shortlyAfterInt(TimeInt_1, TimeInt_2) \wedge$$
$$unloadObjectFromVehicle(A_2, A_3, A_4, TimeInt_3) \wedge shortlyAfterInt(TimeInt_2, TimeInt_3) \wedge personCarryObjectAway(A_2, A_3, -$$
$$A_4, TimeInt_4) \wedge shortlyAfterInt(TimeInt_3, TimeInt_4) \wedge embark(A_1, A_3, Time_2, Time_3) \wedge withinInterval(TimeInt_4, Time_2) \wedge$$
$$driveVehicle(A_3, TimeInt_5) \wedge shortlyAfterInt(TimeInt_4, TimeInt_5) \Rightarrow objectDelivery(A_1, A_2, A_3, A_4, TimeInt_5)$$

Table 4: First order predicate logic (FOPL) rules for detecting object delivery activity involving 4 agents and a sequence of sub-events identified by the inferred predicates discussed in table 3

Objects were tracked using Kalman filtering and classified as humans or vehicles based on aspect ratio of the bounding box. Sequences $Seq4$ and $Seq5$ are high definition videos of resolution 1280 × 720 from VIRAT dataset[16].

Semantic Scene Labeling: Our MLN based semantic scene label refinement is evaluated on the video sequence $Seq3$. We ran image segmentation[5] to generate a total of 217 segments(zones) in the image. We first classify the regions into various geometric classes as $C = \{SKY$, *HORIZONTAL* , *VERTICAL*$\}$ using appearance and geometric cues computed over these image segments. The footprints of the targets detected in the image regions are used to improve the confidence of groundplane labels (classified as $HORIZONTAL$) as described in section 4. Figure 9 shows the refined labels ground plane obtained from MLN inference. In order to identify image regions corresponding to vehicle garage exits in the scene, we extract image regions that do not lie on the image border and that are classified as vertical with high frequency of vehicles appearing. Figure 4(two figures on the right) shows the image regions classified as garage exits. The rightmost figure is the false detection obtained from MLN inference.

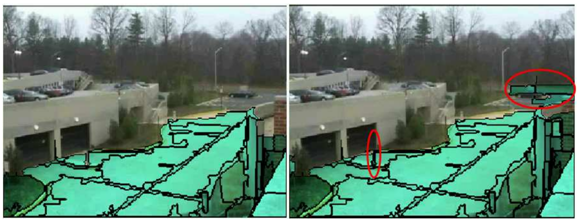


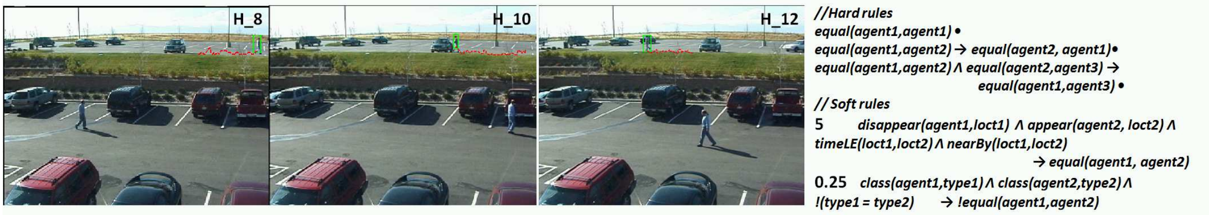
Figure 9: (Left) Ground plane regions obtained from the contextual scene element classification algorithm proposed by Hoiem et. al[7] *(Right)* Refined labels obtained by incorporating domain knowledge rules using MLN

Track Association: Figure 8 shows the results of track merging in the video sequence $Seq1$. For merging two tracks, we specified the spatial proximity threshold to be 5 pixels and temporal proximity to be 3 seconds. These thresholds denote that two tracklets can only be merged if end point of one and starting point of the other are close to each other in time and space. We manually labeled the number of true track merges based on the threshold. For the specific video sequences, there were 5 test instances when the tracks should be merged of which 3 mergings were detected and no false detections were observed.

Structure Learning: We evaluated our structure learning framework by training on the synthetic bounding boxes and testing on the randomly sampled bounding boxes as well as on real dataset. Figure 5(left) shows an example test case obtained by randomly sampling 4 bounding boxes(B1-red, B2-green, B3-yellow, B4-blue) and the inferred relationship between the two, based on the co-ordinates of the bounding boxes. We observed more than 98% accuracy in all the experiments on synthetic data. To evaluate our framework on real data set, we ran person and car detector on images from PASCAL VOC 2010 dataset, and perform depth order reasoning ($occludeBy(A, B)$ and $further(A, B)$) using the rules discovered (by structure learning) in the dataset of observed predicates. Figure 5 shows some of the results on the real images from the dataset.

Complex Event Modeling: We evaluate our MLN based system for complex event recognition on the sequences listed in the table 5. As the time range of a complex event is losely defined for a complex event, we report the results using only the number of true positives and number of false detections of the events observed in a video. For tractability issues, MLN is evaluated only on the instances of time intervals it has seen in the grounded atoms obtained from the visual processing. The complex event is therefore always detected as the time interval of the last constituting sub-event of the complex event observed in the video. The visual predicates generated by the visual modules were fed in the MLN framework as evidence to recognize an event. The weights of our soft rules were manually set whereas the marginal probabilities of all the predicates were set to 0. The inferred predicates obtained from the MLN inference were sorted according to the probabilities and used for outputing the events. In all of our experiments, we ran 10000 MCMC iterations for inference. Table 6 summarizes the complex event detection results of our system.

We applied MLN inference to detect complex activity of $objectDelivery(\cdots)$ involving multiple interacting targets in the video sequence $Seq5$. In order to avoid high computational cost of MLN inference due to large number of predicates generated for the 20,000 frames, we adopt our hierarchical processing (discussed in section 7) to perform inference in MLN for complex activity recognition in the video. We split the video into overlapping temporal windows of 3600 frames each, with an overlap of 120 frames. We use time interval as a single variable composed of starting and ending time. The spatio-temporal predicates (denoting primitive events such as *move, appear, dissappear* and *stationary*) were generated at fixed intervals of 120 frames from the target tracks. We formulate MLN rules to recognize each of the above events in the video for all the temporal windows. The inferred predicates are fed as inputs to higher level MLN for making decision on the complex activity of $objectDelivery(\cdots)$ that extended over three temporal windows. The activity consists of six sub-events that should happen within short time interval of previous event. We use an additional threshold of 900 frames (30 seconds) to detect whether two time instances happen shortly after another. The MLN at higher level acts as a fusion framework inputting the predicates from the MLN inference at the lower level. We retain a predicate if its probability is higher than a pre-specified threshold $T = 0.67$. The inference of the long rule for $objectDelivery(\cdots)$ takes approximately two hours to complete and generates $\sim 400K$ possible ground



```
//Hard rules
equal(agent1,agent1) •
equal(agent1,agent2) → equal(agent2, agent1)•
equal(agent1,agent2) ∧ equal(agent2,agent3) →
                              equal(agent1,agent3) •
// Soft rules
5       disappear(agent1,loct1) ∧ appear(agent2, loct2) ∧
timeLE(loct1,loct2) ∧ nearBy(loct1,loct2)
                                        → equal(agent1, agent2)
0.25   class(agent1,type1) ∧ class(agent2,type2) ∧
!(type1 = type2)      → !equal(agent1,agent2)
```

Figure 8: Track association for the video sequence *Seq*1. Targets H_8, H_{10} and H_{12} are the same agent who moves out of view briefly, comes back to the view and undergoes full occlusion due to the vehicle. The red colored trajectories are target tracklets. The hard and soft rules of the MLN used for merging the tracklets are shown in the right

Events (no. of agents)	No. of Events	True Detections	False Detections
Track Merges	5	3	0
Park vehicle(1)	2	2	0
Drive Vehicle(1)	2	2	0
Meeting(2)	2	2	0
Embarking(2)	4	4	3
Disembarking(2)	5	5	0
Approach(2)	3	2	1
Person Carry Object(2)	5	5	0
Person Near Vehicle(2)	6	6	0
Loading Object to Vehicle(3)	2	1	0
Person Carry Object Away(3)	4	1	0
Unload Object from Vehicle(3)	2	1	0

Table 6: Event recognition results. There are high number of false detections for Embarking event. This is due to loss of tracking of the human target when it is near the vehicle. This triggers embarking event

atoms. This is due to very high arity of the predicate for $objectDelivery(\cdots)$ causing it to be evaluated for all possible combinations of grounded atoms.

9. CONCLUSION

In this work we investigated the use of Markov Logic Networks as a framework for knowledge representation, information fusion and decision making under uncertainty. We proposed techniques to make inference in Markov Logic Networks scalable to longer videos and support more complex multi-agent activities. We also applied MLN framework to the problem of semantic and functional labeling of image regions in a scene. We investigated structure learning using both clean and noisy data and observed that structure learning in MLN is able to learn the most relevant predicates in the presence of irrelevant predicates. This is a useful property of MLN that can be applied to problems such as knowledge discovery in unstructured texts, images or videos.

10. REFERENCES

[1] A. F. Bobick and Y. A. Ivanov. Action recognition using probabilistic parsing. In *CVPR*, pages 196–202, 1998.

[2] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *CVPR*, pages 994–999, 1997.

[3] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence.* Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2009.

[4] P. Domingos and M. Richardson. Markov logic: A unifying framework for statistical relational learning. *In Intro. to Statistical Relational Learning MIT Press, Cambridge,MA*, 2007.

[5] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[6] A. Gupta, P. Srinivasan, J. Shi, and L. S. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *CVPR*, pages 2012–2019, 2009.

[7] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *International Journal of Computer Vision*, 75(1):151–172, 2007.

[8] S. S. Intille and A. F. Bobick. Visual recognition of multi-agent action using binary temporal relations. In *CVPR*, pages 1056–, 1999.

[9] A. Kembhavi, T. Yeh, and L. S. Davis. Why did the person cross the road (there)? scene understanding using probabilistic logic models and common sense reasoning. *ECCV*, pages 693–706, 2010.

[10] S. Kok and P. Domingos. Learning the structure of markov logic networks. In *ICML*, pages 441–448, 2005.

[11] S. Kok and P. Domingos. Learning markov logic network structure via hypergraph lifting. In *ICML*, page 64, 2009.

[12] L. Lina, H. Gongb, L. Lic, and L. Wangd. Semantic event representation and recognition using syntactic attribute graph grammar. *Pattern Recognition Letters*, 30(2):180–186, 2009.

[13] V. I. Morariu and L. S. Davis. Multi-agent event recognition in structured scenarios. *CVPR*, pages 3289–3296, 2011.

[14] J. Muncaster and Y. Ma. Activity recognition using dynamic bayesian networks with automatic state selection. *WACV*, 59(2):39–47, 2007.

[15] R. Nevatia, T. Zhao, and S. Hongeng. Hierarchical language-based representation of events in video stream. *IEEE Proc. of CVPRW on Event Mining*, 59(2):39–47, 2003.

[16] S. Oh. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR*, pages 3153–3160, 2011.

[17] M. Perse, M. Kristan, S. Kovacic, G. Vuckovic, and J. Pers. A trajectory-based analysis of coordinated team activity in a basketball game. *Computer Vision and Image Understanding*, 113(5):612–621, 2009.

[18] M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

[19] M. S. Ryoo and J. K. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *CVPR (2)*, pages 1709–1718, 2006.

[20] A. Sadilek and H. A. Kautz. Recognizing multi-agent activities from gps data. In *AAAI*, 2010.

[21] Y. Shi, A. F. Bobick, and I. A. Essa. Learning temporal sequence model from partially labeled data. In *CVPR (2)*, pages 1631–1638, 2006.

[22] M. Sridhar, A. G. Cohn, and D. C. Hogg. Unsupervised learning of event classes from video. *AAAI*, pages 180–186, 2010.

[23] S. D. Tran and L. S. Davis. Event modeling and recognition using markov logic networks. *ECCV (2)*, pages 610–623, 2008.

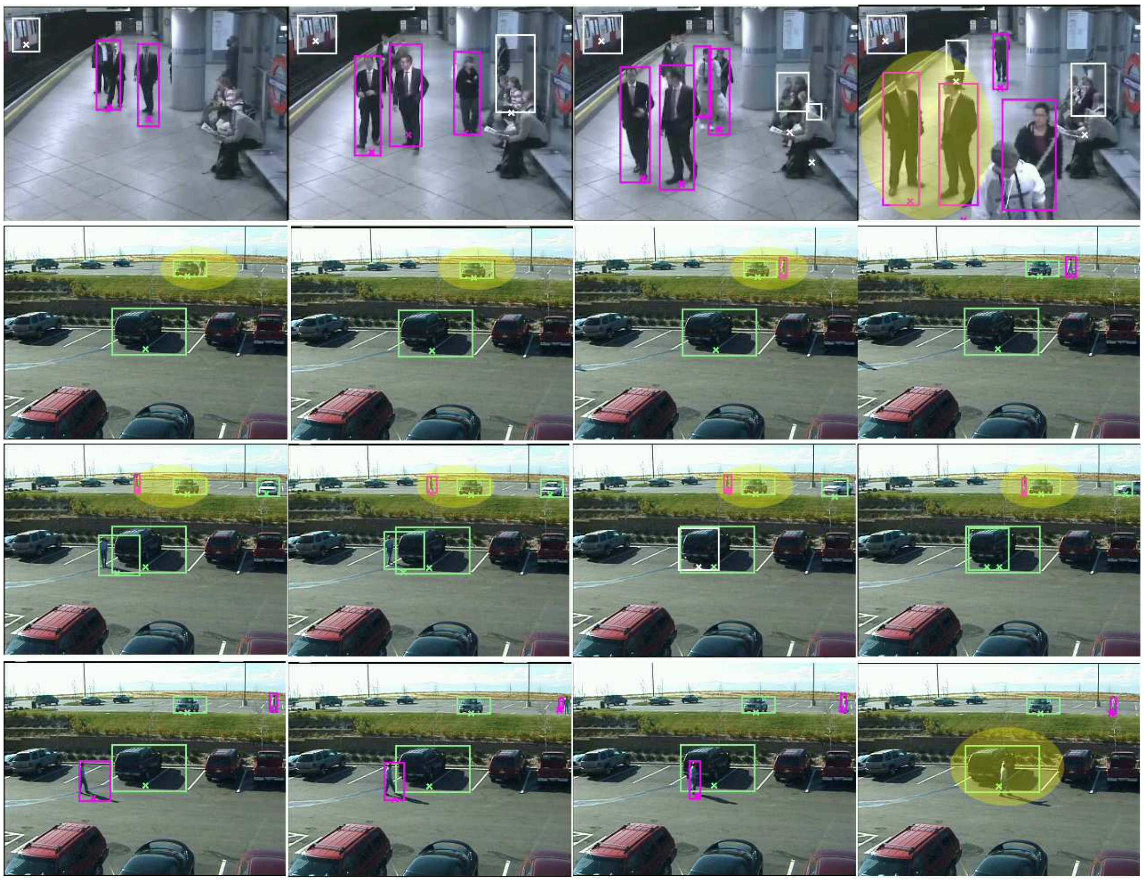


Figure 10: (Top row) Image Snapshot of the meeting event. The meeting event happens only when the two agents are stationary and sufficiently close to other for more than minimum time ; *(Second row)* Disembark event correctly recognized by our system ;*(Third row)* Approach event correctly detected using MLN,*(Fourth row)* False detection of embark event due to failure of target detection

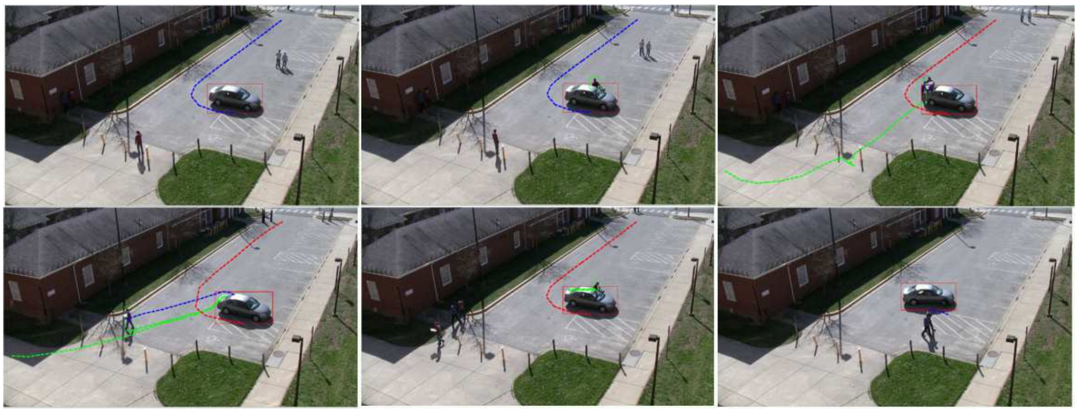


Figure 11: (Top Left) Detection of the event $parkVehicle(\cdots)$; *(Top Middle)* Disembarking event detected in the video ; *(Top Right)* Unloading event is detected as there are two humans near the vehicle, when the box object appears near the vehicle location ; *(Bottom Left)* Object unloading event involving a human carrying object away from the vehicle; *(Bottom Middle)* Embarking event detected in the video. The track associated to the bounding boxes is due to the agent moving towards the vehicle trunk and coming back to embark the vehicle ; *(Bottom Right)* $driveVehicle(\cdots)$ event is recognized when the vehicle is stationary in current time interval and starts moving in the successive time step