Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-10)

Recognizing Multi-Agent Activities from GPS Data

Adam Sadilek and **Henry Kautz**
Department of Computer Science
University of Rochester
Rochester, NY 146 27
{sadilek, kautz}@cs.rochester.edu

Abstract

Recent research has shown that surprisingly rich models of human behavior can be learned from GPS (positional) data. However, most research to date has concentrated on modeling single individuals or aggregate statistical properties of groups of people. Given *noisy real-world* GPS data, we—in contrast—consider the problem of modeling and recognizing activities that involve *multiple related individuals playing a variety of roles*. Our test domain is the game of capture the flag—an outdoor game that involves many distinct cooperative and competitive joint activities. We model the domain using Markov logic, a statistical relational language, and learn a theory that jointly denoises the data and infers occurrences of high-level activities, such as capturing a player. Our model combines constraints imposed by the geometry of the game area, the motion model of the players, and by the rules and dynamics of the game in a probabilistically and logically sound fashion. We show that while it may be impossible to directly detect a multi-agent activity due to sensor noise or malfunction, the occurrence of the activity can still be inferred by considering both its impact on the future behaviors of the people involved as well as the events that could have preceded it. We compare our unified approach with three alternatives (both probabilistic and nonprobabilistic) where either the denoising of the GPS data and the detection of the high-level activities are strictly separated, or the states of the players are not considered, or both. We show that the unified approach with the time window spanning the entire game, although more computationally costly, is significantly more accurate.

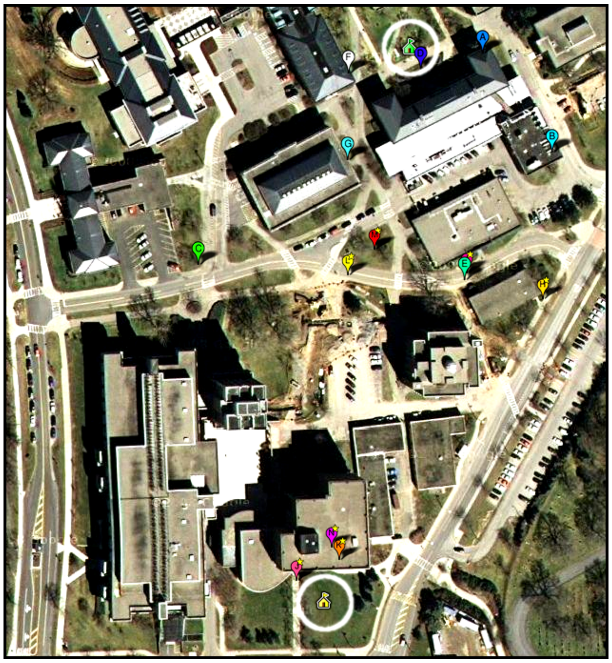

Figure 1: A snapshot of a game of capture the flag that shows the game area. Players are represented by pins with letters. In our version of CTF, the two "flag areas" are stationary and are shown as white circles near the top and the bottom of the figure. The horizontal road in the middle of the image is the territory boundary. The data is shown prior to any denoising or corrections for map errors. Videos of our recorded games are available from the first author's website.

Introduction

Motivation

Imagine two teams—seven players each—playing capture the flag on a university campus, where each player carries a consumer-grade global positioning system (GPS) that logs its location every second (see Fig. 1). Accuracy of the GPS data varies from 1 to more than 10 meters. In open areas, readings are typically off by 3 meters, but the discrepancy is much higher in locations with tall buildings or other obstructions. The error has a systematic component as well as a significant stochastic component. Errors between devices are poorly correlated, because subtle differences between players, such as the angle at which the device sits in the player's pocket, can dramatically affect accuracy. Moreover, since we consider multi-agent scenarios, the errors in individual players' readings can add up, thereby creating a large discrepancy between the reality and the recorded dataset. Because players can move freely through open areas, we cannot reduce the data error by assuming that the players move along road or walkways, as is done in much work on GPS-based activity recognition (e.g., (Liao, Fox, and Kautz 2004)). Finally, traditional techniques for denoising GPS data, such as Kalman filtering, are of little help, due to the low data rate (1 sample per second) relative to the small amount of time required for a player to completely change her speed or direction.

Given such raw and noisy data, can we automatically and reliably detect and recognize interesting events that happen within the game (such as one player capturing another player)? Moreover, can we jointly denoise the GPS data and infer instances of game events? In this paper, we present an approach that provides evidence for answering a resounding "yes" to both of the above questions.

Our work is not primarily motivated by the problem of annotating strategy games, although there are obvious applications of our results to sports and combat situations. We are, more generally, exploring relational learning and inference methods for recognizing *multi-agent activities* from location data. We accept the fact that the GPS data at our disposal is inherently unreliable and ambiguous for any one individual. We therefore focus on methods that *jointly and simultaneously* localize and recognize the high-level activities of groups of individuals.

GPS location data can be used to learn (perhaps surprisingly) much about human behavior. Most work to date has concentrated either on inferring activities performed by a single person at a sequence of locations (Bui 2003; Liao, Fox, and Kautz 2004; 2005), or on how locations are typically used by anonymous groups of individuals (Abowd et al. 1997). By contrast, we are interested in using GPS data to discover particular activities that involve *multiple related individuals playing a variety of distinct roles*. For example, consider modeling a situation that includes an elderly person interacting with a circle of caregivers. GPS data could be used to infer various kinds of caregiving activities, such as an adult son taking his mother clothes shopping—an event that would be virtually impossible to capture if we considered each person in isolation.

Many different kinds of cooperative and competitive multi-agent activities occur in games. Because of the rich yet well defined nature of the domain, we had decided to begin our investigation of multi-agent activity recognition with GPS data we collected from people playing capture the flag (details of the data collection are below). The lowest-level joint activities are based on location and movement, and include "approaching" and "being at the same location." Note, that noise in the GPS data often makes it difficult or impossible to directly detect these simple activities. At the next level come competitive multi-agent activities including capturing and attacking; cooperative activities include freeing; and there are activities, such as chasing and guarding, that may belong to either category or to both categories. There are also more abstract tactical activities, such as making a sacrifice, and overall strategies, such as playing defensively. In this paper, we concentrate on activities at the first two levels.

Our Approach

We provide a unified framework for intelligent relational denoising of the raw GPS data while simultaneously labeling instances of a player being captured by an enemy. Both the denoising and the labeling are cast as a learning and inference problem in Markov logic. By denoising, we mean modifying the raw GPS trajectories of the players such that the final trajectories satisfy constraints imposed by the geometry of the game area, the motion model of the players, as well as by the rules and the dynamics of the game. In this paper, we refer to this trajectory modification as "snapping" since we tile the game area with 3 by 3 meter cells and snap each raw GPS reading to an appropriate cell. By creating cells only in unobstructed space, we ensure the final trajectory is consistent with the map of the area.

We express the constraints as weighted formulas in Markov logic (see section "Models" below). Some of the constraints are "hard," in the sense that we are only interested in solutions that satisfy all of them. Hard constraints capture basic physical constraints (e.g., a player is only at one location at a time) and inviolable rules of the game (e.g., a captured player must stand still until freed or the game ends).[1] The rest of the constraints are "soft," meaning there is a finite weight associated with each one. Some of the soft constraints correspond to a traditional low-level data filter, expressing preferences for smooth trajectories that are close to the raw GPS readings. Other soft constraints capture high-level constraints concerning when individual and multi-agent activities *are likely* to occur. For example, a soft constraint states that if a player encounters an enemy on the enemy's territory, the player is likely to be captured. The exact weights on the soft constraints are learned from labeled data, as described below. Fig. 2 gives an English description of our hard and soft constraints for the low-level movement and player capture rules of capture the flag.

The most likely explanation of the data is one that satisfies all the hard constraints while maximizing the sum of the weights of the satisfied soft constraints. Inference is done simultaneously over an entire game (on average, about 10 minutes worth of data). Note that we do not restrict inference to a (small) sliding time window. As the experiments described below show, many events in this domain can only be definitely recognized long after they occur. For example, GPS noise may make it impossible to determine whether or not a player has been captured at the moment of the capture, but as the player thereafter remains in place for a long time, the possibility of his capture becomes certain.

Significance of Results

We show that while it may be impossible to directly detect a multi-agent activity due to sensor noise or malfunction, the occurrence of the activity can still be inferred by considering both its impact on the future behaviors of the people involved as well as the likelihood of the events that potentially preceded it. Our experiments demonstrate that a fairly short theory in Markov logic (which follows directly from the rules of the game) coupled with automatically learned weights, can reliably recognize instances of capture events even at times when other approaches largely fail. Nine out ten captures were correctly identified in tens of thousands of GPS readings and in the presence of hundreds of almost-captures (situations that at first sight look like captures but

[1] Cheating could be accommodated by making the rules highly-weighted soft constraints rather than hard constraints.

Hard Rules:

H1. The final trajectory is consistent with the map of the game area (i.e., players don't magically walk through walls).

H2. Each raw GPS reading is snapped to exactly one cell.

H3. No player is captured at the beginning of the game.

H4. When player a_1 captures player a_2, then both involved players must be snapped to a common cell at that time.

H5. A player can be captured only by an enemy who is on his or her home territory.

H6. A player can be captured only when standing on enemy territory.

H7. A player can be captured only if he or she is currently free.

H8. A player transitions from a free state to a captured state via a capturing event.

H9. A captured player remains captured until freed.

H10. If a player is captured then he or she must remain in the same location.

Soft Rules:

S1. Minimize the distance between the raw GPS reading and the snapped-to cell.

S2. Minimize projection variance, i.e., two consecutive "snappings" should be generally correlated.

S3. Maximize smoothness (both in terms of space and time) of the final player trajectories.

S4. If players a_1 and a_2 are enemies, a_1 is on home territory, a_2 is on a_1's territory, a_2 is not captured already, and they are close to each other, then a_1 *probably* captures a_2.

S5. Capture events are generally rare, i.e., there are typically only a few captures within a game.

Figure 2: Description of the hard and soft rules for capture the flag.

after careful analysis turn out to be false alarms). Furthermore, we show that our approach is applicable even in sizeable domains and is easily decomposable and extensible.

Even though the capture the flag domain doesn't capture all the complexities of life, most of the problems that we are addressing here clearly have direct analogues in more real-life tasks that artificial intelligence needs to address—such as improving smart environments, human-computer interaction, surveillance, assisted cognition, and battlefield control.

Capture The Flag Domain

We collected the CTF dataset by having subjects play three games of capture the flag on a university campus while carrying basic GPS loggers. Players were required to remain outdoors. We manually labeled instances of capture events in all three games based partly on our notes taken during the experiment and partly on our best judgement. We consider our labeling to be the ground truth for both training our models and for evaluation purposes.

The visualization of a game is shown in Fig. 1. In our variant of CTF, we have two teams—each consisting of seven players. The south team's territory is the area south of the road in the middle of the map and analogously for the north team. The flags are actually stationary and are shown as white circles in Fig. 1. The goal is to enter the opponent's flag area within 15 minutes. Players can be captured only while on enemy territory by being touched by the enemy. Upon being captured, they must remain in place until freed or the game ends.

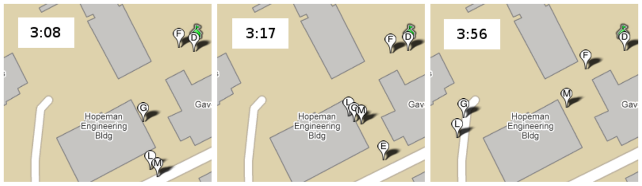

Figure 3: Three snapshots of a game situation that illustrate the need for an approach that exploits both the relational and the far reaching temporal structure of our domain.

If we are to reliably recognize interesting events that happen in these games, we need to consider not only each player individually but also the relationships among them over extended periods of time (possibly the whole length of the game). Consider a real game situation illustrated in Fig. 3. There we see three snapshots of a game projected over a map before any modification of the GPS data. The game time is shown on each snapshot. Players D, F, G are allies and are currently on their home territory near their flag, whereas players L and M are their enemies. In the first snapshot, players L and M head for the opponent's flag but then—in the second frame—they are intercepted by G. At this point it is unclear what is happening because of the substantial error in the GPS data—the three players appear to be very close to each other but in actuality they could have been 20 or more meters apart. However, once we see the third snapshot (note that tens of seconds have passed), in retrospect, we realize that player G actually captured only player M and didn't capture L since he is still chasing him. The fact that player M remains stationary coupled with the fact that neither D nor F attempts to capture him suggests that M has indeed been captured. Our unified model gives the correct labeling even for complex situations like these whereas limited approaches largely fail.

Background

Markov Logic

Given the inherent uncertainty involved in reasoning about real-world activities as observed through noisy sensor readings, we looked for a methodology that would provide an elegant combination of probabilistic reasoning with expressive, relatively natural, and compact but unfortunately strictly true or false formulas of first order logic (FOL). And that is exactly what Markov logic provides thereby allowing us to elegantly model complex relational non-i.i.d. domains (Richardson and Domingos 2006). A Markov logic network (MLN) consists of a set of constants $\mathcal{C}$ and of a set of pairs $\langle \mathcal{F}_i, w_i \rangle$ such that each FOL formula $\mathcal{F}_i$ has a weight $w_i \in \mathbb{R}$ associated with it.

Hard Formulas

$$\forall a_1, a_2, t : \text{capturing}(a_1, a_2, t) \Rightarrow \big(\text{enemies}(a_1, a_2) \wedge \text{onHomeTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t) \wedge \neg\text{isCaptured}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t)\big) \qquad \text{(H4–H7)}$$

$$\forall a_1, a_2, t : \text{samePlace}(a_1, a_2, t) \Rightarrow \big(\exists c : \text{snap}(a_1, c, t) \wedge \text{snap}(a_2, c, t)\big) \qquad \text{(H4)}$$

$$\forall a, t : \big(\neg\text{isCaptured}(a, t) \wedge \text{isCaptured}(a, t+1)\big) \Rightarrow \big(\exists_{=1} a' : \text{capturing}(a', a, t)\big) \qquad \text{(H8)}$$

Soft Formulas

$$\forall a, c, t : \big[\text{snap}(a, c, t)\big] \cdot d_1(a, c, t) \cdot w_p \qquad \text{(S1)}$$

$$\forall a, c_1, c_2, t : \big[\text{snap}(a, c_1, t) \wedge \text{snap}(a, c_2, t+1)\big] \cdot d_2(a, c_1, c_2, t) \cdot w_s \qquad \text{(S2)}$$

$$\forall a_1, a_2, t : \Big[\big(\text{enemies}(a_1, a_2) \wedge \text{onHomeTer}(a_1, t) \wedge \text{onEnemyTer}(a_2, t) \wedge \neg\text{isCaptured}(a_2, t) \wedge \text{samePlace}(a_1, a_2, t)\big) \Rightarrow \text{capturing}(a_1, a_2, t)\Big] \cdot w_c \qquad \text{(S4)}$$

$$\forall a, c, t : \big[\text{capturing}(a, c, t)\big] \cdot w_{cb} \qquad \text{(S5)}$$

Figure 4: Selected formulas in Markov logic. See corresponding constraints in Fig. 2 for an English description. ($\exists_{=1}$ denotes unique existential quantification.)

A MLN can be viewed as a template for a Markov network (MN) as follows: MN contains one node for each possible ground atom of MLN. Each weight w_i in the MLN intuitively represents the relative "importance" of satisfying (or violating, if the weight is negative) the corresponding formula $\mathcal{F}_i$. Thus the problem of satisfiability is relaxed in MLNs. We no longer search for a satisfying truth assignment as in traditional FOL. Instead, we are looking for a truth assignment that maximizes the sum of the weights of all satisfied formulas.

Maximum *a posteriori* inference in Markov logic given the state of the observed atoms reduces to finding a truth assignment to the hidden atoms such that the weighed sum of satisfied clauses is maximal. Even though this problem is in general #P-complete, we achieve reasonable run times by applying Cutting Plane MAP Inference (CPI) (Riedel 2008). CPI can be thought of as a meta solver that incrementally and partially grounds a Markov logic network thereby creating a Markov network that is subsequently solved by any applicable method—such as MaxWalkSAT or via a reduction to an integer linear program. After obtaining this (possibly preliminary) solution, CPI searches for additional grounding that could contribute to the score.

Models

Since to date no results on attacking a comparable multi-agent relational learning, denoising, and recognition problem have been published, we compare our unified approach with three alternative models. The first two models (**baseline** and **baseline with states**) are purely deterministic and they separate the denoising of the GPS data and the labeling of game events. We implemented both of them in Perl. They do not involve any training phase.

On the other hand, the **two-step model** and the **unified model** are probabilistic and are both cast in Markov logic. The unified model handles the denoising and labeling in a joint fashion whereas the two-step approach first performs snapping given the geometric constraints and subsequently labels instances of capturing. The latter two models are evaluated using three-fold cross-validation where in order to test on a given game, we first trained our model on the two other games.

Our models can access the following observed data: raw GPS position of each player at any time and indication whether they are on enemy or home territory, location of each 3 by 3 meter cell, cell adjacency, and list of pairs of players that are enemies. We tested all four models on the same raw GPS data. The following subsections describe each of the four approaches in more detail.

Baseline Model

This model has two separate stages. First we snap each reading to the nearest cell and afterward we label the instances of player *a* capturing player *b*. The labeling rule is simple: we loop over the whole discretized (via snapping) data set and output *capturing(a,b,t)* every time we encounter a pair of players *a* and *b* such that they were snapped (in the first step) to either the same cell or to two mutually adjacent cells at time *t*, they are enemies, and *a* is on its home territory while *b* is not.

Baseline Model with States

This second model builds on top of the previous one by introducing a notion that players have states. If player *a* captures player *b* at time *t*, *b* enters a captured state (in logic, *isCaptured(b,t+*1*)*). Then *b* remains in captured state until it moves (is snapped to a different cell at a later time) or the game ends. As per rules of CTF, a player who is in captured state cannot be captured again.

Thus, this model works just like the previous one except whenever it is about to label a capturing event, it checks the states of the involved players and outputs *capturing(a,b,t)* only if both *a* and *b* are *not* in captured state.

Two-Step Model

In the two-step approach, we have two separate theories in Markov logic. The first theory is used to perform a preliminary snapping of each of the player trajectories individually using constraints H1, H2, and S1–S3. The second theory then takes this preliminary denoising as a list of observed atoms in the form *preliminarySnap(a,c,t)* (meaning player *a* is snapped to cell *c* at time *t*) and uses the remaining constraints to label instances of capturing, while considering cell adjacency in the same manner as the baseline model. The two-step model constitutes a decomposition of the unified model (see below) and overall contains virtually the same formulas, thus we omit elaborating on it here.

Unified Model

In the unified approach, we express all the hard constraints H1–H10 and soft constraints S1–S5 in Markov logic as a single theory that jointly denoises the data and labels game events. Selected interesting formulas are shown in Fig. 4—their labels correspond to the listing in Fig. 2. Note that formulas S1 and S2 contain real-valued functions d_1 and d_2 respectively. (Markov logic networks that contain such functions are called *hybrid* MLNs.) d_1 returns the distance between agent *a* and cell *c* at time *t*. Similarly, d_2 returns the dissimilarity of the two consecutive “snapping vectors”[2] given agent *a*’s position at time *t* and *t*+1 and the location of the centers of two cells c_1 and c_2. Since w_p and w_s are both assigned negative values during training, formulas S1 and S2 effectively softly enforce the corresponding geometric constraints.

Experiments and Results

The raw data set contains missing data at a rate of approximately 1 in 250, while there are typically contiguous, several seconds long segments of readings missing. Since the players can move quite erratically, in this work we haven’t attempted to explicitly fill in the missing data. Instead we simply assume that a player whose data is missing remains at his or her last seen location. Adding extra formulas to our theory that weigh readings according to their corresponding logged signal quality, which would be lowest for missing data points, has been left for future work.

We apply theBeast software package to do weight learning and MAP inference in our domain. theBeast implements the cutting plane inference meta solving scheme and we use the integer linear program solver as the base solver (as opposed to MaxWalkSAT) since the resulting run times are still relatively short (under an hour even for training and testing the most complex unified model) and we gain exactness of the inference.

We specify the Markov logic formulas by hand and optimize the weights of the soft formulas via supervised on-line learning. We set theBeast to use Margin Infused Relaxed Algorithm (MIRA) for weight updates while the loss function is computed from the number of false positives and false negatives over the hidden atoms.

Table 1 lists for each game the number of raw GPS readings and the number of captures (ground truth), and summarizes our results in terms of precision, recall, and F1 score. We see that the unified approach yields the best results in each case. Overall, it labels 9 out of 10 captures correctly—there is only one false negative. In fact, this tenth capture event is missed by *all* the models because it involves two enemies that appear unusually far apart (about 12 meters) in the raw data. Even the unified approach fails on this instance since the cost of adjusting the players’ trajectories—thereby losing score due to violation of the geometry-based constraints—is not compensated for by the potential gain from labeling an additional capture. Such a situation is not present in the training data for game 3 and thus the weights on the capture recognition formulas are too low in this case.

	#GPS	#C	B	B+S	2-S$_{\mathrm{ML}}$	U$_{\mathrm{ML}}$
Game 1	13,412	2				
Precision			0.006	0.065	1.000	**1.000**
Recall			1.000	1.000	1.000	**1.000**
F1			0.012	0.122	1.000	**1.000**
Game 2	14,400	2				
Precision			0.006	0.011	1.000	**1.000**
Recall			0.500	0.500	0.500	**1.000**
F1			0.013	0.022	0.667	**1.000**
Game 3	3,472	6				
Precision			0.041	0.238	1.000	**1.000**
Recall			0.833	0.833	0.833	**0.833**
F1			0.079	0.317	0.909	**0.909**

Table 1: Summary of the dataset and the results. #GPS denotes the number of raw GPS readings, #C is the number of actual captures, B denotes the baseline model, B+S is the baseline model with states, 2-S$_{\mathrm{ML}}$ and U$_{\mathrm{ML}}$ are the two-step and the unified Markov logic models, respectively.

Note that even the two-step approach recognizes 8 out of 10 captures. It misses one instance in which the involved players are moderately far apart are snapped to mutually nonadjacent cells. On the other hand, the unified model does not fail in this situation because it is not limited by prior nonrelational snapping to a few nearby cells.

Both baseline models perform very poorly, although they yield a respectable recall. They produce an overwhelming amount of false positives ranging from 121 to 332 for the baseline model and from 21 to 89 for the augmented baseline model. This validates our hypothesis that we need to exploit the rich relational and temporal structure of the domain in a probabilistic way.

Related Work

Previous work heavily focused on denoising single-agent GPS traces and subsequent activity recognition (Limketkai, Fox, and Liao 2007; Liao, Fox, and Kautz 2004). Those authors cast the problem as learning and inference in a conditional random field and a dynamic Bayesian network respectively.

The most studied multi-agent activity is undoubtedly *conversation*. Mobile phone data (call and location logs) have been used to infer social networks and user mobility patterns (Eagle and Pentland 2006). However, only a relatively small number of activities can be inferred from cell phone

[2] The initial point of each snapping (projection) vector is a raw GPS reading and the terminal point is the center of the cell we snap that reading to.

data (e.g., conversation, text message, etc.), and GPS information has only an indirect impact on the particular joint activity, since the participants are necessarily not in the same location. A recent effort to collect data on face-to-face conversation along with GPS data (Wyatt et al. 2007) might well be useful for inferring location-based multi-agent activities, but results published to date from that study have not made use of location information.

Recent work on relational spacial reasoning includes an attempt to locate—using spacial abduction—caches of weapons in Iraq based on information about attacks within a geographic area (Shakarian, Subrahmanian, and Spaino 2009).

Finally, a number of researchers in machine vision have worked on the problem of recognizing events in videos of sporting events, such as impressive recent work on learning models of baseball plays (Gupta et al. 2009). Most work in that area has focused on recognizing individual actions (e.g., catching and throwing), and the state of the art is just beginning to consider relational actions (e.g., the ball is thrown from player *a* to player *b*). The computational challenges of dealing with video data make it necessary to limit the time windows of a few seconds; by contrast, we demonstrated above that many events in the capture the flag data can only be disambiguated by considering arbitrarily long temporal sequences. In general, however, both our work and that in machine vision rely upon similar probabilistic models, and there is already some evidence that Markov logic-type relational models can be used for activity recognition from video (Tran and Davis 2008; Biswas, Thrun, and Fujimura 2007).

Conclusions and Future Work

We presented a novel methodology—cast in Markov logic—for effectively combining data denoising with higher-level relational reasoning about a complex multi-agent domain. Experiments on real GPS data validate our approach while leaving an open door for future (incremental) additions to the ML theory.

We compared our unified model with three alternatives (both probabilistic and nonprobabilistic) where either the denoising of the GPS data and the detection of the high-level activities are strictly separated, or the states of the players are not considered, or both. We showed that the unified approach with the time window spanning the entire game, although more computationally costly, is significantly more accurate.

We are currently extending our framework in three directions. The first focuses on recognizing a richer set of game events of various types outlined in the introduction (such as freeing, chasing, hiding, cooperation, failed attempts at an activity, . . .). The events are often tied together and thus recognizing one of them improves the performance on the others (e.g., imagine adding freeing recognition to our theory). The second extension is built on top of the denoising and recognition model and performs reinforcement learning to learn game tactics while exploiting the higher-level information inferred by the base model. Finally, we explore casting our activity recognition and denoising problem as inference in logical hidden Markov models (LHMMs), a generalization of standard HMMs that compactly represents probability distributions over sequences of logical atoms (Kersting, De Raedt, and Raiko 2006).

Acknowledgements

We thank the anonymous reviewers and Wendy Beatty for their constructive comments; and Sebastian Riedel for his help with theBeast. This work was supported by ARO grant #W911NF-08-1-0242, DARPA SBIR Contract #W31P4Q-08-C-0170, and a gift from Kodak.

References

Abowd, G. D.; Atkeson, C. G.; Hong, J.; Long, S.; Kooper, R.; and Pinkerton, M. 1997. Cyberguide: a mobile context-aware tour guide. *Wirel. Netw.* 3(5):421–433.

Biswas, R.; Thrun, S.; and Fujimura, K. 2007. Recognizing activities with multiple cues. In *Workshop on Human Motion*, 255–270.

Bui, H. H. 2003. A general model for online probabilistic plan recognition. In *IJCAI-2003*.

Eagle, N., and Pentland, A. 2006. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing* 10(4).

Gupta, A.; Srinivasan, P.; Shi, J.; and Davis, L. 2009. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *CVPR09*.

Kersting, K.; De Raedt, L.; and Raiko, T. 2006. Logical hidden Markov models. *J. Artif. Int. Res.* 25(1):425–456.

Liao, L.; Fox, D.; and Kautz, H. 2004. Learning and inferring transportation routines. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*.

Liao, L.; Fox, D.; and Kautz, H. 2005. Location-based activity recognition using relational Markov networks. In *IJCAI-2005*.

Limketkai, B.; Fox, D.; and Liao, L. 2007. CRF-filters: discriminative particle filters for sequential state estimation. In *IEEE International Conference on Robotics and Automation*, 3142–3147.

Richardson, M., and Domingos, P. 2006. Markov logic networks. *Mach. Learn.* 62(1-2):107–136.

Riedel, S. 2008. Improving the accuracy and efficiency of MAP inference for Markov logic. In *UAI-08*, 468–475.

Shakarian, P.; Subrahmanian, V.; and Spaino, M. L. 2009. SCARE: A case study with baghdad. In *ICCCD-2009*.

Tran, S., and Davis, L. 2008. Visual event modeling and recognition using Markov logic networks. In *Proceedings of the 10th European Conference on Computer Vision*.

Wyatt, D.; Choudhury, T.; Bilmes, J.; and Kautz, H. 2007. A privacy-sensitive approach to modeling multi-person conversations. In *IJCAI-2007*.