Conditional Random Fields for Activity Recognition

Douglas L. Vail
Computer Science Dept.
Carnegie Mellon University
Pittsburgh, Pennsylvania
dvail2@cs.cmu.edu

Manuela M. Veloso
Computer Science Dept.
Carnegie Mellon University
Pittsburgh, Pennsylvania
veloso@cs.cmu.edu

John D. Lafferty
Computer Science Dept.
Carnegie Mellon University
Pittsburgh, Pennsylvania
lafferty@cs.cmu.edu

ABSTRACT

Activity recognition is a key component for creating intelligent, multi-agent systems. Intrinsically, activity recognition is a temporal classification problem. In this paper, we compare two models for temporal classification: hidden Markov models (HMMs), which have long been applied to the activity recognition problem, and conditional random fields (CRFs). CRFs are discriminative models for labeling sequences. They condition on the entire observation sequence, which avoids the need for independence assumptions between observations. Conditioning on the observations vastly expands the set of features that can be incorporated into the model without violating its assumptions. Using data from a simulated robot tag domain, chosen because it is multi-agent and produces complex interactions between observations, we explore the differences in performance between the discriminatively trained CRF and the generative HMM. Additionally, we examine the effect of incorporating features which violate independence assumptions between observations; such features are typically necessary for high classification accuracy. We find that the discriminatively trained CRF performs as well as or better than an HMM even when the model features do not violate the independence assumptions of the HMM. In cases where features depend on observations from many time steps, we confirm that CRFs are robust against any degradation in performance.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Machine Learning*

General Terms

Algorithms, Design, Performance

Keywords

Conditional Random Fields (CRFs), Activity Recognition, Hidden Markov Models (HMMs)

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

1. INTRODUCTION

In order to act effectively in the presence of others, an agent requires information about the other actors in its environment. Often, these other agents are either neutral, e.g. shoppers oblivious to a security camera, or hostile, e.g. an opposing team in robot soccer [3], towards the agent. In such cases, when an agent cannot rely on others to communicate their goals and behaviors, it must rely on activity recognition to form beliefs from its observations over time. It follows then that activity recognition is a temporal classification problem; an agent must generate a sequence of labels, identifying the roles or behaviors of the other agents, given a sequence of observations.

In this paper, we compare two models for temporal classification: hidden Markov models (HMMs) [9] and conditional random fields (CRFs) [5]. Hidden Markov models have a long history of use in activity recognition, e.g. [2, 11, 14], while conditional random fields, a more recent model, are just now expanding outside of their original natural language processing context into a range of areas such as image processing [4], gesture recognition [13], motion tracking [10], and activity recognition [6]. Studying CRFs and their application to this last domain is the focus of this paper.

Achieving high classification accuracy in complex tasks, such as activity recognition, often requires the use of domain knowledge to construct sophisticated features of the input observations. Such features typically incorporate information from more than a single time step. Features that span time steps violate the independence assumptions of the HMM, but not those of the CRF. We investigate whether or not this affects classification accuracy in either model. In addition to considering features over multiple time steps, we also examine features that link state transitions in the model directly to observations. Such features are difficult to represent in an HMM due to the way it factorizes probabilities. We investigate the utility of such features to see whether this difference in representational power can affect classification accuracy in practice.

In addition to considering features which violate the independence assumptions of an HMM or features that are difficult to represent in an HMM, we also consider the difference between discriminative and generative models for activity recognition. Specifically, we investigate the case where a CRF and an HMM are constructed using identical features. Results in [8], which examines discriminative-generative pairs of models using logistic regression and naive Bayes classifiers, suggest that a CRF should provide higher classification accuracy than an HMM, even when both mod-

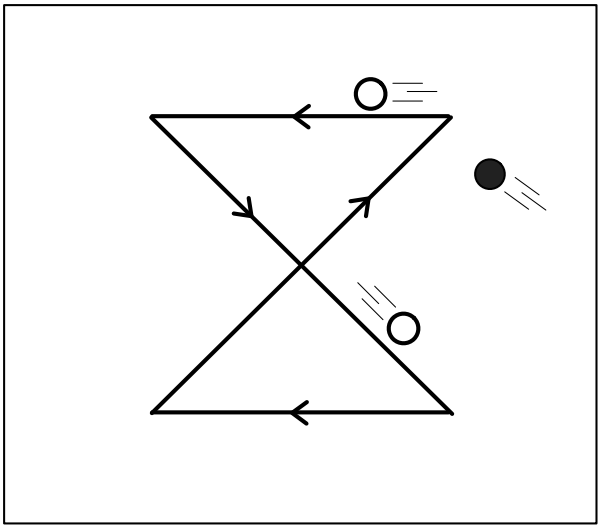

Figure 1: A scale representation of the robot tag domain. A single seeker robot (black circle) tries to *tag* one of the other two players by approaching to within 4 cm. In a simplified variant of the domain, the non-seeker robots follow an hourglass pattern. In an unconstrained variant, the non-seeker robots choose random target points in the playing area and navigate to those points.

els contain identical features. This is because CRFs and HMMs can be viewed as a discriminative-generative pair of models; later, we will provide the construction for converting an HMM into a CRF. And, as [8] demonstrated, discriminative models tend to have lower asymptotic error rates than their generative counterparts.

In the next section, we introduce the simulated robot tag domain that we used in our experiments. We motivate the design and properties of the tag domain in the context of activity recognition as a general problem. Following our domain description, we present an overview of conditional random fields, again in the context of activity recognition, before presenting experimental results..

2. THE ROBOT TAG DOMAIN

We created a simulation domain inspired by the children's game of Tag. In this domain, three robots move about on a playing field. Two of these robots take on passive roles where they navigate to a series of points on the field, oblivious to the actions of the other robots. The third robot takes an active role, which we call being the seeker. The seeker robot attempts to *tag* its closest neighbor and thereby transfer the seeker role to the tagged robot. *Tag*, in our domain, simply means approach within 4 cm of the target robot. To provide a sense of scale, the playing area is 3.5 by 4 meters in size and the robots are 20 cm in diameter. The domain is illustrated in figure 1.

We designed the robot tag domain to capture several real aspects of the activity recognition problem. The primary property of the domain is that we expect models built around the raw observations, i.e. the positions of the three robots, to perform poorly. On the other hand, we expect models built around carefully crafted feature functions, which take the raw observations as inputs, to perform well. The need for sophisticated features that distill meaning from the stream of raw sensor readings is a general requirement and applies to most activity recognition problems.

The robot tag domain, again like many activity recognition problems, has non-Markovian state transition dynamics. The role of seeker does not pass between the robots according to a multinomial distribution that is sampled at each time step. Instead, a single robot will remain the seeker for a period of time until it tags another robot. That is, the system state cannot transition until specific constraints are met. The constraints vary widely across activity recognition tasks, but often activities do have termination criteria. Models can exploit the ability to recognize these termination conditions in order to improve classification accuracy.

The final property of activity recognition that entered into the domain design is simply the fact that some activities are easier to recognize than others. As we will describe, some portions of the seeker's behavior are much easier to capture with a model than others. Specifically, when a robot is first tagged, it pauses in place for a moment before it begins to chase the other robots. It is trivial to recognize this initial paused state. However, once the seeker begins to pursue a target, it becomes much harder to identify; when the seeker is moving, the observations reveal three robots all executing "Navigate-to-point" behaviors. It just happens that the seeker chooses the current position of its target at each time step rather than a fixed point on the field.

In addition to creating easy and difficult portions of the seeker's behavior to identify, we also created two variants of the tag domain, one simple and one more complex, by giving the non-seeker robots two different strategies to use when selecting target points on the field. In the first variant, which we call the *hourglass* variant, the robots choose points so that they trace out an hourglass pattern in the arena (see figure 1). Each robot chooses the next point in the pattern, navigates to that point, and then selects a new point as its target. The only variation is when the seeker robot is located in the target quadrant of the playing field. In these cases, the robot chooses to move vertically rather than following the pattern in the correct order. However, once a target point has been picked, the robot navigates to that point regardless of the seeker's position.

With the second point selection strategy, which we call the *unconstrained* variant, the non-seeker robots choose random points anywhere on the playing field. Although we say this variant is unconstrained, target points that fall within 1 meter of the seeker will be rejected and a new random target picked until a suitable point is found. However, since the seeker moves, over time, any point in the player area may be chosen as a target. Once the target is chosen, the robots navigate to that position regardless of seeker position and only choose a new target once they have reached their destination.

As previously stated, when a robot is tagged (when the seeker approaches within 4 cm of it) and assumes the seeker role, it pauses in place for 5 seconds. This pause allows the previous seeker to move away rather than immediately being tagged in return.

3. CONDITIONAL RANDOM FIELDS

Conditional random fields are undirected graphical mod-

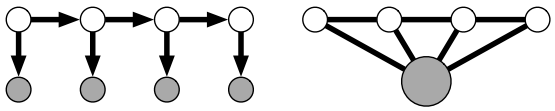

Figure 2: Left: A graphical model view of a hidden Markov model. The shaded nodes correspond to the observations $x_1, x_2, ..., x_T$. The clear nodes correspond to the label variables $y_1, y_2, ..., y_T$. Right: A graphical model view of a conditional random field. The single, large shaded node corresponds to the *entire* observation sequence X. As in the HMM, the clear nodes correspond to the label variables.

els that compactly represent the conditional probability of a particular label sequence, Y, given a sequence of observations, X; succinctly, CRFs model $P(Y|X)$. Modeling the conditional probability of the label sequence rather than the joint probability of both the labels and observations $P(X, Y)$, as done by hidden Markov models, allows CRFs to incorporate complex features of the observation sequence X without violating the independence assumptions of the model. The graphical model representations of an HMM and a CRF makes this difference explicit.

Figure 2 shows a hidden Markov model and a conditional random field as graphical models. The HMM models the joint probability $P(X, Y)$. To make inference in this joint model tractable, the HMM treats the observations as conditionally independent given the state labels. The conditional random field represents a conditional distribution of a Markov random field. The Markov random field represents the joint distribution over both X and Y, but the CRF conditions on X and therefore does not require any independence assumptions between the observation variables in X for tractable inference. When computing the probability of a single label y_t, the CRF can draw on the value of any observation from the entire sequence X without violating the assumptions of the model.

While it does eliminate the independence assumptions between the observations, the conditional random field maintains the same first order Markov assumptions as a hidden Markov model makes over labels. That is, conditioned on X, past labels are independent from future labels given the present label. As a result, inference in a CRF with such a linear chain of labels can be performed with the same time complexity as HMM inference, although training a CRF does require significantly more computation than training an HMM.

The graphical model representation of HMMs and CRFs clearly illustrates the assumptions made in each model to enable tractable inference. This graphical view also reveals the structure of the computation that must take place to compute $P(X, Y)$ or $P(Y|X)$ in an HMM or CRF respectively. In the case of an HMM, the joint probability of the sequence factorizes into pairs of terms, one term corresponding to pairs of labels and a second term for each observation with its parent label. In other words:

$$P(X, Y) = \prod_{t=1}^{T} P(y_t|y_{t-1}) \cdot P(x_t|y_t)$$

This equation follows from the familiar rules of Bayesian networks where distributions are encoded as products of conditional distributions; $P(V) = \prod_{v \in V} P(v|parents(v))$. Undirected graphical models provide no such notion of parents and children. To compute $P(Y|X)$ in a CRF, we talk in terms of cliques[1] and clique potentials rather than the product of conditional distributions as in a directed model.

The probability of a variable configuration in an undirected graph is proportional to the product of a series of non-negative potential functions, with one potential function for each clique of the graph. Intuitively, each potential function computes a value analogous to the probability that the variables in its corresponding clique clique take on a given configuration. The Hammersley-Clifford Theorem proves that such a composition of clique potentials produces a distribution that obeys the conditional independence assumptions encoded by the graph structure [1]. In other words, the probability of a variable configuration in an undirected graph is:

$$P(V) = \frac{1}{Z} \prod_{c \in cliques(V)} \psi(c)$$

where Z is a normalization term which guarantees that the distribution sums to one. Z can be computed exactly by summing over all possible configurations of V.

Returning to the specific case of conditional random fields, the cliques in a CRF consist of an edge between y_{t-1} and y_t as well as the edges from those two labels to the set of observations X. As a result, CRFs represent the conditional probability as:

$$P(Y|X) = \frac{1}{Z} \prod_{t=1}^{T} \psi(t, y_{t-1}, y_t, X)$$

This factorization follows directly from the properties of undirected graphical models and the Hammersley-Clifford Theorem. The definition of a conditional random field also specifies a specific form for the clique potentials ψ, namely that each potential takes the form $\psi(t, y_{t-1}, y_t, X) = exp(w \cdot f(t, y_{t-1}, y_t, X))$, yielding the following functional form:

$$P(Y|X) = \frac{1}{Z} \prod_{t=1}^{T} exp(w \cdot f(t, y_{t-1}, y_t, X)$$

$$Z = \sum_{Y} \prod_{t=1}^{T} exp(w \cdot f(t, y_{t-1}, y_t, X)$$

where w is a set of weights. These weights are the parameters we fit when learning with the model. The weights are multiplied by a vector of computed features $f(t, y_{t-1}, y_t, X)$, where the features are fixed functions designed to capture useful domain information.

It is worth noting that while the normalization constant Z involves summing over exponentially many sequences, for the case of linear chain CRFs, which make a first order Markov assumption between labels, this quantity can be computed efficiently via dynamic programming. This computation is more fully described in [15].

3.1 Training CRFs

Conditional random fields are commonly trained by maximizing the conditional likelihood of a labeled training set to estimate the weight vector w. As is usually the case with

[1]A clique is a fully connected subgraph.

maximum likelihood training, it is more convenient to maximize the log likelihood, which, along with its gradient, is:

$$\ell(Y|X) = \sum_{t=1}^{T} w \cdot f(t, y_{t-1}, y_t, X) - log(Z)$$

$$\frac{d\ell}{dw_i} = \sum_{t=1}^{T} f_i(t, y_{t-1}, y_t, X) - \sum_{Y} P(Y|X) \cdot f_i(t, y_{t-1}, y_t, X)$$

The functional form of the gradient supplies intuition about the model. The gradient will be equal to zero at the maximum likelihood solution. In order for this to be true, the empirical sum of each feature - the expected value of the feature according to the training set - must equal the expected value of that same feature under the model. Also of note, the gradient computation involves an exponential sum over all possible sequences; again, there is an efficient dynamic programming algorithm that computes this sum when a first order Markov assumption on the labels is made.

As for training, given a function and its gradient, training the model becomes a matter of numerical optimization. In the case of CRFs, the objective function is convex and first order methods, such as gradient ascent are directly applicable, although in practice more efficient algorithms such as conjugate gradient offer better performance. In addition to first order methods, an approximate second order method, Limited Memory BFGS [7], has also been used successfully [16, 12].

3.2 From HMM to CRF

In this section, we describe how to translate a hidden Markov model into the feature functions and weights of a conditional random field. This transformation illustrates that CRFs and HMMs form a discriminative-generative pair and also provides intuition about how feature functions can encode domain information such as state transition probabilities using the natural parameterizations of the exponential family of models.

Our goal is to convert:

$$\prod_{t=1}^{T} P(y_{t-1}|y_t) \cdot P(x_t|y_t) \Longrightarrow \prod_{t=1}^{T} exp(w \cdot f(t, y_{t-1}, y_t, X)$$

We will then normalize the latter of these expressions to form a proper conditional distribution that sums to one. This conversion is made easier by first noticing that both models involve a product over t and second that the potential function in the CRF factors into separate terms for each feature as follows: $\prod_i exp(w_i \cdot f_i) = exp(w_1 \cdot f_1) \cdot ... \cdot exp(w_T \cdot f_T)$ Because the clique potential factors in this fashion, we can separately represent individual terms in the HMM with single features or small groups of features in the CRF.

The first terms that we consider are the multinomial distributions for the transition model $P(y_t|y_{t-1})$ and the observation model $P(x_t|y_t)$ for discrete observations. To incorporate these distributions into our exponential model, we introduce features, one for each possible transition and one for each pairing of x_t and y_t that take the following form:

$$f_{i,j}(t, y_{t-1}, y_t, X) = I(y_{t-1} = i) \cdot I(y_t = j)$$
$$f_{s,v}(t, y_{t-1}, y_t, X) = I(y_t = s) \cdot I(x_t = v)$$

I represents the indicator function and takes the value 1 if its argument evaluates to true and 0 otherwise. Each feature has an associated weight which is simply the logarithm of the probability from the multinomial distribution, i.e.:

$$exp(w_{i,j} \cdot f_{i,j}) = P(y_t = j|y_{t-1} = i)$$
$$w_{i,j} \cdot f_{i,j} = log(P(y_t = j|y_{t-1} = i))$$
$$w_{i,j} = log(P(y_t = j|y_{t-1} = i))$$

Continuous features may be incorporated in a similar fashion. Assuming that observations are modeled as univariate Gaussian distributions, features and weights corresponding the the sufficient statistics for a Gaussian must be added to the model. The weight calculation is analogous to the calculation for multinomial distributions above:

$$exp(\vec{w} \cdot \vec{f}) = \frac{1}{\sqrt{\pi\sigma^2}} \cdot exp(-\frac{(x-\mu)^2}{2 \cdot \sigma^2})$$

$$exp(\vec{w} \cdot \vec{f}) = \frac{1}{\sqrt{\pi\sigma^2}} \cdot exp(\frac{-x^2}{2 \cdot \sigma^2}) \cdot exp(\frac{x \cdot \mu}{\sigma}) \cdot exp(\frac{-\mu^2}{2 \cdot \sigma^2})$$

$$\vec{w} \cdot \vec{f} = \frac{-1}{2 \cdot \sigma^2} \cdot x^2 + \frac{\mu}{\sigma} \cdot x + \left(\frac{-\mu^2}{2 \cdot \sigma^2} - log(\sqrt{\pi\sigma^2})\right) \cdot 1$$

We find ourselves with the sufficient statistics of a Gaussian and a constant term that will be absorbed into the normalization constant. Correspondingly, we add the following features to the model:

$$f_{0,j} = I(y_t = j) \cdot 1$$
$$f_{1,j} = I(y_t = j) \cdot x_t$$
$$f_{2,j} = I(y_t = j) \cdot (x_t)^2$$

3.3 A Note about Normalization

Training a CRF amounts to enforcing the moment conditions that the empirical and expected sufficient statistics agree. This can be seen from the gradient of $\ell(Y|X)$:

$$\frac{d\ell}{dw_i} = \sum_{t=1}^{T} f_i(t, y_{t-1}, y_t, X) - \sum_{Y} P(Y|X) \cdot f_i(t, y_{t-1}, y_t, X)$$

which will, of course, be zero at the maximum. At this point, the empirical sum of each feature, the first term, is equal to the expected sum of that feature under the model. With binary features that occur with approximately equal frequency in the training data, each component of the gradient will make a similar contribution to the over all magnitude of the gradient. In other words, the search direction in the optimization algorithm will not be dominated by a single feature.

When features take on continuous values, it is common for their empirical sums in the training set to vary significantly. This imbalance can cause optimization algorithms, such as conjugate gradient, to converge slowly or not at all. In such cases, it is important to normalize the features to be zero mean and have a variance of one. Intuitively, this normalization puts all of the features on equal footing during the optimization so that features with large values do not create a poorly scaled objective function.

4. EXPERIMENTS

In this section, we first present the notation we use to describe the feature functions that we use with the CRF and HMM and then present experimental results from the robot tag domain.

4.1 Notation

In the robot tag domain, we wish to identify which robot is the seeker at each time step of the simulation. We will refer to the ID of the seeker at time t as the label y_t and define the set of labels for the entire sequence $Y = \{y_1, y_2, ..., y_T\}$. Similarly, we define the input observations $X = \{x_1, x_2, ..., x_T\}$ where x_t is a vector of the observations from time step t. In the tag domain, the observation vector contains the two dimensional position of each robot in the environment; in other words, the position of the robot in the Cartesian plane, but not its orientation.

We use the following notation when describing the features used by the models: $\vec{p}_{r,t}$ refers to the position of robot r in the two dimensional environment at time t. Position values are available directly from the observation x_t at each time step. Positions are measured in millimeters and the origin $(0, 0)$ corresponds to the center of the environment. We also use $\vec{v}_{r,t} = (\vec{p}_{r,t} - \vec{p}_{r,t-1}) \cdot dt$ to refer to the estimated velocity of robot r at time t.

4.2 Results

We generated experimental data from both the hourglass and the unconstrained tag domains. This data was generated from a physics-based simulator for holonomic robots that included realistic acceleration and velocity constraints on the robots. We generated training and test sets that were each approximately 20 minutes in length. Since the simulator operates at a rate of 60 hz, this means that each sequence contained more than 70,000 observation vectors, each labeled with the identity of the seeker at that time step. The observation vectors themselves consisted of three two-dimensional positions, one for each robot. These positions form the basis of the features passed to the models for classification.

The goal of our experiments was to compare HMMs and CRFs using identical features in each model. Features in a conditional random field are functions that take the form $f(t, y_{t-1}, y_t, X)$. It is difficult to use these functions directly as input to an HMM due to the presence of y_{t-1} in the feature definitions. Because of this difficulty, we dropped the y_{t-1} terms of functions when constructing the HMM observation model and instead had the HMM model $P(g(t, X)|y_t)$, where $g(t, X)$ is the portion of the CRF's feature function that does not depend on either y_{t-1} or y_t. We used a Gaussian observation model in the HMM where each feature was treated independently from the other features.

In all experiments, the CRF included intercept features ($f = I(y_t = j)$) as well as transition features ($f = I(y_{t-1} = i) \cdot f(y_t = j)$) as a base model. We added more specialized features on top of these. Because the HMM's transition model and Gaussian observation model provide equivalent functionality, these features were not passed to the HMM as observations.

4.2.1 Raw Positions Only

In the initial set of experiments, we trained the HMM and CRF on the raw positions of each robot; in other words, the models were given x_t as input at each time step. The goal of these experiments was to verify that some sort of computed features are indeed required for reasonable classification accuracy.

To complement the HMM's Gaussian observation model, the CRF included features for $x_t[k]$ and $(x_t[k])^2$ for each of the k features of x_t. Taken with the intercept features in the CRF, this provided the sufficient statistics for a Gaussian observation model. This choice of features makes the CRF a discriminatively trained HMM. Concretely, we added the following features to the CRF:

$$f_{j,k} = I(y_t = j) \cdot x_t[k]$$
$$f_{j,k} = I(y_t = j) \cdot x_t[k]^2$$

The results in table 1 show two things. First, the raw observations by themselves do not provide enough information for either model to perform well. In the case of the hourglass domain, the problem is simple enough that the CRF performs far better than the baseline performance achievable by random guessing (approximately 33%). For this simple domain, at least, there is an advantage to training the model discriminatively as the CRF does better than the HMM, although this advantage seems to disappear with the more complex, unconstrained data set where both models perform poorly.

4.2.2 Velocities

The poor performance resulting from training models on the raw locations of the robots suggests that additional features, which reflect domain knowledge about the classification task, are required for high classification accuracy. The most obvious such features are the velocities of the robots, which can be estimated by subtracting adjacent pairs of x_t observations. As domain experts, we know that velocity is an informative feature because the seeker robot pauses in place for five seconds when it is first tagged. As such, we introduced the following features into the CRF:

$$f_{r,j} = I(y_t = j) \cdot \|\vec{v}_{r,t}\|$$
$$f_{r,j} = I(y_t = j) \cdot \|\vec{v}_{r,t}\|^2$$

As table 1 shows, simply providing the models with robot velocities rather than raw positions causes a dramatic improvement in classification accuracy.

4.2.3 Velocity Thresholds

Incorporating velocity information into the models greatly improved accuracy. One obvious reason for this is the five second pause the seeker makes when first tagged. Rather than feeding raw velocities into the models, we constructed features to test if the velocity is below a certain threshold for a single time step or for a series of consecutive time steps. Such features more clearly capture the notion of a stopped or stopping robot. As a natural complement to testing if a robot's velocity is below a threshold, it makes intuitive sense to test if the robot's velocity is above a threshold as well. This second form of feature will be correlated with non-seeker robots as well as the seeker when it has finished pausing after first being tagged. The particular features used took the form:

$$f_{r,j}(k, w) = I(y_t = j) \cdot \prod_{i=t-w+1}^{t} I(\|\vec{v}_{r,i}\| \leq k)$$

$$f_{r,j}(k, w) = I(y_t = j) \cdot \prod_{i=t-w+1}^{t} I(\|\vec{v}_{r,i}\| > k)$$

In our experiments, we chose the velocity threshold k to be 20% of the robots' maximum velocity.

	Hourglass			Unconstrained		
Features	HMM Acc.	CRF Acc.	$\ell(Y\|X)$	HMM Acc.	CRF Acc.	$\ell(Y\|X)$
Positions	33.1	53.6	-959.7	37.1	37.8	-1354.5
Velocities	68.4	89.4	-717.1	55.7	70.4	-1206.5
Velocity Thresholds						
W = $\frac{1}{60}$th sec.	62.5	71.2	-818.0	46.8	58.6	-1148.6
W = 0.1 sec.	63.0	73.9	-784.3	46.0	62.4	-1099.2
W = 0.5 sec.	63.6	80.6	-708.8	68.9	71.9	-983.1
W = 1.0 sec.	60.2	83.1	-721.8	67.8	75.3	-980.9
W = 1.5 sec.	56.9	85.5	-731.7	68.8	77.8	-1004.7
W = 2.0 sec.	53.7	87.1	-751.1	72.1	77.3	-1036.3
Chasing	75.8	95.4	-622.3	65.5	80.4	-1058.3
Distance (U)	46.6	49.5	-779.7	43.5	42.3	-604.4
Distance (N)	46.6	49.9	-200.6	43.5	58.0	-223.4
Distance & Chasing (U)	75.6	99.3	-90.6	65.8	93.9	-181.8
Distance & Chasing (N)	75.6	99.3	-115.3	65.8	97.6	-112.2
Many Features	72.4	98.1	-172.2	63.4	98.5	-178.9
Redundant Features	72.4	95.7	-509.3	52.7	93.8	-6432.3

Table 1: CRF and HMM accuracy for identifying the seeker in the simplified hourglass and unconstrained tag domains. (U) and (N) indicate that the binary distance threshold features were either unnormalized or normalized in the corresponding trials.

In addition to testing the effectiveness of incorporating specific domain knowledge into the models, these velocity thresholds allow us to perform another test as well; hidden Markov models assume that observations are independent given their labels. Creating features from overlapping portions of the observation sequence X, invalidates the assumptions of the model. An HMM with such overlapping features is no longer a proper generative model and the likelihood function for the HMM is no longer correct due to over counting of the evidence. By testing the HMM and CRF with differently sized windows, we can test how much, if at all, violating the independence assumptions of the HMM harms classification accuracy.

The results for the CRF in table 1 are unambiguous. The conditional random field becomes more accurate as the size of the window is increased. Incorporating information from many different time steps into feature functions does not violate any of the independence assumptions made by the CRF.

The HMM results are less clear cut. With the hourglass data set, the accuracy of the HMM decreases as the window size is increased, perhaps due to the increasingly severe violation of the independence assumptions of that model. In contrast, HMM performance increases as the window size is increased with the more complex, unconstrained data set. It is not clear why performance suffers with larger window sizes on the simple dataset but then improves on the more complex data.

4.2.4 Chasing Features

The velocity threshold features showed that using domain knowledge to create features can improve model accuracy. Aside from the seeker pausing in place after being tagged, the other defining characteristic of the seeker role is that the seeker chases other robots. To capture this property of the tag domain, we created a "chasing" feature to capture whether each robot was moving towards one of the other robots. This feature computes the cosine of the angle between a robot's velocity and the position of its nearest teammate, where r_1 is the potential seeker and r_2 is its closest neighbor:

$$f_{r_1,r_2,j} = I(y_t = j) \cdot \frac{1}{\|\vec{p}_{r_2,t} - \vec{p}_{r_1,t}\|} \cdot (\vec{p}_{r_2,t} - \vec{p}_{r_1,t}) \cdot \frac{1}{\|\vec{v}_{r_1,t}\|} \cdot \vec{v}_{r_1,t}$$

As with the velocity threshold functions, incorporating domain knowledge in the form of feature functions improves accuracy. Table 1 shows that classification accuracy in the hourglass domain breaks 95% for the first time while accuracy in the unconstrained domain breaks 80% for the first time.

Up until this point, we have been creating features that can be incorporated into both conditional random fields and hidden Markov models. Some of these features, such as the velocity thresholds, broke independence assumptions made by the HMM, but it was clear how to add them to the model in spite of this. Now we turn our attention to a class of features that can be represented in a CRF, but not in an HMM because the HMM splits y_{t-1} and x_t into separate factors.

4.2.5 Distance Thresholds

Activity recognition in the game of tag is a matter of detecting transitions. In one time step, robot 1 is the seeker. During the next time step, robot 1 approaches close enough to tag a different robot and that new robot becomes the seeker. We are interested in detecting the points where the seeker role is passed between players.

By looking at the distances between pairs of robots over a series of adjacent time steps, we can create features to detect when a hand off of the seeker role may have occurred. At first, this simply appears to be a matter of tracking whenever two robots are in close proximity and having a feature act as a flag at these points, with one flag for each pair of robots that could be involved in the transaction.

However, there are many time steps when robots will be close to each other and no hand off will take place; when a

robot is first tagged, it halts in place. It takes the former seeker time to deaccelerate and move away from the newly created seeker. During this time, both robots will be within the threshold distance, although no hand off is possible. In other words, it is only possible for a hand off to occur during the first frame that two robots are within the threshold distance.

A second complication arises because the execution order of the robot behaviors is not fixed in the simulator. In some situations, the non-seeker will close the gap between itself and the seeker and in others the seeker will close the gap with its action. Depending on which order these events take place, the hand off of the seeker role can occur in either the first time step when the robots are below the threshold distance or during the second time step. Accounting for both possible times when a hand off could have occurred, our distance threshold feature takes the form:

$$f_{r_1,r_2,i,j} = I(y_{t-1} = i) \cdot I(y_t = j) \cdot I(dist_t(r_1, r_2) \leq k \vee dist_{t-1}(r_1, r_2) \leq k) \cdot I(dist_{t-2}(r_1, r_2) > k)$$

We also include an inverted version of the above features to indicate situations where it would be impossible for a transition to have occurred:

$$f_{r_1,r_2,i,j} = I(y_{t-1} = i) \cdot I(y_t = j) \cdot I(dist_t(r_1, r_2) > k \wedge dist_{t-1}(r_1, r_2) > k)$$

The model is able to learn negative weights to make label sequences Y in which such transitions occur less likely than sequences that lack forbidden state transitions.

The results in table 1 are somewhat surprising. These distance threshold based state transitions exactly capture the underlying dynamics of the process that generated the data, yet using only these features results in lackluster performance. The first set of results, labeled "Distance (U)", show what happens when the feature values are not normalized to have mean zero and a variance of one. After all, such normalization is rarely necessary with binary features.

However, consider how often two robots will first approach close enough together for a tag event to take place. These are the only situations where the distance threshold features will take on a non-zero value. This situation happens approximately 200 times in the twenty minute training set. Now consider how often the features corresponding to self-transitions take on a non-zero value. One of these features fires for virtually each of the roughly $70,000$ time steps in the training set. The expected value of the regular transition features is much larger than the expected value of the distance threshold features. Any difference in the empirical value of the regular transitions and their expected value under the model will dominate the function gradient during optimization.

Rather than eliminating the regular transition features, which would leave the optimizer stranded in a plateau and unable to make any progress, we normalize the distance thresholded transitions. Normalizing the distance threshold features produces no appreciable change in the accuracy in the hourglass domain and the accuracy in the unconstrained domain improves slightly. However, there is a major difference in the log likelihood of the test set for both domains, so clearly the model achieves a better fit on the test set even if this improvement is not enough to greatly change the accuracy of the models.

4.2.6 Distance Thresholds with Chasing Features

A natural question that arises is whether or not the distance threshold features are capable of increasing the accuracy of the CRF. In theory, they perfectly capture the characteristics of the domain, but in practice training the model with only these features is difficult. As a test, we combined the distance threshold features with the chasing features to see if the combination of the features together would out perform either set of features on its own.

As the results in table 1 illustrate, there is indeed a synergistic effect between the two feature sets. One set of features, the chasing features, captures the behavior of the seeker when it is moving. The other set, the distance threshold features, captures when the seeker role transitions from one robot to another. The resulting models exhibit higher log likelihoods on the test set as well as, when the distance thresholds are normalized, better accuracies than any other features tried thus far.

4.2.7 Many Features

In the final set of experiments, we examined how the models perform when all of the previously discussed features were used at the same time. In the case of the windowed velocity thresholds, only a single threshold corresponding to a window size of 1 was used to avoid deliberately breaking the independence assumptions of the HMM.

As a second experiment, we included all of these "helpful" features, as well as redundant features to test how vulnerable the models are to over fitting. The extra features we added were the raw position observations, the change in distance between each robot and its closest neighbor, each robots v_x and v_y velocity components, and a series of dot products along each robots trajectory to measure how linearly the robot was moving as non-seeker robots tend to travel in straight lines. While all of these additional features potentially include information about the domain, this information is either provided by the existing features or somewhat dubious in quality.

The results in table 1 show an initial increase in accuracy for the CRF when the first collection of features are added to the model and then a decrease with the addition of redundant features on the unconstrained data set. On the hourglass dataset, even the addition of the first set of features lowers the final accuracy of the model, probably because of over fitting. Over fitting reduces accuracy immediately with the simple data set but only with the addition of redundant features for the more complex data. The degree of over fitting is especially evident from the large decrease in the log likelihood of the test set for the unconstrained data set with redundant features.

In our experiments, we did not apply any regularization to the models. Regularization would probably help avoid the obvious over fitting that took place, particularly in the case of the unconstrained data set moving from the full compliment of helpful features to the larger set that included irrelevant features. It is trivial to apply L2 regularization during CRF training. Regularization is discussed in more detail in [15].

5. CONCLUSIONS

We have presented a comparison of conditional random fields and hidden Markov models on a simple activity recognition task. We have demonstrated how CRFs are able to easily incorporate a wide variety of computed features which allow domain knowledge to be added to the models. Furthermore, we have shown that due to the independence assumptions inherent in HMMs, such computed features are not nearly as effective for improving classification accuracy.

In all of our experiments, conditional random fields scored higher or on par with hidden Markov models in terms of classification accuracy. This was the case even in those situations where the CRF was effectively a discriminatively trained HMM. These results make sense when placed in the context of [8], in which Ng and Jordan compare the effectiveness of generative versus discriminative classifiers. In their case, they consider Naive Bayes versus Logistic Regression because those two classifiers form what they term a discriminative-generative pair. The same thing can be said for HMMs and CRFs. And, as we would expect, the discriminative model has a lower asymptotic error rate.

We have also demonstrated how conditional random fields are able to incorporate features that are difficult to represent in a hidden Markov model, namely features that link state transitions directly to observations, e.g. the distance threshold features, which exactly captured the dynamics of the process.

Sophisticated computed features which link observations to transitions or incorporate observations from several time steps are often required for good classification accuracy in activity recognition tasks. It then follows that conditional random fields are suitable models for activity recognition. As a model, CRFs do not make independence assumptions between the observations, they can link a particular observation (or any function computed on the observation sequence) to state transitions, and, as discriminatively trained models, they will often have lower error rates than the corresponding generative model.

Acknowledgments

This research was sponsored in part by the Department of the Interior, National Business Center under contract no. NBCHD030010 and SRI International under subcontract no. 03-000211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

6. REFERENCES

[1] J. Hammersley and P. Clifford. Markov fields on finite graphs and lattices, 1971.

[2] K. Han and M. Veloso. Automated robot behavior recognition applied to robotic soccer. In J. Hollerbach and D. Koditschek, editors, *Robotics Research: the Ninth International Symposium*, pages 199–204. Springer-Verlag, London, 2000.

[3] H. Kitano, editor. *RoboCup-97: Robot Soccer World Cup I*. Springer-Verlag, London, UK, 1998.

[4] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 1150, Washington, DC, USA, 2003. IEEE Computer Society.

[5] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, pages 282–289. Morgan Kaufmann, San Francisco, CA, 2001.

[6] L. Liao, D. Fox, and H. Kautz. Hierarchical Conditional Random Fields for GPS-based activity recognition. In *Proc. of the International Symposium of Robotis Research (ISRR 2005)*. Springer Verlag, 2005.

[7] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Math. Program.*, 45(3):503–528, 1989.

[8] A. Y. Ng and M. I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 841–848, Cambridge, MA, 2002. MIT Press.

[9] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[10] D. A. Ross, S. Osindero, and R. S. Zemel. Combining discriminative features to infer complex trajectories. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 761–768, New York, NY, USA, 2006. ACM Press.

[11] P. Rybski and M. Veloso. Using sparse visual data to model human activities in meetings. In *Modeling Other Agents from Observations Workshop (MOO 2004)*, 2004.

[12] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 134–141, Morristown, NJ, USA, 2003. Association for Computational Linguistics.

[13] C. Sminchisescu, A. Kanaujia, Z. Li, and D. Metaxas. Conditional models for contextual human motion recognition. In *IEEE International Conference on Computer Vision (ICCV 2005)*, pages 1808–1815, 2005.

[14] G. Sukthankar and K. Sycara. Robust recognition of physical team behaviors using spatio-temporal models. In *Proceedings of Fifth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, May 2006.

[15] C. Sutton and A. McCallum. An introduction to conditional random fields for relational learning. In L. Getoor and B. Taskar, editors, *Introduction to Statistical Relational Learning*. MIT Press, 2006. To appear.

[16] H. Wallach. Efficient training of conditional random fields. Master's thesis, University of Edinburgh, 2002.