Feature Selection in Conditional Random Fields for Activity Recognition

Douglas L. Vail
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA
dvail2@cs.cmu.edu

John D. Lafferty
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA
lafferty@cs.cmu.edu

Manuela M. Veloso
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA
veloso@cs.cmu.edu

***Abstract*— Temporal classification, such as activity recognition, is a key component for creating intelligent robot systems. In the case of robots, classification algorithms must robustly incorporate complex, non-independent features extracted from streams of sensor data. Conditional random fields are discriminatively trained temporal models that can easily incorporate such features. However, robots have few computational resources to spare for computing a large number of features from high bandwidth sensor data, which creates opportunities for *feature selection*. Creating models that contain only the most relevant features reduces the computational burden of temporal classification. In this paper, we show that ℓ_1 regularization is an effective technique for feature selection in conditional random fields. We present results from a multi-robot tag domain with data from both real and simulated robots that compare the classification accuracy of models trained with ℓ_1 regularization, which simultaneously smoothes the model and selects features; ℓ_2 regularization, which smoothes to avoid over-fitting, but performs no feature selection; and models trained with no smoothing.**

I. INTRODUCTION

Temporal classification, such as activity recognition, is a key component for creating intelligent systems that perceive and respond appropriately to their environments. In the case of robot systems, temporal classification is challenging for two reasons. First, robots must process sensor information and react in real time, which leaves little time for computation. The second challenge is that robot sensors provide vast amounts of data that must be sifted through in order to extract useful information; sensors, such as color cameras, accelerometers, joint encoders, and laser range finders produce an enormous amount of data. The individual pieces of data, e.g. the color of a single pixel, contain little information, but the data taken as a whole, e.g. an entire image, contains rich information about the environment.

Classifiers address the challenge of extracting information from raw sensor data through the use of *features*. They operate on functions of the sensor data, e.g. the output of a color blob tracker in a vision system, rather than the raw data itself. Of course, computing features from sensor data requires time and computational resources, which are limited on robot platforms. Therefore, to perform temporal classification in a robot system, we require two things: models capable of robustly incorporating many non-independent features of the sensor data and a means of performing *feature selection* so that only the most relevant features are incorporated into those models. In this paper, we demonstrate that conditional random fields [9], trained with ℓ_1 regularization, meet these requirements.

Conditional random fields (CRFs) are discriminatively trained graphical models for labeling sequential data. Their key property, in our context, is that they can directly incorporate many non-independent features of their observations, i.e. features of the sensor data, without violating independence assumptions in the model. This is because CRFs conditionally model the probability of a label sequence given the observations. In part due to this property, conditional random fields are increasingly being applied to tasks that are relevant to robots, such as image classification [8], gesture recognition [17], motion tracking [15], and activity recognition [11], [18]. This last domain, activity recognition, is where we consider *feature selection* in CRFs via ℓ_1 regularization using robots from the Small Size league of RoboCup [6].

II. ROBOT TAG DOMAIN

We consider the problem of feature selection in CRFs using a domain inspired by the children's game of Tag and implemented using the robots of the CMDragons'06 robot soccer team [2]. In this domain, three holonomic robots move on a playing field. Two of the robots take on passive roles where they navigate to a series of randomly chosen points on the field. The third robot takes an active role, which we call being the seeker. The seeker attempts to *tag* its closest neighbor and thereby transfer the seeker role to the tagged robot. In our domain, *tag* simply means approach to within a threshold distance of the target robot.

The two non-seeker robots are passive in the sense that once they choose a target in the playing area, they navigate to that target without actively avoiding the seeker. Once they reach the target, they sample a new target, drawn uniformly, but with rejection, from within the playing area. The non-seeker robots reject candidate target points that lie within 1 meter of the seeker, so there is bias away from the seeker when choosing a target point, but the robots do not avoid the seeker once they start moving to the target. To provide a sense of scale, the playing area is 3.5 by 4 meters in size, the robots are 20 cm in diameter, and the seeker must approach within 4 cm of another robot in order to tag it. When a robot is first tagged, it pauses in place for 5 seconds in order to avoid chasing and immediately re-tagging the previous seeker. This tag domain provides a

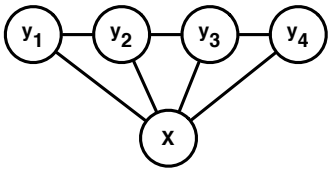

Fig. 1. The graphical structure of a conditional random field. The variables y_t represent the labels at each time step t. The entire sequence of observations $\{x_1, x_2, ..., x_t\}$ is represented by the single node X.

useful benchmark problem for testing classifiers for activity recognition in a real robot system.

The system as a whole operates at a frame rate of 60 hz. At the beginning of each frame, two overhead cameras capture images of the field. A color vision system [3] uses colored markers on the tops of the robots to extract their position and orientation from the camera images. An extended Kalman filter tracks the robots over time and this information is used for real time path planning and by the behaviors that control the individual robots. The tracked robot positions also serve as the input for activity recognition.

In our benchmark problem, the roles of each robot are, of course, known with certainty. However, e.g. in the case of robot soccer, activity recognition can provide useful information about opponent robots. For example, it would be useful for soccer behaviors to adapt their strategy when activity recognition detects that an opponent defender is marking a particular offensive player. The classification task is then to map from a series of observations that arrive at 60 hz to a series of labels that specify useful state information. In the robot tag domain, the label at each time step identifies the robot currently executing the seeker role and the observations are the locations of the three robots in the playing area.

III. CONDITIONAL RANDOM FIELDS

Conditional random fields are undirected graphical models that compactly represent the conditional probability of a label sequence, $Y = \{y_1, y_2, ..., y_T\}$, given a sequence of observations, $X = \{x_1, x_2, ..., x_T\}$. CRFs represent this conditional probability, $P(Y \mid X)$, as the product of potential functions that are computed over each clique in the graph. In turn, these potential functions are computed in terms of feature functions of the observations and adjacent pairs of labels. To make this concrete, consider the case of our robot tag activity recognizer.

The activity recognizer must identify which of the three robots is seeker sixty times per second given the positions of the three robots as input. Intuitively, the label at each time step, y_t, takes its value from the set $\{1, 2, 3\}$, corresponding to the statement: the robot with ID y_t was seeker at time t. The observation at each time step, x_t, is a six-tuple that specifies the Cartesian coordinates of the three robots in the playing area.

The graphical structure shown in figure 1 illustrates the independence assumptions made by a CRF. We see that adjacent pairs of labels, (y_{t-1}, y_t), are linked by undirected edges, which corresponds to a first-order Markov assumption over labels given the observations; past labels are conditionally independent of future labels given the present label and observations. As conditional models CRFs make no independence assumptions between the observations. This means that the model is free to use the entire observation sequence, X, when computing clique potentials over adjacent pairs of labels. This stands in contrast to directed models, such as hidden Markov models [14], which assume the observations from any single time step, x_t, are independent of all other observations given the label y_t.

A. Clique Potentials and Features

CRFs represent the conditional $P(Y \mid X)$ as the product of potential functions. These potential functions are computed over each clique of the graph and they are built from feature functions. In the graphical structure of a CRF, shown in figure 1, the cliques consist of adjacent pairs of labels and the entire observation sequence, so the clique potentials take the form: $\psi(t, y_{t-1}, y_t, X)$, where y_{t-1} and y_t represent label assignments at a pair of adjacent nodes, X represents the entire observation sequence, and t is an index into the sequence identifying where the edge (y_{t-1}, y_t) falls in the sequence. Because CRFs are log-linear models, the potential functions take the specific form: $\psi(t, y_{t-1}, y_t, X) = \exp(w^T f(t, y_{t-1}, y_t, X))$, where w represents a vector of weights and f is a vector of feature functions. The weight vector contains the parameters of the model and is estimated during training. The feature functions that make up f identify relevant configurations of (t, y_{t-1}, y_t, X). As stated above, the conditional probability of the label sequence is represented as the product of all of the clique potentials:

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

$$Z = \sum_{Y} \prod_{t=1}^{T} exp(w^T f(t, y_{t-1}, y_t, X)). \qquad (2)$$

where Z is a normalizing constant.

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

$$\ell(Y \mid X; w) = \sum_{t=1}^{T} w^T f(t, y_{t-1}, y_t, X) - log(Z) \qquad (3)$$

$$\frac{d\ell}{dw_i} = \sum_{t=1}^{T} f_i(t, y_{t-1}, y_t, X) - \sum_{Y} P(Y \mid X) f_i(t, y_{t-1}, y_t, X). \qquad (4)$$

This yields a convex objective function and its first derivative. Standard optimization techniques, such as conjugate

gradient and limited memory BFGS [12] are easily applied and, indeed, among the fastest known methods for training CRFs [16], [19].

Often, classification accuracy can be improved by adding a regularization or smoothing term to the objective function to reduce over-fitting during training. This is justified by the usual argument that models with fewer parameters or smaller parameters are more parsimonious. We examine two smoothing techniques, namely ℓ_1 and ℓ_2 regularization. We begin by describing the later.

C. ℓ_2 Regularization

A common penalty function used to regularize CRFs during training is the ℓ_2 norm. Rather than maximizing the conditional likelihood alone, penalty terms for each weight, proportional to w_i^2, are subtracted from the likelihood:

$$\max_w \ell(Y \mid X; w) - \lambda w^T w \tag{5}$$

The parameter λ controls the degree of smoothing that is applied to the model. High values of λ correspond a large amount of smoothing and a λ of zero results in no smoothing. The penalized objective function remains convex and differentiable and therefore training a CRF with an ℓ_2 penalty requires about the same computational effort as training a CRF without regularization. The only difficulty lies in choosing a good value for λ, which can be done, for example, by evaluating different settings of λ through cross-validation or by making use of a hold out set.

D. ℓ_1 Regularization

Another common penalty function for regularizing models is the ℓ_1 norm. Rather than penalizing the objective function by w_i^2, as in the ℓ_2 norm, the ℓ_1 norm applies penalties that are proportional to $\mid w_i \mid$, which, in the case of a CRF, results in the penalized objective function:

$$\max_w \ell(Y \mid X; w) - \lambda \sum_i \mid w_i \mid, \tag{6}$$

where λ is again a parameter that controls the degree of smoothing during training. While the penalized objective function 6 remains convex, adding the ℓ_1 penalty means that it is no longer differentiable.

Regularizing with an ℓ_1 penalty complicates training, but has the advantage of producing sparse models [5]; ℓ_1 regularization produces smoothed models where some of the weights are exactly equal to zero. This is equivalent to feature selection because features with zero weights can be eliminated from the model. For intuition, consider the weight $w_i = 0$. Under an ℓ_2 penalty, the partial derivative of the penalty is $2\lambda w_i$. To a first order approximation, the change in the penalty is zero for moving w_i away from zero. With an ℓ_1 penalty, the relevant partial is $\pm\lambda$, so the change in the penalty is proportional to λ. Under an ℓ_2 penalty, a small non-zero derivative in the *unpenalized* objective function will move w_i away from zero. In the ℓ_1 case, λ serves as a threshold and prevents w_i from becoming non-zero to buy small improvements in the unpenalized objective function.

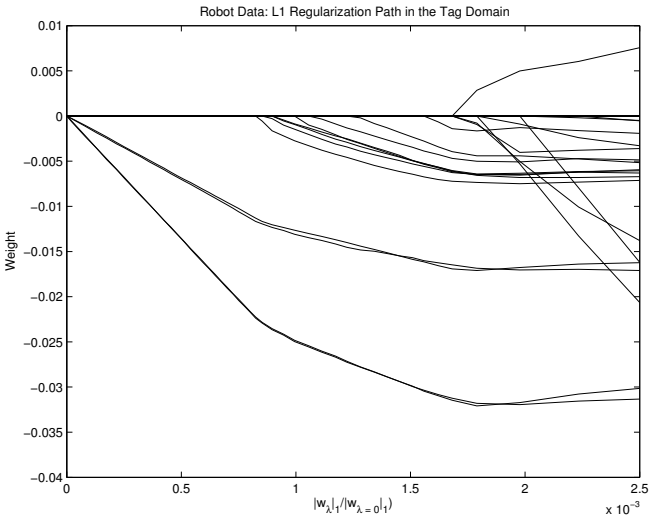

Fig. 2. A portion of the ℓ_1 regularization path for a CRF trained for activity recognition in the robot tag domain. Each line in the plot represents the weight associated with a feature in the CRF. For high values of the regularization parameter λ (left side), all features have weights of zero and the model contains no features. As λ is decreased, features enter the model by taking on non-zero weights. The horizontal axis corresponds to $|w_\lambda|_1/|w_{\lambda=0}|_1$, that is, the ℓ_1 norm of the weights for each value of λ normalized by the ℓ_1 norm of the weights that result when the model is trained without regularization.

Recent work in the machine learning community has shown that ℓ_1 regularization is viable in generalized linear models, the family of models that includes both CRFs and logistic regression. In [13], Park and Hastie consider the entire family of generalized linear models and introduce a predictor-corrector method that recovers the entire regularization path of a model, although potentially at a high computational cost. In [10], Lee et al. consider the case of logistic regression. They exploit the existence of efficient algorithms for performing ℓ_1 regularized linear regression for efficiently training in the logistic regression case. Koh et al. also consider logistic regression in [7], where they use an interior point method to approximate the regularization path of the model; they do not recover the exact positions where features enter and exit the model, but they show that their method can efficiently recover many points along the path, which is sufficient for feature selection in most settings.

The work of Koh et al. [7] is the most similar to our contribution. In our work, we consider conditional random fields rather than logistic regression. Both are log-linear models, but CRFs include the notion of time and require a dynamic programming step to compute the partition function. We investigated the interior point method of Koh et al., but we met with more success using projected conjugate gradient, as described in the next section, for training. Following Koh et al., we use a warm start technique to recover the regularization path of a CRF. We use the regularization path, along with a held out portion of the data set for feature selection in the CRF. A portion of such a regularization path is shown in figure 2.

E. Training with an ℓ_1 Penalty

When training a CRF with an ℓ_1 penalty, the objective function 6 has an unconstrained domain and it is not differentiable. Rather than maximizing 6 directly, we optimize a different objective function over a constrained domain. This new objective function 8, has a well defined first derivative. We generate the new objective function by reparameterizing the original function in terms of two vectors, w^+ and w^-, that are related to w from equation 6 by $w = w^+ - w^-$. We constraint the entries of w^+ and w^- to be non-negative, yielding the new optimization problem:

$$\max_{w^+,w^-} \ell(Y \mid X; w) - \lambda \sum_i w_i^+ - \lambda \sum_i w_i^-, \quad (7)$$

$$s.t.\ w_i^+ \in w^+ \geq 0, w_i^- \in w^- \geq 0 \quad (8)$$

We solve this constrained optimization problem using projected conjugate gradient. Briefly, algorithms such as conjugate gradient optimize through a series of minimizations along a line. Each minimization begins with an initial point x_0 and a search direction d. The objective function is evaluated and candidate points x, where $x = x_0 + \alpha d$, $\alpha \geq 0$, and a line search algorithm is used to choose the optimal value of α. The key idea of projected conjugate gradient is that candidate points x are projected back into the feasible region before being evaluated. In our case, that amounts to computing x_p, such that $x_p = max(0, x)$. This type of projected optimization algorithm has long been used to solve constrained optimization problems with simple inequality constraints [1] and is an efficient method for training CRFs with an ℓ_1 penalty for a specific value of λ.

Of course, to produce the regularization path, many values of λ must be considered. For the case of ℓ_1 regularized linear regression, an efficient algorithm to compute the full regularization path is known [4]. No such algorithm is known for the case of generalized linear models [13], however, Koh et al. [7] noted that an approximation to the regularization path, which does not produce the exact values of λ where features enter and exit the model, can be computed efficiently via a warm start technique. That is, the model is first trained with the largest value of λ and then the process is repeated for the next smaller value of λ, initializing the optimization with the weights from the previous step, which dramatically reduces the computational requirements at each step.

IV. EXPERIMENTS

To explore the effectiveness of ℓ_1 regularization for both smoothing and feature selection, we created a CRF to identify which robot was seeker in the robot tag domain. We trained this CRF with either no regularization, ℓ_1 regularization, or ℓ_2 regularization. The motivation for this last form of training is to compare smoothing techniques that do not simultaneously perform feature selection to ℓ_1 regularization, which does both simultaneously.

We generated datasets using both actual robots and a robot simulator to test how well models trained on simulated data generalized to data from real robots. In both cases, the training, hold out, and test sets each contained ten minutes of data, or since the system operates at 60 hz, 36,000 time steps. We trained the models with the same implementation of conjugate gradient and used the same termination criteria, except in the case of the ℓ_1 regularization, where the optimization algorithm was modified to project back into the feasible region of the search space.

The experiments with regularization required testing many different values of the regularization parameter λ. In the case of training with an ℓ_1 penalty, there is a clear notion of a regularization path starting with a λ set high enough that all features have zero weights and extending through lower values of λ until most or all weights in the model have non-zero weights. To generate such a path, we chose an initial value λ_0 large enough that no features were present in the model and then set $\lambda_{k+1} = .9\lambda_k$ for each succeeding trial. In the case of ℓ_2 regularization, there is not such a clear notion of a regularization path, so we reused the same λ values that were used in the ℓ_1 trials.

A. Features in the CRF

The CRF included the same features, described below, in all experiments. There were a total of 1174 features in the model, all of which were designed to be relevant to the classification task. In all feature descriptions, $I(condition)$ represents the identity function and evaluates to 1 if the condition evaluates to true and 0 otherwise. For example, $I(y_t = i)$ tests if the label at time t identifies robot i as the seeker. We use $pos(i)$ to denote the position vector for robot i and $vel(i)$ to denote the velocity of robot i, as estimated from the difference of two adjacent position observations.

1) Intercept and Transition Features: $f = I(y_t = i)$ and $f = I(y_{t-1} = i)I(y_t = j)$. These features allow the model to encode prior probabilities $P(y_t)$ and transition probabilities $P(y_t \mid y_{t-1})$.

2) Raw Observations: $f = I(y_t = i)x_t[k]$ and $f = I(y_t = i)x_t[k]^2$. These features allow the model to estimate sufficient statistics corresponding to modeling each component of the observation x_t as a one dimensional normal distribution.

3) Robot Velocities: $f = I(y_t = i)vel(j)$ Velocities are of interest because the seeker robot pauses in place after being tagged.

4) Chasing Features: $f = I(y_t = i)(pos(j) - pos(k))^T vel(k)$. As an example of a more complex feature, the chasing features attempt to capture whether or not robot k appears to be chasing the closest other robot j. It computes the direction from robot k to robot j and then projects the velocity of robot k onto this direction to determine if robot k is moving towards robot j.

5) Velocity Correlations: $f = I(y_t = i)vel(j)^T vel(k)$ We included features to capture the correlation between the velocities of robot j and robot k by considering the inner product of their velocities.

6) Velocity Thresholds: $f = I(y_t = i)I(vel(j) \leq k)$ These features test whether the velocity of robot j falls below

TABLE I

ERROR RATES WITH DIFFERENT REGULARIZERS

	Simulation			Real Robots		
Regularizer	None	ℓ_1	ℓ_2	None	ℓ_1	ℓ_2
Num. Features	1156	58	1156	1174	82	1174
Error Rate (%)	27.9	1.15	1.15	30.7	6.27	7.42

TABLE II

GENERALIZATION BETWEEN SIMULATED AND REAL ROBOT DATA

Training Data Source	Test Set Data Source	Error Rate (%)
Real Robots	Real Robots	6.27
Real Robots	Simulation	1.10
Simulation	Simulation	1.15
Simulation	Real Robots	6.36

a threshold k. The thresholds k were chosen to range from 5 to 100 mm/sec in steps of 5 mm/sec.

7) Distance Thresholds: $f = I(y_{t-1} = i)I(y_t = j)I((pos(a) - pos(b))^T(pos(a) - pos(b)) \leq k^2)$ Distance thresholds k where chosen from 200 mm to 1000 mm in steps of 50 mm. Recall that the robots have radii of 100 mm; distance thresholds ranged from robots touching to almost a meter apart. We also include equivalent features to indicate when the distance was greater than the same thresholds.

B. Error Rates with Different Regularizers

The results in table I show the classification error rates on the robot tag data when no regularization is used and when the CRF is trained under either an ℓ_1 or ℓ_2 penalty. The results show that some form of regularization is necessary to achieve high classification accuracy; models containing hundreds or thousands of features suffer from over-fitting in the absence of smoothing. Also of note is the number of features in the final model under each of the three schemes. Both the no regularization and ℓ_2 penalty cases include all, in the case of the real robot data, or nearly all, with the noise-less simulated data, of the 1174 features. This is expected as neither of those techniques is known for producing sparse models. When ℓ_1 regularization is used, the CRF achieves comparable accuracy to the larger models, but only using 7 and 5 percent of the total features in the real robot data and simulated data cases respectively.

C. Generalization between Simulated and Real Robot Data

Table II shows the error rates on test data when the features selected by ℓ_1 regularization and the corresponding weights are applied to different data sets. It is encouraging to note that the features and weights learned when the CRF was trained on data from real robots perform well on data from the simulated robots and vice versa. It is typically much easier to generate large data sets, which may be required for more complex activity recognition tasks, from simulations and these results suggest that features discovered on such large simulated data sets may carry over to real robots.

D. Retraining a CRF with Only the Selected Features

Training models under an ℓ_1 penalty produces both an active subset of the features as well as a set of weights for those features. The set of experiments summarized by table III considers the question of whether or not the weights from ℓ_1 training should be used directly or if a new model, containing only the selected features, should be retrained, possibly with regularization; the role of regularization during this second iteration of training would be smoothing only, so we examined ℓ_2 regularization rather than ℓ_1 because the former translates into an unconstrained optimization problem and generally requires less computation to solve. Also of note is that this second round of optimization may be much less computationally expensive than the first round where features were selected; many fewer features are present in the second model and the reduced amount of data may make caching strategies much more effective.

TABLE III

RETRAINING THE CRF WITH ONLY THE SELECTED FEATURES

Feature Selection Data Source	Retraining Data Source	Regularizer for Retraining	Error Rate (%)
Real Robots	Real Robots	None	9.55
Real Robots	Real Robots	ℓ_2	6.88
Simulation	Real Robots	None	10.8
Simulation	Real Robots	ℓ_2	5.83
Simulation	Simulation	None	5.15
Simulation	Simulation	ℓ_2	1.21
Real Robots	Simulation	None	4.17
Real Robots	Simulation	ℓ_2	1.27

The results in table I show that error rates of 6 or 7 percent are expected on the real robot data and error rates of approximately 1 percent are expected on the simulated data when the full compliment of features is used and the CRF is smoothed during training. The results in table III show that retraining the CRF with only the selected features to optimize the level of smoothing is not necessary. Additionally, experiments where features selected with one data set are used in a CRF trained on the other data set support the results in table II that show good generalization between real and simulated data.

E. Feature Selection Results

The results in figure 3 show the error rate on the hold out set and the number of features with non-zero weights in the model as the regularization parameter λ is varied. In both the real and simulated data cases, the error rate drops off suddenly as informative features enter into the model. This drop off to low error rates is then followed by a gradual increase in the error as additional features increase the amount of over-fitting. The key observation is that the error rate approaches a minimum value when the number of active features in the model remains modest. Feature selection is yielding both improved accuracy and a reduced computational cost (by reducing the number of features) in the final model.

V. CONCLUSION

We have shown that ℓ_1 regularization is an effective method for smoothing and feature selection in conditional random fields. Feature selection is particularly important in

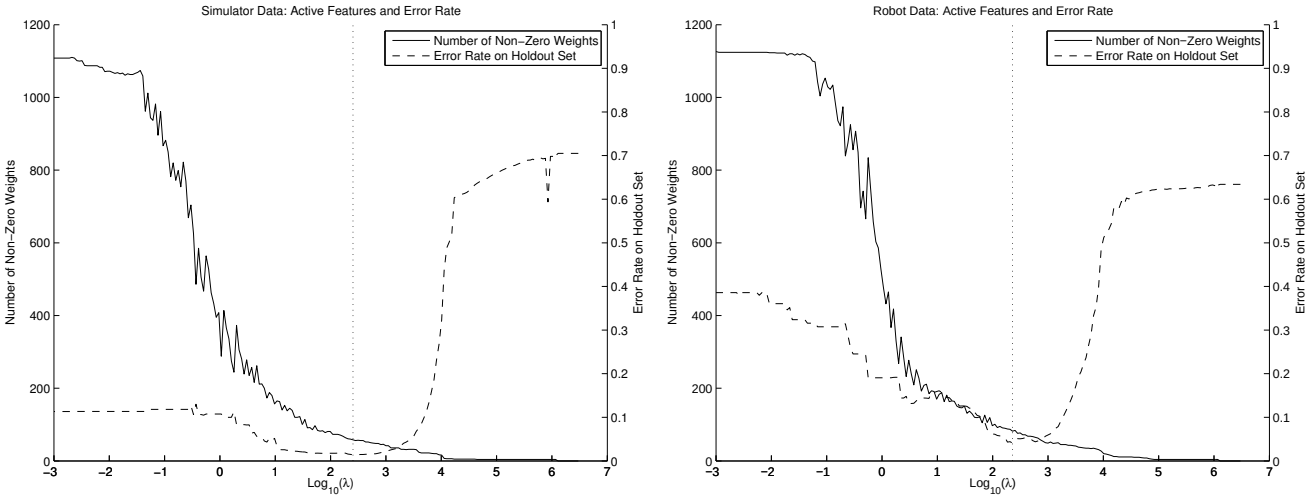

Fig. 3. The error rate on the hold out set and number of features in the tag domain for CRFs trained on simulated data (left) and on data from real robots (right). On the right side of each plot, the regularization parameter, λ, is assigned a high value and no features are included in the model. As λ is decreased, the number of features with non-zero weights (solid line, scale on left axis) increases. At the same time, the error rate of the classifier trained with the corresponding λ (dashed line, scale on right axis) decreases. In both domains, the error rate declines sharply while the number of features in each model is relatively low. The error rate increases again for low values of λ as the model includes less relevant features and over-fits on the training data. The dotted vertical lines indicate the setting of λ with the lowest error rate on the hold out set.

robot domains, where classifiers depend on complex features to extract information from their sensor readings, because robots have a limited amount of processing power that must be shared by many systems. Results in an activity recognition domain using data from both real and simulated robots show that ℓ_1 regularization can produce error rates comparable to those produced by other smoothing algorithms while at the same time eliminating more than 90% of the candidate features in the model.

We have also shown that reparameterization coupled with projected conjugate gradient is an effective method to recover the ℓ_1 regularization path. This serves as a simple alternative to, e.g. interior point methods such as the one in [7], which have been applied to logistic regression.

VI. Acknowledgments

The authors extend their warm thanks to James Bruce and Stefan Zickler for their help with the CMDragons robot soccer system. This research was sponsored in part by the Department of the Interior, National Business Center under contract no. NBCHD030010, SRI International under subcontract no. 03-000211, and NSF grant IIS-0427206.

References

[1] Dimitri P. Bertsekas. Projected newton methods for optimization problems with simple constraints. *SIAM Journal on Control and Optimization*, 20(2):221–246, 1982.

[2] James Bruce, Michael Bowling, Brett Browning, and Manuela Veloso. Multi-robot team response to a multi-robot opponent team. In *Proceedings of ICRA'03, the 2003 IEEE International Conference on Robotics and Automation*, Taiwan, May 2003.

[3] James Bruce and Manuela Veloso. Fast and accurate vision-based pattern detection and identification. In *Proceedings of ICRA'03, the 2003 IEEE International Conference on Robotics and Automation*, Taiwan, May 2003.

[4] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Ann. Statist.*, 32(2):407–499, 2004.

[5] T. Hastie, R. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer, August 2001.

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

[7] K. Koh, S.J. Kim, and S. Boyd. An interior-point method for large-scale l1-regularized logistic regression. *Under Submission*, 2006.

[8] S. Kumar and M. Hebert. Discriminative random fields: A discriminative framework for contextual interaction in classification, 2003.

[9] John Lafferty, Andrew McCallum, and Fernando 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.

[10] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y. Ng. Efficient l1 regularized logistic regression. In *AAAI*, 2006.

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

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

[13] Mee-Young Park and Trevor Hastie. An L1 regularization path algorithm for generalized linear models. *JRSSB*, To Appear.

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

[15] David A. Ross, Simon Osindero, and Richard 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.

[16] Fei Sha and Fernando 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.

[17] Cristian Sminchisescu, Atul Kanaujia, Zhiguo Li, and Dimitris Metaxas. Conditional models for contextual human motion recognition. In *IEEE International Conference on Computer Vision (ICCV 2005)*, pages 1808–1815, 2005.

[18] Douglas L. Vail, Manuela M. Veloso, and John D. Lafferty. Conditional random fields for activity recogntion. In *AAMAS*, 2007.

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