Feature Selection for Activity Recognition in Multi-Robot Domains

Douglas L. Vail and **Manuela M. Veloso**
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA USA
{dvail2,mmv}@cs.cmu.edu

Abstract

In multi-robot settings, activity recognition allows a robot to respond intelligently to the other robots in its environment. Conditional random fields are temporal models that are well suited for activity recognition because they can robustly incorporate rich, non-independent features computed from sensory data. In this work, we explore feature selection in conditional random fields for activity recognition to choose which features should be included in the final model. We compare two feature selection methods, grafting, a greedy forward-selection strategy, and ℓ_1 regularization, which simultaneously smoothes the model and selects a subset of the features. We use robot data recorded during four games of the Small Size League of the RoboCup'07 robot soccer world championship to empirically compare the performance of the two feature selection algorithms in terms of accuracy of the final model, the number of features selected in the final model, and the time required to train the final model.

Introduction

Activity recognition is an important component for creating intelligent robot systems. For example, robot soccer is a domain where activity recognition has the potential to make a large contribution. In robot soccer, a team of robots can use activity recognition to classify the roles of their opponents and base strategic decisions on that classification. In general, activity recognition is important in any domain where robots must act intelligently in the presence of other agents.

Activity recognition is a temporal classification problem. A robot maps from a temporal sequence of observations to the roles of the other agents in its environment. Roles extend across time steps and a single role may consist of several actions. Returning to robot soccer, a *defender* in soccer blocks shots, dribbles the ball around opponents, and passes the ball upfield; all three actions are different, but all fall under the single role of *defender*. Activity recognition is challenging because observations do not directly map to roles. We must infer roles from low level information extracted from sensory data, such as positions and velocities. In practice, we train classifiers to map from *features* of the observations to roles rather than directly from observations to roles. Features are functions of the observations that inject domain knowledge into the classification by transforming the observations into a more useful form for the classifier. As an example, in robot soccer, the distance to the ball is an important feature. Rather than passing the coordinates of the robot and the ball to the classifier, we input the actual distance instead.

Choosing an appropriate set of features is important for accurate activity recognition. As humans designing features, we can easily define prototypes for good candidate features. For example, in soccer, a feature that tests if the distance between two objects is less than a threshold is useful. Instantiating this prototype with different pairs of objects and different thresholds results in a large pool of candidate features, particularly in multi-robot domains where the number of objects and relationships between objects is large. We use *feature selection* to choose a small subset of the candidate features to include in the final model. Reducing the number of candidate features is important to reduce over-fitting, which would reduce the accuracy of the final model, and to reduce the computational cost of classification so that roles may be recognized and responded to online.

In this paper, we focus on feature selection for activity recognition in multi-robot domains. Specifically, we consider two approaches to feature selection in conditional random fields (Lafferty, McCallum, & Pereira 2001). We consider grafting, a greedy forward-selection algorithm (Perkins, Lacker, & Theiler 2003), and ℓ_1 regularization (Hastie, Tibshirani, & Friedman 2001), which simultaneously selects features and smoothes the model. We compare the algorithms using robot data that were recorded (and generously shared) by the CMDragons'07 team (Bruce, Veloso, & Zickler 2008) during four games of the RoboCup'07 robot soccer world championship (Kitano 1998).

The RoboCup Small Size League

We compare feature selection via grafting and ℓ_1 regularization with robot data that were recorded by the CMDragons'07 robot soccer team during the Small Size League games of RoboCup 2007. In the Small Size League, two teams with five robots per team compete in a twenty minute soccer match, as shown in figure 1. Each team is fully autonomous and controlled wirelessly by an off-board computer. The computers control the robots using global infor-

Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Figure 1: In the Small Size League, two teams with five robots per team compete in a soccer match. The robots use rotating rubber bars to grip a golf ball for "dribbling" and can kick the ball at velocities of up to 10 m/s to pass or shoot on goal. The robots are controlled wirelessly by an off-board computer that tracks all ten robots and the balls using overhead cameras mounted above the field.

mation from overhead cameras. The robots are 18 cm in diameter and the field is approximately 5 m by 3 m in size. Due to the scale of the robots, a golf ball is used as a soccer ball. The robots move at up to 2.5 m/s and can kick the ball at velocities of up to 10 m/s, which results in fast-paced games. The role of goalie on each team is fixed, but the remaining four robots take on a variety of offensive and defensive roles as the game evolves. These roles are what we recognize during classification.

Roles

In principle, each team defines its own unique set of roles. In practice, many roles are shared across teams. We would like to apply a classifier to new opponents to identify roles from this common set. However, in the current work, our training data comes from a single team and we are limited to predicting the roles of the robots from that team. To test generalization with our limited data, we use two games as a training set, a single game as a hold out set to choose model parameters, and test exclusively on data from the final match. The final is never used as a training or hold out set. Table 1 lists the roles of the CMDragons'07 robot soccer team, many of which are general and observed in other teams as well. The classification task, which we formalize below, is to recognize those roles from the available sensor data, which consists of the positions of the robots and the ball. The goal is to create a classifier that can provide useful information to robots that are playing against the team whose roles are being classified.

The Classification Task

The classification task is to map from a sequence of *observations* about the world $X = \{x_1, x_2, ..., x_T\}$ to a sequence of *labels* $Y = \{y_1, y_2, ..., y_T\}$. We use x_t and y_t to refer to observations or labels from a single time step t and

Role Name	Description
Kickoff	Position before play starts
Mark Opponent	Man-to-man defense against a particular robot
Position for Pass	Create openings for passing
Receive Chip	Receive an incoming chip pass
Position	Position on the field when not executing specialized behaviors to receive passes
Penalty Kick	Take a penalty kick.
Wall	Form a wall with teammates to block a kick
Set-Play-Kick	A coordinated play where one robot passes the ball to another which deflects the pass towards the goal
Attacker	Primary offensive role
Defend Circle	Defend the goal at a fixed radius

Table 1: Roles of the CMDragons'07 robot soccer team.

T as the length of the sequence. Individual observations x_t are vectors of real values that contain the observed information from a single time step. In our domain, x_t contains 32 real valued elements that specify the positions and orientations of the ten robots as (x, y, θ) and the position of the ball (x, y). The labels y_t are drawn from the set of roles listed in table 1 and correspond to the role of a single robot at time t. Notably, the label y_t does not contain the joint role for all robots; recovering the roles of multiple robots requires running several classifiers, one per robot, in parallel. We present our rational for using multiple, independent classifiers after we introduce CRFs below.

Conditional Random Fields

Conditional random fields are undirected graphical models for structured classification (Lafferty, McCallum, & Pereira 2001). In our setting, we use CRFs, where the labels form a linear chain, to represent the conditional probability of the label sequence given an observation sequence. Conditional random fields are built from a vector of weights, which we learn during training, and a vector of *features*. There is a 1-to-1 correspondence between weights and features. The features take the form $f_i(t, y_{t-1}, y_t, X)$, where i is an index in the feature vector f, t is an offset into the sequence, y_{t-1}, and y_t are the values of the label pair at times $t-1$ and t respectively and X represents the entire observation sequence across all values of t. The conditional probability of a label sequence given an observation sequence is computed from the weighted sum of the features as:

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

$$Z_X = \sum_{Y'} \prod_{t=1}^{T} \exp(w^T f(t, y'_{t-1}, y'_t, X)) \quad (2)$$

Z_X is a normalization constant. Z_X can be computed efficiently via dynamic programming for tree-structured mod-

els (Sutton & McCallum 2006).

We train conditional random fields by finding a weight vector w^* that maximizes the conditional log-likelihood of labeled training data:

$$\ell(Y|X;w) = w^T f(t, y_{t-1}, y_t, X) - \log(Z_X) \quad (3)$$

$$w^* = \underset{w}{\operatorname{argmax}}\, \ell(Y|X;w) \quad (4)$$

Features in conditional random fields are functions of the form $f_i(t, y_{t-1}, y_t, X)$. We use the *indicator function* I, which evaluates to 1 if its argument is true and 0 otherwise, to test whether the labels y_{t-1} and y_t match particular roles. Features sometimes include a function $g(t, X)$, which could, for example, compute the distance between two robots at time t. Common feature prototypes are:

$$f_i(t, y_{t-1}, y_t, X) = I(y_t = \mathit{role}) \quad (5)$$

$$f_j(t, y_{t-1}, y_t, X) = I(y_{t-1} = \mathit{role}_1) I(y_t = \mathit{role}_2) \quad (6)$$

$$f_k(t, y_{t-1}, y_t, X) = I(y_t = \mathit{role}) g(t, X) \quad (7)$$

Equation 5 tests whether y_t takes on a particular value. After training, its weight will be proportional to the log-likelihood of y_t = role. Similarly, equation 6 captures first-order Markov transition dynamics and, after training, its weight is proportional to the log-likelihood of the transition. Equation 7 adds information from the observation sequence in the form of an arbitrary, real valued function $g(t, X)$. Such features can be viewed as compatibility functions that, with their associated weights, link the likelihood of being in particular states y_t to information in the observation sequence.

CRFs for Activity Recognition

Conditional random fields are well suited to activity recognition because they can incorporate complex, non-independent features of the entire observation sequence. In general, discriminatively trained models, such as CRFs, are more accurate than equivalent generative models, such as hidden Markov models (Ng & Jordan 2002). In the particular setting of robot activity recognition, we have found that CRFs are indeed more accurate than HMMs (Vail, Veloso, & Lafferty 2007), which is why we chose CRFs over hidden Markov models, even though HMMs have previously been used for activity recognition in robot soccer (Han & Veloso 2000).

Robot soccer is a multi-robot activity recognition problem; dependencies exist between the roles of the different robots. However, exact inference is NP-hard if we directly model dependencies with links between separate label chains, e.g. using a dynamic conditional random field (Sutton, McCallum, & Rohanimanesh 2007). Approximate inference techniques, such as loopy belief propagation are applicable (Murphy, Weiss, & Jordan 1999), but they are slower than exact inference in models that treat the labels as independent, linear chains.

As a compromise between NP-hard exact inference and ignoring dependencies between robots, we can create features that provide information about the roles of the other robots. Such features could incorporate actual predictions of the other robot roles, e.g.,

$$f_i(t, y_{t-1}, y_t, X) = I(y_t = r_1\text{'s role}) \; I(g_{r_2}(t, X) = r_2\text{'s role}) \quad (8)$$

where the CRF is predicting the role of robot 1 and g_{r_2} is a classifier that predicts the role of robot 2 based on the observations, but not on the role of robot 1. Alternately, we can use *relational features* to capture information about the roles of the other robots.

We define relational features in terms of relationships between objects in the environment. For example, to determine which team has possession of the ball in robot soccer, we compute the distance between the ball and the robot on each team that is closest to the ball. We use the relationship *closest to*, defined with respect to an object (the ball), to dynamically choose an object (one robot) from a set of candidate objects (a team of robots). Relational features compute quantities, e.g. distances, using information from the observations to dynamically select objects in the environment.

Relational features provide a succinct means for specifying features in terms of the relationships between different robots in multi-robot domains. They also provide a way of incorporating information about the roles of other robots into the classifier. For example, in robot soccer, the robot that is closest to the ball is usually in the *attacker* role. We can define a function $h(t, X)$ as the distance between the ball and the subject of the classification's teammate (not including the subject) that is closest to the ball. We then create

$$f_i(t, y_{t-1}, y_t, X) = I(y_t = \text{attacker}) h(t, X) \quad (9)$$

that helps the CRF predict the subject's role; when another teammate is close to the ball, it is unlikely that the subject is the attacker. These sorts of features that provide information about the roles of other robots offer a compromise between NP-hard exact inference if we include links directly between label chains and the extra error that results from not including any information about the roles of others.

Feature Selection

We compare two embedded feature selection algorithms, grafting (Perkins, Lacker, & Theiler 2003) and ℓ_1 regularization (Hastie, Tibshirani, & Friedman 2001). Embedded algorithms combine feature selection with the process of training the model (Blum & Langley 1997). The algorithms that we compare produce a series of candidate models, where each candidate generally contains more features than the preceding candidate model. After generating many candidate models, we use held out data to choose among them.

Grafting

Grafting is an embedded feature selection method that uses a greedy forward selection strategy to choose features while training the model (Perkins, Lacker, & Theiler 2003). Grafting begins with no features and adds one feature per iteration. We retrain the model after adding a new feature using the weights from the previous iteration, which contained one fewer feature, as a starting point. Starting with the weights from the previous iteration greatly speeds training. Grafting

chooses which feature to add based on the gradient of the objective function (the conditional log-likelihood) with respect to the weights. Grafting selects the feature whose weight makes the largest absolute contribution to the gradient.

ℓ_1 Regularization

Training a model under an ℓ_1 penalty results in a sparse model where many of the weights are exactly zero (Hastie, Tibshirani, & Friedman 2001) and ℓ_1 training is an effective method for feature selection in CRFs (Vail, Lafferty, & Veloso 2007). To train a CRF with an ℓ_1 penalty, we find w^* that maximizes:

$$w^*_\lambda = \underset{w}{\text{argmax}}\ \ell(Y|X) - \lambda \sum_i |w_i|, \qquad (10)$$

The penalty term in 10 is all that differs from the standard objective function. λ is a positive scalar value that controls the degree of smoothing and the sparsity of the final model. High values of λ result in sparser models. The penalty term is not differentiable at zero and special care must be taken during training to address this. Koh et al. and Andrew & Gao describe training in logistic regression, which is an equivalent task (Koh, Kim, & Boyd 2006; Andrew & Gao 2007). In our experiments, we used the orthant-wise projected L-BFGS method of Andrew and Gao. To select features via ℓ_1 regularization, we vary the smoothing parameter λ and use a hold out set to select among the resulting models. We begin with λ_0 chosen such that w^* = 0 and decay it according to $\lambda_{k+1} = .95 \cdot \lambda_k$. We initialize each succeeding round using the weights from the previous round of training, which greatly speeds the convergence of each iteration.

Other Feature Selection Methods

Other methods for feature selection in conditional random fields include a greedy forward selection algorithm that uses a fast heuristic to induce features for named entity recognition and noun phrase chunking tasks in text data due to McCallum (McCallum 2003). McCallum's algorithm selects features to add by training a relaxed version of the model with each candidate feature and computing the gain in the relaxed model. In a more closely related domain, Liao et al. chose features by removing links in the model between time steps and introduced a modification of Boosting (Freund & Schapire 1996) that takes virtual evidence into account to select features for activity recognition (Liao *et al.* 2007).

Experiments

We used data recorded by the CMDragons'07 small size team to compare grafting and ℓ_1 regularization for feature selection in conditional random fields. We used data from the final match as a test set and data from a semi-final, quarter-final, and one round robin game as the training and hold out sets. The hold out set was used to choose parameters such as λ and to choose among the models produced by grafting. The CMDragons system operates at a frame rate of 60 hz. To reduce the amount of data, we subsampled from the full data sets at a rate of 2 hz.

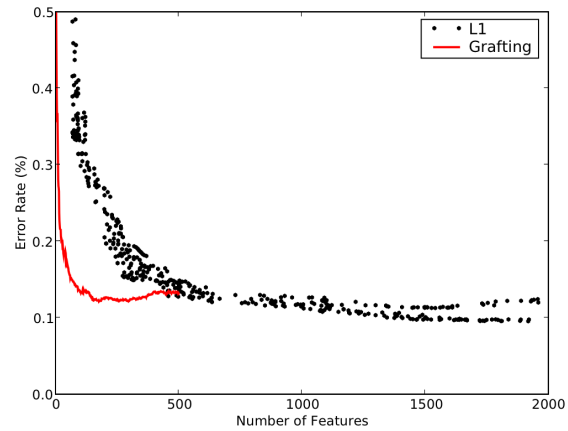

Figure 2: Error rate on test data versus model size is shown for the successive models proposed by grafting and ℓ_1 regularization. Grafting quickly converges to a minimum and then shows evidence of over-fitting. ℓ_1 regularization produces larger models, but achieves a lower error rate overall.

Features

We created feature prototypes, like those shown in 7, and instantiated them to create a large number (92,310) of candidate features. Typical prototypes took the form

$$f_i(t, y_{t-1}, y_t, X) = I(y_t = \mathit{role})g(t, X) \qquad (11)$$

where $g(t, X)$ was chosen to capture key properties of the robot soccer domain such as:

$$g(t, X) = \text{distance}(\text{select-closest}(\text{teammates}, \text{ball}), \text{ball}) \qquad (12)$$

We refer to 12 as a *relational feature* because it is defined in terms of relationships between objects in the environment that are detected online. The function $g(t, X)$ selects the robot that is closest to the ball from the set *teammates*, which includes all of the other robots on the same team as the robot whose role is being classified, and computes the distance between that closest robot and the ball. The feature is relational because the robot is specified dynamically rather than statically by specifying a robot id.

We defined approximately thirty relational feature prototypes, by instantiating them with different combinations of roles and robot ids, generated 92,310 candidate features. We considered distances between the robots, the ball, the two goals, and the center of the field. We included raw distances as a feature as well as binary features testing if distances were above or below a threshold for a wide range of thresholds. We also included binary features testing if one robot or object was closer to another versus a third object.

Results

In feature selection, there is a trade-off between the size of the model and the accuracy of the model. Up to a point, adding features will decrease the error rate of the model.

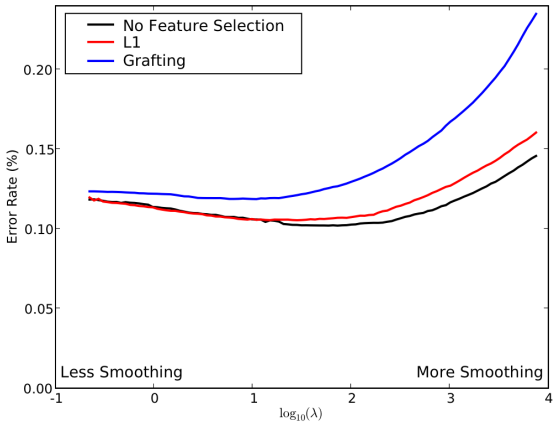

Figure 3: Error rate on test data versus the smoothing parameter λ is shown for retraining models that contain only the features selected by grafting and ℓ_1 regularization using ℓ_2 regularization for smoothing. On average, grafting selected 220 features and ℓ_1 regularization selected 1823. For comparison, we also show the ℓ_2 regularization path for the no feature selection case that included all 92,310 features. Grafting, which uses the fewest features shows the highest error rate. ℓ_1 regularization, with an intermediate number of features performs almost identically to the CRF with all 92,310 features.

Past a certain point, additional features increase the error rate of the model due to over-fitting. Figure 2 shows error rate versus model size for grafting and ℓ_1 regularization. Grafting, the purely greedy algorithm, shows a more rapid decrease in the error rate as features are added. ℓ_1 regularization shows a less steep decline in the error rate, but achieves, on average, a lower error rate than grafting because it is less greedy and able to remove or swap out one feature for another during the course of training. The results in figure 2 are from three trials where a round-robin, quarter-final, and semi-final game were each used in turn as the hold out set and the other two were used for training. The final was always used as the test set. The results for grafting, which predictably adds a single feature per iteration are shown as an average over the three trials. The results for ℓ_1 regularization are plotted as independent points for each trial because the model size changed unpredictably between iterations.

Grafting does not explicitly smooth the final model. We explored using ℓ_2 regularization to smooth models containing only the selected features to see if it would lower the error rate further. As a comparison, we also applied ℓ_2 regularization to smooth the full model that contained 92,310 features. Figure 3 shows the average error rate on test data as the smoothing parameter λ varies during ℓ_2 penalized training. Briefly, an ℓ_2 penalty is analogous to an ℓ_1 penalty, except the model is penalized by $w^T w$ rather than by $|w|$. With smoothing, grafting shows a higher error rate than either ℓ_1 regularization or the full model (both also with smoothing), possibly because grafting omits relevant features by being too greedy during the feature selection process. The features selected by ℓ_1 regularization achieve an error rate that is virtually identical to the full model even though the former is a small subset of the features present in the full model.

Training	Error Rate	Model Size	Time (hours)
ML	15.7%	92,310	3.1
ML / ℓ_2	10.5%	92,310	11.2
ℓ_1	10.3%	1,823	6.9
ℓ_1 / ℓ_2	10.7%	1,823	1.7 (8.6)
Grafting	12.3%	220	1.9
Grafting / ℓ_2	12.0%	220	.6 (2.5)

Table 2: While training on data from three earlier games and testing with data from the final, we compared maximum likelihood training with no smoothing; maximum likelihood training with ℓ_2 smoothing; ℓ_1 regularization for feature selection / smoothing; ℓ_1 regularization for feature selection followed by ℓ_2 smoothing; grafting for feature selection; and grafting for feature selection followed by ℓ_2 smoothing. We show average error rates on test data for the chosen subset of features, the size of the final model, and the training time required to discover the final model. When feature selection was followed by smoothing, we show the time required for smoothing followed by the cumulative time for feature selection and smoothing in parentheses.

Table 2 gives the error rates for the models selected using the hold out set on the test data. Both the regular and ℓ_2 smoothed models produced via ℓ_1 regularization perform almost identically to the full model with ℓ_2 smoothing. Some sort of smoothing, either by selecting a small subset of the available features or by applying an ℓ_2 penalty is necessary to achieve low error rates as the full model without ℓ_2 smoothing has the highest error rate. Grafting falls in the middle between the most accurate training methods and the unsmoothed full model. However, if we use the size of the final model or the training time required to discover the final model as a metric, then grafting comes out ahead. On average, grafting discovers smaller models more quickly than ℓ_1 regularization. And both feature selection algorithms discover the final model in less time than it takes to train the full model with ℓ_2 smoothing, which provides strong motivation for using feature selection.

Table 2 shows that we can accurately predict the roles of a single team against different opponents. Table 3 provides similar results for the case where we train the model using data from the first half of a game (the final) and test against data from the second half of the same game. The motivation for this experiment is to show that we can, in principle, use only data gathered during the first half of a game to perform accurate activity recognition during the second half. In reality, there is not enough time between halves of a robot soccer game to label training data and train a model. However, these results show that in general, conditional random fields trained by grafting or ℓ_1 regularization can achieve high accuracies for activity recognition from a limited amount of training data.

Training	Error Rate	Model Size
ML	18.6%	92310
ℓ_1	8.0%	1449
Grafting	10.9%	137
Grafting / ℓ_2	10.6%	137

Table 3: We tested generalization across different portions of the same game (the final) by training on date from the first half of the final and testing on data from the second half.

Conclusion

Conditional random fields are well suited to activity recognition in multi-robot domains because they robustly incorporate large numbers of complex, non-independent features. Feature selection can dramatically reduce the number of features required by CRFs to achieve error rates that are close to, in the case of grafting, or identical to, with ℓ_1 regularization, the error rate achieved by the model with its full compliment of features. Reducing the number of features dramatically speeds up online classification. Surprisingly, feature selection also speeds training, even though additional work is being performed to select the best features for the final model; both feature selection algorithms that we compared required less training time than the full model with ℓ_2 smoothing. As some form of smoothing, be it via feature selection or ℓ_2 regularization, is required to achieve the lowest error rates, there is a strong case for performing feature selection when using CRFs for activity recognition. Our empirical comparison of grafting and ℓ_1 regularization showed that grafting produced smaller models, but with a penalty to the accuracy of the models. In contrast, ℓ_1 regularization chose more features than grafting, but performed as well as the full model with ℓ_2 smoothing.

Acknowledgments

The authors warmly thank James Bruce and Stefan Zickler for sharing the CMDragons'07 log data from the RoboCup'07 games. The CMDragons'07 team consisted of James Bruce, Michael Licitra, Stefan Zickler, and Manuela Veloso. This research was sponsored in part by United States Department of the Interior under Grant No. NBCH-1040007, and in part by BBNT Solutions under subcontract no. 950008572. The views and conclusions contained in this document are those of the authors only.

References

Andrew, G., and Gao, J. 2007. Scalable training of L1-regularized log-linear models. *Proceedings of the 24th International Conference on Machine learning* 33–40.

Blum, A., and Langley, P. 1997. Selection of relevant features and examples in machine learning. *Artificial Intelligence* 97(1-2):245–271.

Bruce, J. R.; Veloso, M.; and Zickler, S. 2008. CMDragons: Dynamic Passing and Strategy on a Champion Robot Soccer Team. In *Proceedings of ICRA'2008*.

Freund, Y., and Schapire, R. 1996. Experiments with a new boosting algorithm. *Machine Learning: Proceedings of the Thirteenth International Conference* 148:156.

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

Hastie, T.; Tibshirani, R.; and Friedman, J. H. 2001. *The Elements of Statistical Learning*. Springer.

Kitano, H., ed. 1998. *RoboCup-97: Robot Soccer World Cup I*. London, UK: Springer-Verlag.

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

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

Liao, L.; Choudhury, T.; Fox, D.; and Kautz, H. 2007. Training conditional random fields using virtual evidence boosting. In *IJCAI*, 2530–2535.

McCallum, A. 2003. Efficiently inducing features of conditional random fields. *Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI03)*.

Murphy, K.; Weiss, Y.; and Jordan, M. 1999. Loopy belief propagation for approximate inference: An empirical study. *Proceedings of Uncertainty in AI* 467–475.

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

Perkins, S.; Lacker, K.; and Theiler, J. 2003. Grafting: fast, incremental feature selection by gradient descent in function space. *The Journal of Machine Learning Research* 3:1333–1356.

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

Sutton, C.; McCallum, A.; and Rohanimanesh, K. 2007. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *The Journal of Machine Learning Research* 8:693–723.

Vail, D. L.; Lafferty, J. D.; and Veloso, M. M. 2007. Feature selection in conditional random fields for activity recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Vail, D. L.; Veloso, M. M.; and Lafferty, J. D. 2007. Conditional random fields for activity recogntion. In *AAMAS*.