In Advances in Neural Information Processing Systems (NIPS-11)
pages 396-402, Denver, Colorado, December 1998

Approximate learning of dynamic models

Xavier Boyen
Computer Science Dept. 1A
Stanford, CA 94305-9010
xb@cs.stanford.edu

Daphne Koller
Computer Science Dept. 1A
Stanford, CA 94305-9010
koller@cs.stanford.edu

Abstract

Inference is a key component in learning probabilistic models from partially observable data. When learning temporal models, each of the many inference phases requires a complete traversal over a potentially very long sequence; furthermore, the data structures propagated in this procedure can be extremely large, making the whole process very demanding. In [2], we describe an approximate inference algorithm for monitoring stochastic processes, and prove bounds on its approximation error. In this paper, we apply this algorithm as an approximate forward propagation step in an EM algorithm for learning temporal Bayesian networks. We also provide a related approximation for the backward step, and prove error bounds for the combined algorithm. We show that EM using our inference algorithm is much faster than EM using exact inference, with no degradation of the quality of the learned model. We then extend our analysis to the online learning task, showing a bound on the error resulting from restricting attention to a small window of observations. We present an online EM learning algorithm for dynamic systems, and show that it learns much faster than standard offline EM.

1 Introduction

In many real-life situations, we are faced with the task of understanding the underlying dynamics of a complex stochastic process from limited observations about its state over time. As in any inductive learning task, the first decision we must make concerns the representation of our learned hypotheses. Until now, *hidden Markov models (HMMs)* [12] have played the largest role as a representation for learning models of stochastic processes. Recently, however, there has been increasing use of more structured models of stochastic processes, such as *factored HMMs* [8] or *dynamic Bayesian networks (DBNs)* [4]. Such structured decomposed representations allow complex processes over a large number of states to be encoded using a much smaller number of parameters, allowing much better

generalization from limited amounts of data [8, 7, 13]. Furthermore, the natural structure of such processes makes it easier for a human expert to incorporate prior knowledge about the domain structure into the model, thereby improving its inductive bias.

Both parameter and structure learning algorithms for dynamic models [12, 7] use probabilistic inference as a crucial component. An inference routine is called multiple times in order to "fill in" missing data with its expected value according to the current hypothesis; the resulting *expected sufficient statistics* are then used to construct a new hypothesis. The inference step is used many times, each of which iterates over the entire sequence. This behavior is problematic in two important respects. First, in many settings, we may not have access to the entire sequence in advance. Second, the various structured representations of stochastic processes do not admit an effective inference procedure. The messages propagated by exact inference algorithms include an entry for each possible state of the system; the number of states is exponential in the size of our model, rendering this type of computation infeasible in all but the smallest of problems. In this paper, we describe and analyze an approach that helps us address both of these problems.

In [2], we proposed a new approach to approximate inference in stochastic processes, where *approximate* distributions that admit compact representation are maintained and propagated. We showed empirically that this approach allows orders of magnitude speedup for the task of monitoring a complex stochastic process, at only a small cost in terms of accuracy. We also proved that the accumulated error arising from the repeated approximations remains bounded indefinitely over time. Our analysis relied on a novel result showing that transition through a stochastic process is a contraction for relative entropy (KL-divergence) [3].

Here, we apply this approach to the parameter learning task. This application is not completely straightforward, since our algorithm of [2] and the associated analysis only applied to the forward propagation of messages, whereas the inference used in learning algorithms require propagation of information from the entire sequence. In this paper, we provide an analysis of the error accumulated by an approximate inference process in the backward propagation phase of inference. This analysis is quite different from the contraction analysis for the forward phase. We combine these two results to prove bounds on the error of the expected sufficient statistics relayed to the learning algorithm at each stage. We then present empirical results for a practical DBN, illustrating the performance of this approximate learning algorithm. We show that speedups of 15–20 can be obtained easily, with no discernable loss in the quality of the learned hypothesis.

Our theoretical analysis also suggests a way of dealing with the problematic need to reason about the entire sequence of temporal observations at once. Our contraction results show that it is legitimate to ignore observations that are very far in the future. Thus, we can compute a very accurate approximation to a belief state by considering only a small window of observations in the future. We use this idea to construct a very efficient online learning algorithm. We then show that it converges to a good hypothesis much faster than the standard offline EM algorithm, even in settings favorable to the latter.

2 Preliminaries

A model for a dynamic system is specified as a tuple $\langle \mathcal{B}, \Theta \rangle$ where $\mathcal{B}$ represents the qualitative structure of the model, and Θ the appropriate parameterization. In a DBN, the instantaneous state of a process is specified in terms of a set of state variables $X_1, ..., X_n$. Here, $\mathcal{B}$ encodes a network fragment which specifies, for each time t variable $X_k^{(t)}$, the set of parents $\mathit{Parents}(X_k^{(t)})$; an example fragment is shown in Figure 1(a). The parameters Θ define for each $X_k^{(t)}$ a conditional probability table $\boldsymbol{P}[X_k^{(t)} \mid \mathit{Parents}(X_k^{(t)})]$. We use $\mathcal{T}$ to denote the transition matrix for the stochastic process; i.e., $\mathcal{T}_{i,j}$ is the transition probability

from state i to state j. Note that this concept is well-defined even for a DBN, although in that case, the matrix is represented implicitly via the other parameters.

Our goal is to learn the model for stochastic process from partially observable data. To simplify our discussion, we focus on the problem of learning parameters for a known structure using the *EM (Expectation Maximization)* algorithm [5]; most of our discussion applies equally to other contexts (e.g., [7]). EM is an iterative procedure that searches over the space of parameter vectors for one which is a local maximum of the likeiihood function—the probability of the observed data D given Θ. We describe the EM algorithm for the task of learning HMMs; the extension to DBNs is straightforward. The EM algorithm starts with some initial (often random) parameter vector $\tilde{\Theta}$, which specifies a current estimate of the transition and observation matrices of the process $\tilde{\mathcal{T}}$ and $\tilde{\mathcal{O}}$. The EM algorithm computes the *expected sufficient statistics (ESS)* for D, using $\tilde{\mathcal{T}}$ and $\tilde{\mathcal{O}}$ to compute the expectation. In the case of HMMs, the ESS are an average, over t, of the joint distributions $\boldsymbol{\psi}^{(t)}$ over the variables at time $t-1$ and the variables at time t. A new parameter vector $\tilde{\Theta}'$ can then be computed from the ESS by a simple maximum likelihood step. These two steps are iterated until an appropriate stopping condition is met.

The $\boldsymbol{\psi}^{(t)}$ for the entire sequence can be computed by a simple forward-backward algorithm. Let $r^{(t)}$ be the response observed at time t, and let $\mathcal{O}_{r^{(t)}}$ be its likelihood vector ($\mathcal{O}_{r_j}(i) \triangleq \mathcal{O}(i,j)$ is the probability of observing response r_j in state s_i). The forward messages $\boldsymbol{\alpha}^{(t)}$ are propagated as follows: $\boldsymbol{\alpha}^{(t)} \propto (\boldsymbol{\alpha}^{(t-1)} \cdot \mathcal{T}) \times \mathcal{O}_{r^{(t)}}$, where $\times$ is the outer product.[1] The backward messages $\boldsymbol{\beta}^{(t)}$ are propagated as $\boldsymbol{\beta}^{(t)} \propto (\mathcal{T} \cdot (\boldsymbol{\beta}^{(t+1)} \times \mathcal{O}_{r^{(t+1)}})^{\top})^{\top}$. The estimated belief at time t is now simply $\boldsymbol{\alpha}^{(t)} \times \boldsymbol{\beta}^{(t)}$ (suitably renormalized); similarly, the joint belief $\boldsymbol{\psi}^{(t)}$ is proportional to $(\boldsymbol{\alpha}^{(t-1)} \times \boldsymbol{\beta}^{(t)} \times \mathcal{T} \times \mathcal{O}_{r^{(t)}})$.

This message passing algorithm has an obvious extension to DBNs. Unfortunately, it is feasible only for very small DBNs. Essentially, the messages passed in this algorithm have an entry for every possible state at time t; in a DBN, the number of states is exponential in the number of state variables, rendering such an explicit representation infeasible in most cases. Furthermore even highly structured processes do not admit a more compact representation of these messages [8, 2].

3 Belief state approximation

In [2], we described a new approach to approximate inference in dynamic systems, which avoids the problem of explicitly maintaining distributions over large spaces. We maintain our *belief state* (distribution over the current state) using some computationally tractable representation of a distribution. We propagate the time t approximate belief state through the transition model and condition it on our evidence at time $t+1$. We then approximate the resulting time $t+1$ distribution using one that admits a compact representation, allowing the algorithm to continue. In DBNs, for example, we considered belief state approximations where certain subsets of less correlated variables in the belief state are approximated as being independent. In this case, the approximation at each step consists of a simple projection onto the relevant marginals, which are used as a factored representation of the time $t+1$ approximate belief state. We also showed that the errors arising from the repeated approximation do not accumulate unboundedly, as the stochasticity of the process attenuates their effect. For our DBN approximation, we showed that a factor 15–20 reduction in running time can be achieved, at the cost of but a small error.

These results are directly applicable to the learning task, as the belief state is precisely the forward message in the forward-backward algorithm. Thus, we can apply this approach to

[1] $\vec{x}(i,j) \times \vec{y}(j,k) = \vec{z}(i,j,k)$ where $z_{i,j,k} = x_{i,j} \cdot y_{j,k}$.

the forward step, with the guarantee that the approximation will not lead to a big difference in the sufficient statistics. However, this technique does not fully resolve the computational problem associated with the inference task for learning, as the backward propagation phase is just as expensive as the forward phase. We want to apply the same type of approximate inference algorithm to the backward propagation as we did for the forward one, i.e., maintain and propagate a compactly represented approximate backward message $\tilde{\boldsymbol{\beta}}^{(t)}$.

In order to guarantee bounds on the accumulated error, we could simply extend our analysis of [2] to the backward message. However, this approach turns out to be problematic. Even if we have bounds on relative entropy error of both the forward and backward messages, bounds for the error of the $\boldsymbol{\psi}^{(t)}$ do not follow. The solution turns out to use an alternative notion of distance, which combines additively under Bayesian updating, albeit at the cost of weaker contraction rates.

Definition 1 Let $\boldsymbol{\rho}$ and $\tilde{\boldsymbol{\rho}}$ be two positive vectors of same dimension. Their *projective distance* is defined as $\boldsymbol{D}_{\text{Proj}}[\boldsymbol{\rho}, \tilde{\rho}] \stackrel{\triangle}{=} \max_{i,i'} \ln[(\boldsymbol{\rho}_i \cdot \tilde{\boldsymbol{\rho}}_{i'})/(\boldsymbol{\rho}_{i'} \cdot \tilde{\boldsymbol{\rho}}_i)]$.

We note that the projective distance is an upper bound on the relative entropy.

Lemma 1 $\boldsymbol{D}_{\text{Proj}}[\boldsymbol{\psi}^{(t)}, \tilde{\boldsymbol{\psi}}^{(t)}] \leq \boldsymbol{D}_{\text{Proj}}[\boldsymbol{\alpha}^{(t-1)}, \tilde{\boldsymbol{\alpha}}^{(t-1)}] + \boldsymbol{D}_{\text{Proj}}[\boldsymbol{\beta}^{(t)}, \tilde{\boldsymbol{\beta}}^{(t)}]$.

Based on the results of [1], we show that projective distance contracts when messages are propagated through the stochastic transition matrix, in either direction:

Lemma 2 *Let* $k = \min_{\{i,j,i',j':\mathcal{T}_{i,j}\cdot\mathcal{T}_{i',j'}\neq 0\}} \sqrt{(\mathcal{T}_{i,j'} \cdot \mathcal{T}_{i',j})/(\mathcal{T}_{i,j} \cdot \mathcal{T}_{i',j'})}$, *and define* $\kappa_{\mathcal{T}} \stackrel{\triangle}{=} 2 \cdot k/(1+k)$. *Then* $\boldsymbol{D}_{\text{Proj}}[\boldsymbol{\alpha}^{(t)}, \tilde{\boldsymbol{\alpha}}^{(t)}] \leq (1-\kappa_{\mathcal{T}}) \cdot \boldsymbol{D}_{\text{Proj}}[\boldsymbol{\alpha}^{(t-1)}, \tilde{\boldsymbol{\alpha}}^{(t-1)}]$, *and* $\boldsymbol{D}_{\text{Proj}}[\boldsymbol{\beta}^{(t)}, \tilde{\boldsymbol{\beta}}^{(t)}] \leq (1-\kappa_{\mathcal{T}}) \cdot \boldsymbol{D}_{\text{Proj}}[\boldsymbol{\beta}^{(t+1)}, \tilde{\boldsymbol{\beta}}^{(t+1)}]$.

We can now show that, if our approximations do not introduce too large an error, then the expected sufficient statistics will remain close to their correct value.

Theorem 3 *Let S be the ESS computed via exact inference, and let $\tilde{S}$ be its approximation. If the forward (resp. backward) approximation step is guaranteed to introduce at most ε (resp. δ) projective error, then $\boldsymbol{D}_{\text{Proj}}[S, \tilde{S}] \leq (\varepsilon+\delta)/\kappa_{\mathcal{T}}$, and therefore also $\boldsymbol{D}_{\text{KL}}[S \,\|\, \tilde{S}] \leq (\varepsilon+\delta)/\kappa_{\mathcal{T}}$.*

We applied our ideas to the task of parameter learning in DBNs, using the approximation scheme of [2], where messages are represented in a factored way as marginals over specified clusters of variables. The clique tree algorithm [10] was used to propagate the messages forward and backward. For example, to compute $\tilde{\boldsymbol{\beta}}^{(t)}$ from $\tilde{\boldsymbol{\beta}}^{(t+1)}$, we generate a clique tree over the DBN and incorporate $\tilde{\boldsymbol{\beta}}^{(t+1)}$ in it. Then, $\tilde{\boldsymbol{\beta}}^{(t)}$ is obtained by reading off the relevant marginals from the tree ($\tilde{\boldsymbol{\beta}}^{(t)}$ is implicitly defined as their product).

We tested our algorithms on the task of learning the parameters for the BAT network shown on Figure 1(a), used for traffic monitoring [6]. The training set was a fixed sequence of 400 slices, generated from the correct network distribution. Our test metric was the average log-likelihood (per slice) of a fixed test sequence of 50 slices. All experiments were conducted using three different random starting points for the parameters (the same in all the experiments). We ran EM with different types of structural approximations, and evaluated the quality of the model after each iteration of the algorithm. We used four different structural approximations: (i) exact propagation; (ii) a 5+5 clustering of the ten state variables; (iii) a 3+2+4+1 clustering; (iv) each variable in a separate cluster. The results for one random starting point are shown on Figure 1(b). As we can see, the impact of (even

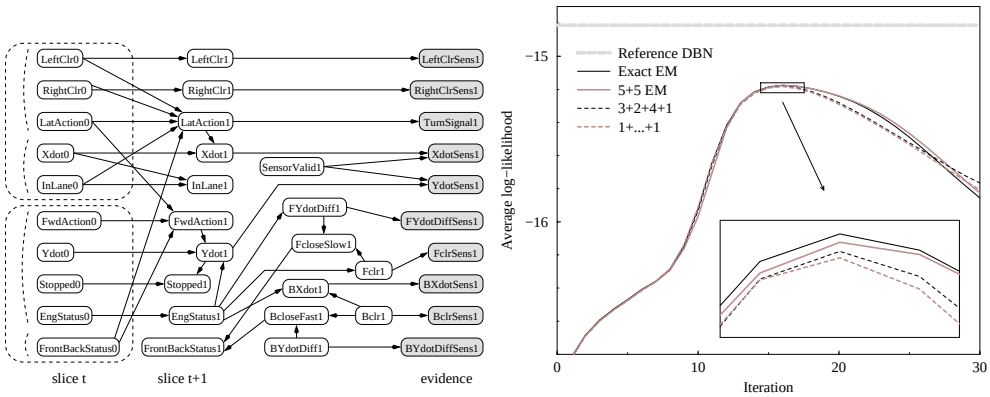

Figure 1: (a) The BAT DBN. (b) Structural approximations for batch EM.

severe) structural approximation on learning accuracy is negligible. For the other starting points, the approximate algorithms also tracked the exact algorithm very closely. In all of the runs, the difference in the peak log-likelihood reached by the different algorithms was at most 0.04. This phenomenon is rather remarkable, especially in view of the substantial savings caused by the approximations: on a Sun Ultra II, the computational cost of learning was 46 min/iteration in the exact case, vs. 3 min/iteration for the 5+5 clustering, and less than 2 min/iteration for the other two.

4 On-line learning

Our analysis also gives us the tools to address another important problem with learning dynamic models: the need to reason about the entire temporal sequence at once. One consequence of our contraction result is that the effect of approximations done far away in the sequence decays exponentially with the time difference. In particular, the effect of an approximation which ignores observations that are far in the future is also limited. Therefore, if we do inference for a time slice based on a small window of observations into the future, the result should still be fairly accurate. More precisely, assume that we are at time t and are considering a window of size w. We can view the uniform message as a very bad approximation to $\boldsymbol{\beta}^{(t+w)}$. However, as we propagate this approximate backward message from $t+w$ to t, the error of the approximation will attenuate exponentially quickly. Thus, if $\tilde{\boldsymbol{\beta}}^{(t)}$ is the message used in this process to update $\boldsymbol{\psi}^{(t)}$, then $\boldsymbol{D}_{\text{Proj}}[\boldsymbol{\beta}^{(t)}, \tilde{\boldsymbol{\beta}}^{(t)}]$ will decrease exponentially with w.

Based on these insights, we experimented with various online algorithms that use a small window approximations. Our online algorithms are based on the approach of [11], in which ESS are updated with an exponential decay every few data cases; the parameters are then updated correspondingly. The main problem with frequent parameter updates in the online setting is that they require a recomputation of the messages computed using the old parameters. For long sequences, the computational cost of such a scheme would be prohibitive. In our algorithms, we simply leave the forward messages unchanged, under the assumption that the most recent time slices used parameters that are very close to the new ones, so that the current message is fairly similar to its updated value. Our contraction result tells us that the use of very old parameters far back in the sequence has a negligible effect on the message. We tried several schemes for the update of the backward messages. In the *dynamic-400* approach, we use a backward message computed over 400 slices, with the closer messages recomputed very frequently as the parameters are changed, based on

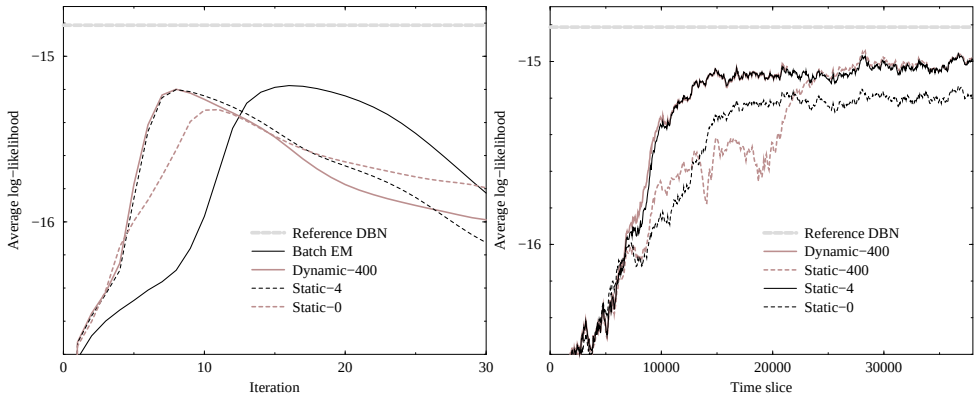

Figure 2: Temporal approximations for (a) batch setting; (b) online setting.

cached messages that used older parameters. The 8 closest messages are updated every parameter update, the next 16 every other update, etc. This approach is the closest realistic alternative to a full update of backward messages. In the *static-400* approach, we use a very long window (400 slices), but do not recompute messages; when the window ends, we use the current parameters to compute the messages for the entire next window. In the *static-4* approach, we do the same, but use a very short window of 4 slices. Finally, in the *static-0* approach, there is no lookahead at all; only the past and present evidence is used to compute the joint beliefs. The latter case is precisely the technique used in Kalman filters [9] for online learning of the process parameters. To minimize the computational burden, all tests were conducted using the 5+5 structural approximation. The running time for the various algorithms are: 0.4 sec/slice for batch EM; 1.4 for dynamic-400; 0.5 for static-400 and for static-4; and 0.3 for static-0.

We evaluated these temporal approximations both in an on-line and in a batch setting. In the batch experiments, we used the same 400-step sequence used above. The results, including results for batch EM, are shown in Figure 2(a). In terms of the accuracy at the peak, we can see that the static-4 method is almost equivalent to both batch EM and to the dynamic-400 method. The accuracy of the static-0 approach is somewhat lower. All methods overfit fairly badly, due to the small amount of training data. In terms of running time, the static-0 approximation reaches its peak in about half the time needed by static-4. Among the algorithms that reach the highest accuracy, static-4 is by far the fastest.

The second set of experiments focused on a real online learning task, using a single long sequence of 40,000 slices. The results are displayed on Figure 2(b). The static-4 approach is almost indistinguishable in terms of accuracy from the dynamic-400 approach. The static-0 approach converges more slowly, and does not seen to achieve the same accuracy as the other algorithms. The static-400 approach ultimately converges to the same accuracy as the dynamic-400 and static-4 approaches, but much more slowly. Our results show that even a very short window allows rapid convergence to the right answer, and that frequent updates over short windows are significantly better than infrequent updates over longer ones.

5 Conclusion and extensions

In this paper, we suggested the use of simple structural approximations in the inference algorithm used in an E-step. Our results show that even severe structural approximations have negligible effects on the accuracy of learning. The advantages of approximate inference in the learning setting are even more pronounced than in the case of pure inference [2];

here, the small errors caused by approximation are negligible compared to the larger ones induced by the learning itself. Our techniques provide the first feasible approach for learning structured models of complex dynamic systems, with the resulting advantages of generalization and the ability to incorporate prior knowledge. We also presented results for the online learning task, showing that even a very small time window suffices for doing learning.

The work most comparable to ours is the variational approach to approximate inference applied to learning factored HMMs [8]. While we have not done a direct empirical comparison, our algorithms track exact EM so closely, that any improvement in accuracy would be negligible. Our algorithm is also much simpler, and therefore may be faster. Most importantly, however, its applicability is much wider, as it can be used for DBN models whose structure is more complex than that of factored HMMs, and in that it can also be used for online learning.

The most obvious extension to our results is an integration of our ideas with structure learning algorithm for DBNs [7]. This integration is not completely trivial, as the structural approximation used in our algorithm is designed to produce the expected sufficient statistics required for learning parameters for a given structure. In a structure learning algorithm, the set of sufficient statistics required changes constantly; therefore, the structure of our approximation will also need to be more flexible. We believe that the resulting algorithm will be able to learn structured models for real-life complex systems.

References

[1] M. Artzrouni and X. Li. A note on the coefficient of ergodicity of a column-allowable nonnegative matrix. *Linear algebra and applications*, 214:93–101, 1995.

[2] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proc. UAI*, 1998. To appear.

[3] T. Cover and J. Thomas. *Elements of Information Theory*. Wiley, 1991.

[4] T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Comp. Int.*, 5(3), 1989.

[5] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B39:1–38, 1977.

[6] J. Forbes, T. Huang, K. Kanazawa, and S.J. Russell. The BATmobile: Towards a Bayesian automated taxi. In *Proc. IJCAI*, 1995.

[7] N. Friedman, K. Murphy, and S.J. Russell. Learning the structure of dynamic probabilistic networks. In *Proc. UAI*, 1998. To appear.

[8] Z. Ghahramani and M.I. Jordan. Factorial hidden Markov models. In *NIPS 8*, 1996.

[9] R.E. Kalman. A new approach to linear filtering and prediction problems. *J. of Basic Engineering*, 1960.

[10] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *J. Roy. Stat. Soc.*, B 50, 1988.

[11] R.M. Neal and G.E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In M.I. Jordan, editor, *Learning in Graphical Models*. Kluwer, 1998.

[12] L. Rabiner and B. Juang. An introduction to hidden Markov models. *IEEE Acoustics, Speech & Signal Processing*, 1986.

[13] G. Zweig and S.J. Russell. Speech recognition with dynamic bayesian networks. In *Proc. AAAI*, 1998. To appear.