Bayesian Clustering of Sensory Inputs by Dynamics

Paola Sebastiani
Statistics Department
The Open University
p.sebastiani@open.ac.uk

Marco Ramoni
Knowledge Media Institute
The Open University
m.ramoni@open.ac.uk

Paul Cohen
Department of Computer Science
University of Massachusetts, Amherst
cohen@cs.umass.edu

Abstract

This paper describes a Bayesian approach to the abstraction of sensor dynamics using a new clustering algorithm for time series to learn prototypical behaviors of a robot's sensory inputs. Each sensor stream reading is modeled as a Markov chain (MC). The abstraction process is performed by an unsupervised clustering algorithm returning the most probable set of clusters capturing the robot's sensory experiences. In order to increase efficiency, the algorithm uses an heuristic search strategy merging the closest MCs according to a measure of similarity based on entropy.

Introduction

We are developing methods whereby autonomous agent — a mobile robot — can learn about its actions and their effects in its environment. The methods in this paper accomplish two kinds of learning from experience; first, learning Markov chain (MC) representations of the dynamics in sensor time series, and second, clustering these time series by their dynamics to discover prototype experiences. For example, the robot has learned clusters that correspond to passing an object and moving toward an object. It is important to the goals of our project that the robot's learning should be *unsupervised*, which means we do not tell our algorithms — either explicitly or implicitly through a training signal — which MCs and clusters to learn. The robot learns those that are supported by its experience. It can use these chains and clusters to recognize its activities when it repeats them and to predict how the activities will unfold.

The techniques in this paper differ from other methods of finding Markov chain models and clusters (e.g., (Rosenstein & Cohen 1998; Rosenstein *et al.* 1997)) in being fundamentally Bayesian. A Bayesian approach is particularly well suited for these tasks because it frames the learning process as continuous updating rather than a batch analysis of data. Furthermore, a Bayesian approach provides a principled way to integrate prior and current evidence; prior knowledge representing past experience is updated, by processing current evidence, into posterior knowledge and this in turn will be the prior knowledge when future data are seeing. As the robot gains more experience, it requires proportionately more evidence to modify or discount its prior conclusions.

While these methods are general, they have been developed to implement learning in a Pioneer 1 mobile robot. The Pioneer 1 is a small platform with two drive wheels and a trailing caster, and a two degree of freedom paddle gripper. For sensors the Pioneer 1 has shaft encoders, stall sensors, five forward pointing and two side pointing sonars, bump sensors, a pair of IR sensors at the front and back of its gripper, and a simple vision system that reports the location and size of color-coded objects. Our configuration of the Pioneer 1 has roughly forty sensors, though the values returned by some are derived from others.

We show how MCs can represent the dynamics of sensory experiences. A MC represents a dynamic process as a transition probability matrix. For each experience the robot has, we construct one such matrix for each sensor. Each row in the matrix represents a state of the sensor, and the columns represent the probabilities of transition from that state to each other state of the sensor on the next time step. The result is a set of conditional probability distributions, one for each state of the sensor, that can be learnt from the past experiences of the agent. After m experiences, the robot has learned m transition matrices for each sensor. Next, a Bayesian clustering algorithm groups experiences that produce similar transition probability matrices. Each group is then characterized by its average or prototypical dynamics. The learned model of dynamics enables the agent to classify its current experience by computing the probability of an experience being in a particular cluster given sensor readings, and to predict future experiences, conditional on current input and cluster membership.

The rest of the paper is organized as follows. After reviewing background material on MCs, we describe a learning method to induce the transition probability matrix of a MC from sensor readings, and then describe the Bayesian clustering algorithm to sequentially merge streams that induce similar MCs.

Markov Chains

The dynamics of a sequence of sensory values can be modeled by a Markov Chain (MC). The sensor X is regarded as a random variable taking values $1, 2, ..., s$. The process generating the stream $x = (x_1, x_2, ..., x_{i-1}, x_i, ..)$ is a MC if $p(X = x_t|(x_1, x_2, ..., x_{t-1})) = p(X = x_t|x_{t-1})$ for any x_t in x. In words, the probability of the transition $x_{t-1} \rightarrow x_t$ is only a function of x_{t-1} or, by letting X_t be the variable representing the sensor values at time t, X_t is conditionally independent of $X_0, X_1, ..., X_{t-2}$ given X_{t-1}. The assumption of conditional independence allows us to represent a MC by a vector of probabilities $p_0 = (p_{01}, p_{02}, ..., p_{0s})$, denoting the distribution of X_0 (the initial state of the chain) and a matrix of transition probabilities:

$P = (p_{ij}) =$

X_{t-1} \ X_t	1	2	$\cdots$	s
1	p_{11}	p_{12}	$\cdots$	p_{1s}
2	p_{21}	p_{22}	$\cdots$	p_{2s}
$\vdots$		$\cdots$		
s	p_{s1}	p_{s2}	$\cdots$	p_{ss}

where $p_{ij} = p(X_t = j|X_{t-1} = i)$. By using the Chapman-Kolmogorov Equations (Ross 1996), the expected value of X_t is $p_0 P^t$ which, for increasing values of t, gives the average trajectory.

Discovering Markov Chains

During its interaction with the world, the robot records the values of about 40 sensors every 1/10 of a second. In an extended period of wandering around the laboratory, the robot will engage in several different activities — moving toward an object, losing sight of an object, bumping into something — and these activities will have different sensory signatures. Because we insist that the robot's learning is unsupervised, we do not tell the robot which activities it is engaging in, or even that it has switched from one activity to another. Instead we define a simple event marker — simultaneous change in three sensors — and we define an *episode* as the period between event markers. Each transition matrix is built from the data from one sensor for one episode. Then we cluster transition matrices with similar dynamics.

Learning A Markov Chain From Sensor Readings

Suppose the robot has generated a stream of the sensor X for one episode. The transition probabilities (p_{ij}) are unknown parameters to be inferred. The sensor stream can be summarized into an $s \times s$ contingency table with the frequencies of transitions $n_{ij} = n(X_{t-1} = i \rightarrow X_t = j)$:

$N =$

X_{t-1} \ X_t	1	2	$\cdots$	s	Tot
1	n_{11}	n_{12}	$\cdots$	n_{1s}	n_1
2	n_{21}	n_{22}	$\cdots$	n_{2s}	n_2
$\vdots$		$\cdots$			
s	n_{s1}	n_{s2}	$\cdots$	n_{ss}	n_s

These counts are used to estimate the parameters p_{ij}.

An intuitive way to estimate p_{ij} is to use the relative frequencies of transitions, so that the probability of the transition $X_{t-1} = i \rightarrow X_t = j$, that we will denote as $i \rightarrow j$, is just the ratio between the number of times the transition has been observed and all observations on the variable in state i: $p_{ij} = n_{ij}/n_i$. However, this method estimates the transition probability p_{ij} as 0 whenever $n_{ij} = 0$. Thus, when the chain is observed over a relatively short time interval, or a transition probability is small, it is very easy to conclude that some transition is impossible. A Bayesian estimation of the transition probabilities overcomes this problem, and uses any prior knowledge about the process. The unknown probabilities p_{ij} are regarded as random variables themselves. Before data collection, knowledge of the stochastic process (gathered, for instance, from past experience) is used to elicit a *prior* distribution for $p(p_{ij})$. The information conveyed by the current episode is then used to update the prior distribution of $p(p_{ij})$, providing a *posterior* distribution via Bayes' theorem:

$$p(p_{ij}|N) = \frac{p(p_{ij})p(N|p_{ij}N)}{p(N)}$$

where $p(N|p_{ij})$ is the joint probability of the data N, and $p(N)$ is the marginal probability. Once the posterior distribution is found, the estimate of the unknown probabilities (p_{ij}) are computed as the expected probability value given the data.

$$\hat{p}_{ij} = E(p_{ij}|N) = \int p_{ij} p(p_{ij}|N) dp_{ij}.$$

This value is called the *posterior expectation.* (Ramoni & Sebastiani 1999) provide a more detailed description of the rationale of this operation.

For estimation purposes,it is convenient to represent the chain as a Bayesian Belief Network (BBN) in which the variable at time $t-1$, say X_{t-1}, is a parent of the variable X_t at time t (Heckerman, Geiger, & Chickering 1995). Thus, the learning problem is that of estimating the conditional probability table of $X_t|X_{t-1}$ and the solution is well known (Spiegelhalter & Lauritzen 1990). The basic idea is to assume a *conjugate* prior distributions for (p_{ij}) that is a product of s independent Dirichlet distributions with hyper-parameters $(\alpha_{i1}, ..., \alpha_{is})$ $(\alpha_{ij} > -1)$. We use the notation $D(\alpha_{i1}, ..., \alpha_{is})$ to denote a Dirichlet distribution whose density function is proportional to $\prod_j p_{ij}^{\alpha_{ij}}$, and the overall prior density is $\prod_{ij} p_{ij}^{\alpha_{ij}}$. When positive, the hyper-parameters represent the prior knowledge via a table P_c of counts of an imaginary sample of size α, where $\alpha = \sum_{ij} \alpha_{ij}$ is the *global* prior precision:

$$P_c = \begin{array}{c|cccc} & X_t & & & \\ X_{t-1} & 1 & 2 & \cdots & s \\ \hline 1 & \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1s} \\ 2 & \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2s} \\ \vdots & & \cdots & & \\ s & \alpha_{s1} & \alpha_{s2} & \cdots & \alpha_{ss} \end{array}$$

and the prior probability of the transition $i \to j$ is the $j-th$ prior mean $(\alpha_{ij}+1)/(\alpha_i+s)$ and $\alpha_i = \sum_j \alpha_{ij}$. The prior variance is $(\alpha_{ij}+1)(\alpha_i+s-\alpha_{ij})/[(\alpha_i+s)^2(\alpha_i+s+1)]$, which is a decreasing function of α_i when the prior means are fixed. Since small variance implies a large precision about the mean, α_i is the *local precision* about the conditional distribution $X_t|X_{t-1}=i$ and it shows the level of confidence about the prior specification. This family of distributions is rich enough to represent different levels of prior knowledge. When $\alpha_{ij}=0$ for all i,j, then the table of fictitious counts represents lack of prior knowledge, and the prior probabilities of transitions are all uniform.

If no data are missing, the posterior distribution of (p_{ij}) is still a product of independent Dirichlet distributions, with hyper-parameters given by summing up Pc and N and hence $\alpha_{ij}+n_{ij}$. Thus, the Bayesian estimate of the transition probability p_{ij} is the posterior mean

$$\begin{aligned} \hat{p}_{ij} &= \frac{\alpha_{ij}+n_{ij}+1}{\alpha_i+n_i+s} \\ &= \frac{\alpha_{ij}+1}{\alpha_i+s}\frac{\alpha_i+s}{\alpha_i+n_i+s} + \frac{n_{ij}}{n_i}\frac{n_i}{\alpha_i+n_i+s} \end{aligned}$$

and the estimate of the transition probability matrix is $\hat{P} = (\hat{p}_{ij})$. The estimate $\hat{p}_{ij}$ turns out to be a weighted average of the prior value $(\alpha_{ij}+1)/(\alpha_i+s)$ and the standard estimate n_{ij}/n_i, with weights that depend on the prior precision α_i and the sample size n_i. As the sample size n_i becomes large relative to α_i, the estimate p_{ij} will approach n_{ij}/n_i and the effect of the prior input is overcome by data. However, when α_i is large relative to n_i the effect of the prior input is dominating. Note also that the posterior variance of p_{ij} is $(\alpha_{ij}+n_{ij}+1)(\alpha_i+n_i+s-\alpha_{ij}-n_{ij})/[(\alpha_i+n_i+s)^2(\alpha_i+n_i+s+1)]$ that is a decreasing function of the *posterior precision* α_i+n_i. Hence, the quantity α_i+n_i can be taken as a measure of the *confidence* in the estimates: the larger the sample size, the stronger the confidence in the estimate.

The result shows another interpretation of the hyper-parameters. The quantity $\alpha_{ij}+1$ gives an *adjustment* to the probability that would be computed as observed relative frequency. In particular, one effect is that the estimate of a transition probability will not be zero if, *a priori*, the transition is not believed to be impossible.

Example. The table below reports the frequencies of transition observed in a stream of 296 readings for the sensor `vis-a-x`, which represents the horizontal location of an object in the visual field. The sensor returns continuous values in the range -140, 140. We discretized these values into 5 equally spaced bins labeled 1 to 5.

	1	2	3	4	5
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	228	2	0
4	0	0	1	50	2
5	0	0	0	1	11

With a prior global precision $\alpha = 0$ (corresponding to uniform prior probabilities), the learned transition matrix is:

$\hat{P}=$

	1	2	3	4	5
1	0.20	0.20	0.20	0.20	0.20
2	0.20	0.20	0.20	0.20	0.20
3	0.00	0.00	0.99	0.01	0.00
4	0.00	0.00	0.02	0.93	0.05
5	0.02	0.02	0.02	0.09	0.86

This matrix represents (to those of us familiar with the robot and its activities) an episode in which an object was in the visual field but not near the robot (the values 3, 4 and 5 represent the range -28,140.) The high confidence on the distributions of transitions

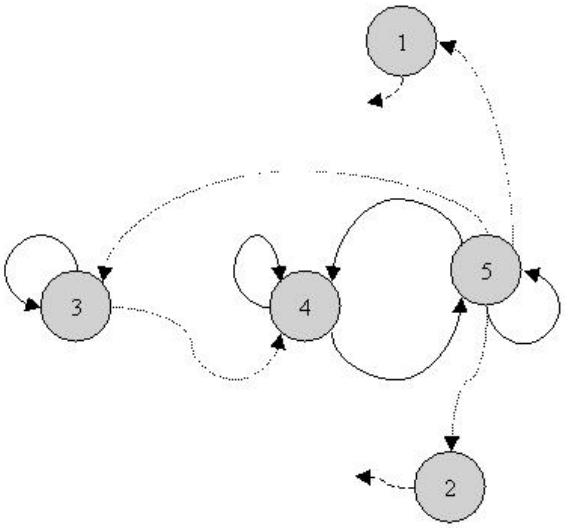

Figure 1: Markov Chain induced from data. Dotted lines represent rare transitions and dashed lines unknown transitions.

from state 3 and 4 (respectively 230 and 53 derived from the sample sizes n_3 and n_4) essentially rules out the possibility that either states 1 or 2 can be reached from 3 and 4. However, the small number of transitions observed from state 5 ($n_5 = 12$) does not rule out the possibility of transitions from 5 to either 1, 2 or 3 and the lack of information about transitions from states 1 and 2 results in these transitions getting uniform probabilities with a large uncertainty.

A summary of the induced MC is in Figure 1 in which dotted paths represent rare transitions and the dashed paths from states 1 and 2 represent unknown transitions.

Clustering

The second step of the learning process is to cluster MCs based on their dynamics. The available data is a set $S = \{S_i\}$ of m episodes (not necessarily of the same length) for each sensor sensor X. As we saw in the previous section, the data from one sensor and one episode can produce a MC. The challenge is to identify $c < m$ clusters for each sensor, or said differently, to group the m MCs for each sensor into c clusters that have similar dynamics and represent c different activities.

Suppose, initially, to know that the robot was engaged in only c different activities, some of which were repeated, and suppose we can identify those activities

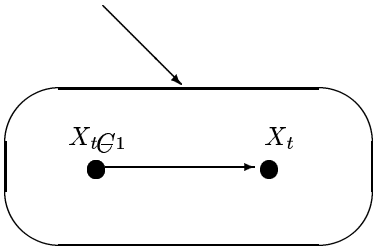

Figure 2: Clusters of several Markov Chains. The categorical variable C represents the cluster membership.

by attaching, to each episode, the value of a dummy variable C — taking c values — that represents the activity. This is equivalent to assuming that the process generating the data, i.e. the m sequences, is a directed graphical model M_c, as that shown in Figure 2, in which the variable C denotes the cluster membership and specifies a transition probability matrix for each activity.

The model M_c is specified by the distribution of C, and the conditional probability tables of $X_t|X_{t-1}, C = k$, one for each value of C. For example, if C has only two categories, and hence there are two clusters corresponding to two activities, the conditional probability table of $X_t|X_{t-1}, C = k$ is

		X_t			
C	X_{t-1}	1	2	$\cdots$	s
1	1	p_{111}	p_{112}	$\cdots$	p_{11s}
	2	p_{121}	p_{122}	$\cdots$	p_{12s}
	$\vdots$		$\cdots$		
	s	p_{1s1}	p_{1s2}	$\cdots$	p_{1ss}
2	1	p_{211}	p_{212}	$\cdots$	p_{21s}
	2	p_{221}	p_{222}	$\cdots$	p_{22s}
	$\vdots$		$\cdots$		
	s	p_{2s1}	p_{2s2}	$\cdots$	p_{2ss}

The top-half matrix $P_1 = (p_{1ij})$ contains the transition probabilities $p(X_t = j|X_{t-1} = i, C = 1)$ for the MC in cluster 1, while the bottom-half matrix $P_2 = (p_{2ij})$ contains the transition probabilities $p(X_t = j|X_{t-1} = i, C = 2)$ for the MC in cluster 2. The variable C is defined by its probability distribution which, in this example, is specified by $p(C = 1)$ and $p(C = 2) = 1 - p(C = 1)$. As the number of categories of C increases, there will be an increasing number of MCs, identified by the conditional probabil-

ity tables $P_k = (p_{kij})$ containing the transitions probabilities $p(X_t = j|X_{t-1} = i, C = k)$, and the probability distribution of C will be specified by $p_k = p(C = k)$ $(k = 1, ..., c)$. The probabilities (p_{kij}, p_k) are the unknown parameters to be induced from data. We can then collect the frequencies of transitions $i \rightarrow j$ for each cluster k, say n_{kij}, into the $c \times s \times s$ contingency table shown in Table 1 and use this data to estimate the transition probability matrix, for each cluster, as we did in the previous section.

However, in the original data set, the variable C is not observed because we do not tell the algorithm the clusters to learn. The goal is to let the robot discover the number of different activities it was engaged with — hence the number of categories of C — and to map each episode to one of these activities. Technically, this is equivalent to discovering the variable C and hence the model M_c that originated the data. The novelty of our approach is to regard this as a model selection problem in a Bayesian framework. The key idea is to find the model that makes the observed data most likely and it works as follows.

The number of states of the variable C is unknown, but the fact that there are initially m episodes imposes a bound, since C can have a number of categories that is any integer c between 1 and m. For any c, there is then the problem of mapping each episode to one of the c categories to produce c clusters. Each of these combination — number of categories of C and mapping — identifies a model M_c. If we could explore the set of all models, we would evaluate the posterior probability of each model and then select the one with the largest posterior probability. Let $p(M_c)$ be the prior probability of M_c. By Bayes' Theorem, the posterior probability of M_c, given the sample S is

$$p(M_c|S) = \frac{p(M_c)p(S|M_c)}{p(S)}.$$

The quantity $p(S)$ is the marginal probability of the data, and since $p(S)$ is constant for every model, in the comparison between different models it is sufficient to consider $p(M_c)p(S|M_c)$. In particular, if, *a priori*, all models are equally likely, the comparison can be based on the *marginal likelihood* $p(S|M_c)$, which is a measure of how likely the data are if the model M_c is true. The quantity $p(S|M_c)$ can be computed from the dependencies specified by the model M_c, and these are the marginal distribution of C, i.e., (p_k), and the conditional distribution of $X_t|X_{t-1} = i, C = k$, i.e., (p_{kij}). Formally, $p(S|M_c)$ is found as solution of the multiple integral

$$\int p(S|p_{kij}, p_k)p(p_{kij}, p_k)d(p_{kij})d(p_k)$$

		X_t				
C	X_{t-1}	1	2	$\cdots$	s	n_{ki}
1	1	n_{111}	n_{112}	$\cdots$	n_{11s}	n_{11}
	2	n_{121}	n_{122}	$\cdots$	n_{12s}	n_{12}
	$\vdots$		$\cdots$			
	s	n_{1s1}	n_{1s2}	$\cdots$	n_{1ss}	n_{1s}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
k	1	n_{k11}	n_{k12}	$\cdots$	n_{k1s}	n_{k1}
	2	n_{k21}	n_{k22}	$\cdots$	n_{k2s}	n_{k2}
	$\vdots$		$\cdots$			
	s	n_{ks1}	n_{ks2}	$\cdots$	n_{kss}	n_{ks}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	
c	1	n_{c11}	n_{c12}	$\cdots$	n_{c1s}	n_{c1}
	2	n_{c21}	n_{c22}	$\cdots$	n_{c2s}	n_{c2}
	$\vdots$		$\cdots$			
	s	n_{cs1}	n_{cs2}	$\cdots$	n_{css}	n_{cs}

Table 1: Transition probability matrices for model M_c.

where $p(S|p_{kij}, p_k)$ is the probability of the observed sample S, given M_c, and is a function of the unknown probabilities (p_{kij}, p_k), and $p(p_{kij}, p_k)$ is their prior distribution. Thus, we need to specify $p(p_{kij}, p_k)$ and $p(S|p_{kij}, p_k)$.

As in the previous section, the prior distribution for (p_{kij}) is a product of Dirichlet distributions, one for each row of Table 1. The prior distribution can be described in terms of an equivalent table of prior counts α_{kij} that represent the prior belief about the probability of transitions if this model were true. The prior distribution of (p_k) is conveniently represented by a Dirichlet distribution $D(\alpha_1, ..., \alpha_c)$, that is independent of the other distributions associated with the conditional probabilities (p_{kij}).

Conditional on M_c, data are summarized into the $c \times s \times s$ contingency table shown in Table 1. Let $n_{ki} = \sum_j n_{kij}$ be the number of transitions observed from state i in cluster k. Let also m_k be the number of episodes that are merged into cluster k. The observed frequencies (n_{kij}) and (m_k), are the data relevant to learning the probabilities (p_{kij}) and (p_k) respectively, and together with the prior hyper-parameters are all is needed to compute $p(S|M_c)$, which is given by two components

$$p(S|M_c) = p(S|C)p(S|X_t, X_{t-1}, C)$$

and

$$p(S|C) = \frac{\Gamma(\alpha)}{\Gamma(\alpha+m)} \prod_{k=1}^{c} \frac{\Gamma(\alpha_k + m_k)}{\Gamma(\alpha_k)}$$
$$p(S|X_t, X_{t-1}, C) = \prod_{i=1}^{s} \frac{\Gamma(\alpha_{ki} + sm_k)}{\Gamma(\alpha_{ki} + n_{ki} + sm_k)}$$
$$\times \prod_{j=1}^{s} \frac{\Gamma(\alpha_{kij} + n_{kij} + m_k)}{\Gamma(\alpha_{kij} + m_k)}$$

where $\Gamma(\cdot)$ denotes the Gamma function. This result can be derived by applying a standard technique as shown, for instance, in (Cooper & Herskovitz 1992). Note that, once a most probable model is found, the transition probability matrix in cluster k — obtained by merging m_k episodes — can be estimated with the same approach described in the previous section. Hence,

$$\hat{p}_{kij} = \frac{\alpha_{kij} + n_{kij} + m_k}{\alpha_{ki} + n_{ki} + sm_k}.$$

We also have the posterior distribution of the variable C that is Dirichlet, with updated parameters $\alpha_k + m_k$. These quantities can be in the robot's reasoning, as we will show in the next section. We conclude this section by suggesting a possible choice of the hyper-parameters. Since it seems infeasible to ask for prior hyper-parameters for all the models considered during the search process, we can use uniform prior distributions for all the transition probability matrices considered at the beginning of the search process. The initial $m \times s \times s$ hyper-parameters are set equal to $\alpha/(ms^2)$ and, when two MCs are similar and the corresponding observed frequencies of transitions are merged together, their hyper-parameters are summed up. Thus, the hyper-parameters of a cluster corresponding to the merging of m_k initial MCs will be $m_k\alpha/(ms^2)$. In this way, the specification of the prior distribution requires only the elicitation of the prior global precision α whose magnitude measures the degree of confidence in the prior model. Similarly, the hyper-parameters associated with the prior distribution of p_k can be chosen uniformly, by setting $\alpha_k = \alpha'/m$ initially and, when two episodes are merged, the corresponding hyper-parameters are summed up.

A Heuristic Search

Clearly, as m increases, the exhaustive search in the set of all possible models becomes intractable and a heuristic search strategy is required. The solution we propose is to use a measure of similarity between estimated transition probability matrices to guide the search process. The algorithm performs a bottom-up search by merging the closest MCs and evaluating if the resulting model is more probable of the model where these MCs are separated. When this is the case, the procedure replaces the two MCs with the cluster resulting by their merging and tries to cluster two other MCs. Otherwise, as a safety measure, the algorithm tries to merge the second best, the third best, and so on, until the set of pairs is empty.

For each sensor X_i, the algorithm applies the following procedure:

Input: A set S of sensor readings sequences.

Output: A set of transition matrices.

Initialization: Initialize as follows:

- MATRIX ESTIMATION: For each sequence $S_i \in S$, estimate the transition probability matrix $\hat{P}_i$ as described above and define the set $T_c = \{\hat{P}_i\}$ of all transition probability matrices.
- LIKELIHOOD ESTIMATION: Compute the marginal likelihood $p(S|M_c)$, where M_c represents the model in which each episode is generated by a different MC, and set $B = p(S|M_c)$. Note that, in this initial step, $c = m = |\mathcal{S}|$
- DISTANCE: Create the set $\mathcal{D}$ of the pairwise distances between each transition probability matrix in T_c according to some measure.
- SORT: Sort the set $\mathcal{D}$ in descending order of distance.

Iteration: Iterate until B does not increase any longer, then return T_c:

- CLUSTERING: Create the cluster C_k by merging the transition frequencies of the two closest transition probability matrices $\hat{P}_i$ and $\hat{P}_j$. Estimate the resulting transition probability matrix $\hat{P}_k$. Create the set T_c', by replacing $\hat{P}_i$ and $\hat{P}_j$ by $\hat{P}_k$. Create the set $\mathcal{D}'$ by inserting each distance between $\hat{P}_k$ and each other $\hat{P}_i$ in T_c in the ordered set $\mathcal{D}$ and by removing the distances from $\hat{P}_i$ and $\hat{P}_j$.
- LIKELIHOOD ESTIMATION: Compute the marginal likelihood $p(S|M_c)$, where M_c represents the model in which the episodes S_i and S_j are supposed to be generated by P_k.
- CLOSURE: If $p(S|M_c) > B$, set $B = p(S|M_c)$, replace T_c by T_c', $\mathcal{D}$ by $\mathcal{D}'$ and iterate. Otherwise, remove the first element of $\mathcal{D}$ and call the iteration on T_c.

The distance measure guiding the process can be any distance between probability distributions. Let P_1 and P_2 be matrices of transition probabilities of two MCs. Since they are both a collection of s probability distributions, and rows with the same index are probability distributions conditional on the same event, a measure

of similarity can be an average of the Kulback-Liebler distance (KL- distance) (or the cross entropy) between corresponding rows. Let p_{1ij} and p_{2ij} be the probabilities of the transition $i \rightarrow j$ in two MCs labeled with transition probability matrix P_1 and P_2. The KL-distance of these two probability distributions is

$$D_{kl}(p_{1i}, p_{2i}) = \sum_{j=1}^{s} p_{1ij} \log \frac{p_{1ij}}{p_{2ij}}.$$

The average distance between P_1 and P_2 is then $D(P_1, P_2) = \sum_i D_{kl}(p_{1i}, p_{2i})/s$. We have that $D(P_1, P_2) > 0$ and $D(P_1, P_2) = 0$ iff $P_1 = P_2$.

Prototypical Dynamics

In this section we report some results of applying the MC and cluster learning algorithms. In an experimental trial lasting about 30 minutes, the robot's activities were divided into 42 episodes by the criterion mentioned earlier: An episode ends when three or more sensors' values change simultaneously. In all, the data include 11,118 values for each sensor.

Our prior hyper-parameters are computed by distributing the global prior precision uniformly so that the only external input is the specification of the global prior precision α associated with the transition probability matrices, and α' associated with the variable C. A global prior precision $\alpha = 1050(1/210 - 1)$ ensures that, with 42 initial episodes, each transition probability matrix is based on prior counts $\alpha_{kij} = 1/210 - 1$ and the initial adjustment in the 42 transition probability tables is 1/210. This choice of α implies that, in the event of merging the 42 episodes into only one cluster, the adjustment to the estimates of the conditional probabilities is 1/5 corresponding to prior hyper-parameters $\alpha_{ij} = 1/5 - 1$ so that each conditional distribution would count on an initial local precision equal to -4. The global prior precision chosen for α' is 0, so that $\alpha_k = 0$ for all k and this determines an initial adjustment of 1 to the computation of the posterior distribution of C when the 42 episodes are assumed to be generated from different MCs.

The global prior precisions α and α' seem to have an effect on the number of clusters induced from the episodes. Increasing α results in increasing the number of clusters, while with small values of α the tendency is to merge all the episodes into one cluster. With $\alpha = 1050(1/210 - 1)$, the algorithm produces two clusters of dynamics for the sensor related to `vis-a-x`, the horizontal location of an object in the visual field. A summary is given in Figures 3 and 4, where we continue to represent rare transitions with dotted paths, and dashed paths are unknown transitions derived from not observing relevant cases in the streams.

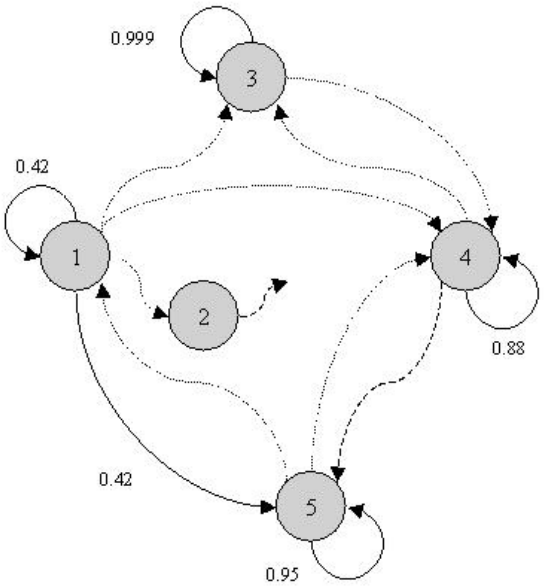

Figure 3: Markov Chain represented by the first cluster.

The first cluster represents the sensor dynamics when an object is not close to the robot. The transitions are limited to states 3, 4 and 5 that correspond to the range -28, 140. The initial state 1 can be reached from state 5, and this represents the fact that the object appears and disappears from the visual field. However, since the estimate of the probability of transitions $5 \rightarrow 1$ and $1 \rightarrow 5$ are derived from only two cases observed in all the episodes merged into cluster 1, the confidence in these estimate is very low. The second cluster, on the other hand, represents the sensor dynamics for an object not far from the robot, since transitions are essentially limited among the first 4 states. The prior specification does not rule out the possibility that either state 1 or 5 be reached from state 4. However, in the 12 episodes merged to create cluster 2, the transitions $4 \rightarrow 5$ and $4 \rightarrow 1$ were never observed, while state 4 was reached only once from state 3. A similar number of clusters was found for the other sensors taking 5 values. Sensors taking binary values produced a larger number of clusters.

This analysis can be extended to provide the robot with tools for recognizing the cluster it is in, given sensor data. Suppose the robot sensor related to `vis-a-x` records the new transition $1 \rightarrow 2$. It can infer cluster membership by applying Bayes theorem. The first cluster is obtained by merging 30 episodes. Since the

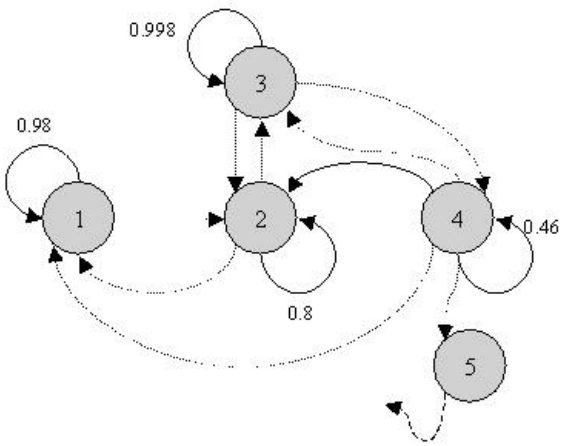

Figure 4: Markov Chain represented by the second cluster.

global precision adopted is $\alpha' = 0$, the posterior distribution of C turns out to be $D(30, 12)$, from which we estimate that, conditional on the data, the probability that $C = 1$ — i.e. that cluster membership is 1 and hence the object is not near the robot — is 0.7. Hence, the probability that $C = 2$ — i.e. that cluster membership is 2 and hence the object is not far from the robot — is 0.3. The probability of observing the transition $1 \rightarrow 2$ when $C = 1$ is 0.05, and becomes 0.013 when $C = 2$. A simple application of Bayes Theorem returns $p(C = 1|0.2 \rightarrow 0.4) = 0.90$ so that the robot is able to detect that, conditional on this new observed transition, it is more likely that it is in cluster 1.

Conclusions

This paper described a new approach to capture the dynamics of sensory inputs as a Bayesian unsupervised classification problem. The method uses MCs to capture the dynamic processes resulting from the interaction between the robot and its environment and then classifies these processes in prototypical experiences. In the examples of this paper, we limited our attention to first order MCs, in which each state in time is affected only by its immediate temporal predecessor. The representations of the sensory inputs obtained with this simple temporal model were enough to produce interesting results in the autonomous modeling process of the robot. However, it is worth noting that the method presented here is not limited to this order of MCs, but can be used to model and classify more complex temporal processes, involving dependencies at different time spans.

Acknowledgements

This research is supported by DARPA/AFOSR under contract(s) No(s) F49620-97-1-0485. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of DARPA/AFOSR or the U.S. Government.

References

Cooper, G., and Herskovitz, E. 1992. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning* 9:309–347.

Heckerman, D.; Geiger, D.; and Chickering, D. 1995. Learning Bayesian networks: The combinations of knowledge and statistical data. *Machine Learning* 20:197–243.

Ramoni, M., and Sebastiani, P. 1999. Bayesian methods in intelligent data analysis. In Berthold, M., and Hand, D., eds., *An Introduction to Intelligent Data Analysis*. New York: Springer. Also available as `http://kmi.open.ac.uk/techreports/KMI-TR-67`.

Rosenstein, M., and Cohen, P. R. 1998. Concepts from time series. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*. AAAI Press.

Rosenstein, M.; Cohen, P. R.; Schmill, M. D.; and Atkin, M. S. 1997. Action representation, prediction and concepts. University of Massachusetts Computer Science Department Technical Report 97-31, also presented at the 1997 AAAI Workshop on Robots, Softbots, Immobots: Theories of Action, Planning and Control.

Ross, S. 1996. *Stochastic Processes*. New York: Wiley.

Spiegelhalter, D., and Lauritzen, S. 1990. Sequential updating of conditional probabilities on directed graphical structures. *Networks* 20:157–224.