Unsupervised Classification of Sensory Inputs in a Mobile Robot

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

1. Introduction

Suppose one has a batch of univariate sequences generated by one or more unknown processes, and the processes have characteristic dynamics. The task of an unsupervised classifier consists of clustering these sequences into mutually exclusive classes, so that the elements of each class have similar dynamics. Suppose a batch contains a sequence of stride length for every episode in which a person moves on foot from one place to another. An unsupervised classifier might find classes corresponding to "ambling," "striding," "running," and "pushing a shopping cart," because the dynamics of stride length are different in these processes. Similarly, pathologies of the heart can be characterized by the patterns of sistolic and diastolic phases; dance steps, hand gestures and facial expressions can be characterized by the dynamics of movement of body parts [Johansson, 1973]; economic states such as recession can be characterized by the dynamics of economic indicators; syntactic categories can be categorized by the dynamics of word transitions [Charniak, 1993]; and so on.

The goal of this work is to enable mobile robots to learn the dynamics of their activities. If we regard the sequences of sensory inputs of the mobile robot as time series, we can represent these time series as Markov chain (MC) and then clusters these MCs by their dynamics to learn prototype experiences. For example, our robot has learned prototype experiences 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 algorithm which Markov chains, class and prototypes to learn.

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 learned from the past experiences of the agent. After k experiences, the robot has learned k 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 class of experiences given sensor readings, and to predict future experiences, conditional on current input and class membership.

A Bayesian approach is particularly well suited to clustering by dynamics 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. As our robot gains more experience (i.e., as its "prior" knowledge increases) it requires proportionately more evidence to modify or discount its prior conclusions.

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

2. The Robot Platform

The Pioneer 1 robot 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.

3. Clustering 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. For each episode in each sensor, we build a transition matrix and then we cluster transition matrices with similar dynamics.

3.1 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 sequence $x = (x_0, x_1, x_2, ..., x_{i-1}, x_i, ..)$ is a MC if $p(X = x_t|(x_0, x_1, x_2, ..., x_{t-1})) = p(X = x_t|x_{t-1})$ for any x_t in x. Let X_t be the variable representing the sensor values at time t, then 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:

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

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 sequence.

3.2 Learning A Markov Chain

Suppose the robot has generated a sequence of values from the sensor X for one episode. This sequences can be summarized into a $s \times s$ contingency table that contains the frequencies of transitions $n_{ij} = n(X_{t-1} = i \rightarrow X_t = j)$. These counts are used to estimate the transition probabilities p_{ij} characterizing the dynamic process that generated the data.

An intuitive way to estimate p_{ij} is to use the relative frequencies of transitions n_{ij}/n_i. In this way, the probability of the transition $X_{t-1} = i \rightarrow X_t = j$, that we will denote as $i \rightarrow j$, is estimated as the ratio between the number n_{ij} of times the transition has been observed and all observations on the variable in state i, that is, $n_i = \sum_j n_{ij}$. This estimate is a function of the data only and there may be other sources of information about the process. Furthermore, 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 p_{ij} overcomes this problem as well as using any prior knowledge about the process. This is achieved by augmenting the observed frequencies n_{ij} by *hyperparameters* α_{ij} that encode the prior knowledge about the process in terms of imaginary counts of a sample of size α. The *Bayesian estimate* of p_{ij} is

$$\hat{p}_{ij} = \frac{\alpha_{ij} + n_{ij}}{\alpha_i + n_i} \quad (1)$$

where $\alpha_i = \sum_j \alpha_{ij}$. By writing Equation 1 as

$$\hat{p}_{ij} = \frac{\alpha_{ij}}{\alpha_i}\frac{\alpha_i}{\alpha_i + n_i} + \frac{n_{ij}}{n_i}\frac{n_i}{\alpha_i + n_i} \quad (2)$$

we see that $\hat{p}_{ij}$ is an average of the estimate n_{ij}/n_i and of the quantity α_{ij}/α_i with weights that depend on α_i and the sample size n_i. Rewriting of Equation 1 as 2 shows that α_{ij}/α_i is the estimate of p_{ij} when the data set does not contain transitions from the state i — and hence $n_{ij} = 0$ for all j — and it is therefore called the *prior* estimate of p_{ij} while $\hat{p}_{ij}$ is called the *posterior estimate*. It can be shown that the variance of the prior estimate α_{ij}/α_i is given by $(\alpha_{ij}/\alpha_i)(1-\alpha_{ij}/\alpha_i)/(\alpha_i+1)$ and, for fixed α_{ij}/α_i, the variance is a decreasing function of α_i. Since small variance implies a large precision about the estimate, α_i will be called the *local precision* about the conditional distribution $X_t|X_{t-1} = i$ and it indicates the level of confidence about the prior specification. The quantity $\alpha = \sum_i \alpha_i$ is the *global* precision, as it accounts for the level of precision of all the s conditional distributions.

Equation 2 shows the trade-off between the intuitive estimate n_{ij}/n_i and prior estimate α_{ij}/α_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 that the variance of the posterior estimate p_{ij} is $\hat{p}_{ij}(1 - \hat{p}_{ij})/(\alpha_i + n_i + 1)$ and, for fixed $\hat{p}_{ij}$, it is a decreasing function of $\alpha_i + n_i$, the local precision augmented by the sample size n_i. Hence, the quantity $\alpha_i + n_i$ can be taken as a measure of the *confidence* in the estimates: the larger the sample size, the stronger the confidence in the estimate.

Example. The table below reports the frequencies of transition observed in an episode 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.

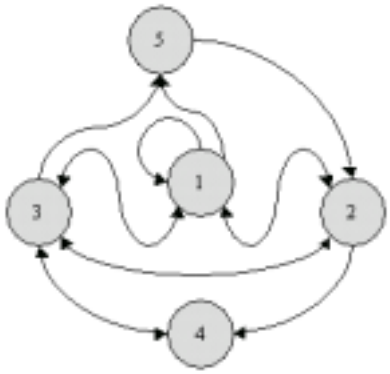

Figure 1: The Markov Chain used in the example.

	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 $\alpha = 25$ and a uniform prior probability distribution, 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 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.

3.3 Clustering

The second step of the learning process is an unsupervised agglomerative clustering of MCs on the basis of their dynamics. The available data is a set $S = \{S_i\}$ of m episodes (not necessarily of the same length) for each sensor and each episode is supposed to be generated by a MC. The task of the clustering algorithm is two-fold: find the set of classes that gives the best partition according to some measure, and assign each MC to one class. A partition is an assignment of MCs to classes such that each episode belongs to one and only one class.

The novelty of our approach is to regard the task of clustering MCs as a Bayesian model selection problem. In this framework, the model we are looking for is the most probable way of partition MCs according to their similarity *given* the data. We will use the posterior probability of a partition given the data as scoring metric to assess its goodness of fit and we will select the most probable model, that is, the model with maximum posterior probability given the data.

There is then the problem of mapping each episode to one class. We regard a partition as a hidden discrete variable C, where each state of C represents a class of MCs. The number c of states of C is unknown, but the number m of MCs to be clustered imposes an upper bound, as c will never exceed m. Each partition identifies a model M_c. 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. Since we are comparing all the models over the same data, $p(S)$ is constant and, for the purpose of maximizing $p(M_c|S)$, it is sufficient to consider $p(M_c)p(S|M_c)$. Furthermore, if all models are *a priori* 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 marginal distribution (p_k) of C and the conditional distribution (p_{kij}) of $X_t|X_{t-1} = i, C_k$ — where C_k represents the class membership of the transition matrix of $X_t|X_{t-1}$ — using a well-known Bayesian method [Cooper and Herskovitz, 1992]. Let n_{kij} be the observed frequencies of transitions $i \rightarrow j$ in class C_k, and let $n_{ki} = \sum_j n_{kij}$ be the number of transitions observed from state i in class C_k. We also define m_k to be the number of episodes that are merged into class C_k. The observed frequencies (n_{kij}) and (m_k) are the data required to learn the probabilities (p_{kij}) and (p_k) respectively and, together with the prior hyper-parameters α_{kij}, they are all is needed to compute $p(S|M_c)$ as

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

where

$$p(S|C) = \frac{\Gamma(\alpha)}{\Gamma(\alpha + m)} \prod_{k=1}^{c} \frac{\Gamma(\alpha_k + m_k)}{\Gamma(\alpha_k)}$$

and

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

where $\Gamma(\cdot)$ denotes the Gamma function. Once created, the transition probability matrix of a class C_k — obtained by merging m_k episodes — can be estimated as

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

In principle, we just need a search procedure over the set of possible partitions and the posterior probability of each partition as a scoring metric. However, the number of possible partitions grows exponentially with the number of MCs to be considered and, therefore, a heuristic method is required to make it feasible. 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 recursively merging the closest MCs (representing either a class or a single episode) and evaluating whether the resulting model is more probable than the model where these MCs are separate. When this is the case, the procedure replaces the two MCs with the cluster resulting from their merging and tries to cluster two other MCs. Otherwise, the algorithm tries to merge the second best, the third best, and so on, until the set of pairs is empty and, in this case, returns the most probable partition found so far. The rationale behind this ordering is that merging closer MCs first should result in better models and increase the posterior probability sooner. Note that the agglomerative nature of the clustering procedure spares us the further effort of assigning each single episode to a class because this assignment comes as a side effect of partitioning process.

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

Input: A set S of sensor readings episodes.

Output: A set of clusters and cluster assignments.

Initialization: Initialize as follows:

- MATRIX ESTIMATION: For each episode $S_i \in S$, 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: 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.

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

- CLUSTERING: Create the cluster C_k by summing the transition frequencies corresponding to 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 involving either $\hat{P}_i$ or $\hat{P}_j$.
- LIKELIHOOD: 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 iterate 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 between corresponding rows. Let p_{1ij} and p_{2ij} be the probabilities of the transition $i \to j$ in P_1 and P_2. The Kulback-Liebler distance of these two probability distributions is

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

The average distance between P_1 and P_2 is then $D(P_1, P_2) = \sum_i D(p_{1i}, p_{2i})/s$. Note that this distance becomes 0 when $P_1 = P_2$ and it is otherwise greater than zero.

We conclude this section by suggesting a choice of the hyper-parameters α_{kij}. 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 α_{kij} are set equal to $\alpha/(ms^2)$ and, when two MCs are similar and the corresponding observed frequencies of transitions are merged, 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 hyper-parameters requires only the prior global precision α, which measures the confidence in the prior model. An analogous procedure can be applied to the hyper-parameters α_k associated with the prior estimates of p_k. We will denote by α' the global prior precision associated to p_k.

4. Prototypical Dynamics

In an experimental trial lasting about 30 minutes, the robot's activities were divided into 42 episodes by the following criterion: An episode ends when three or more

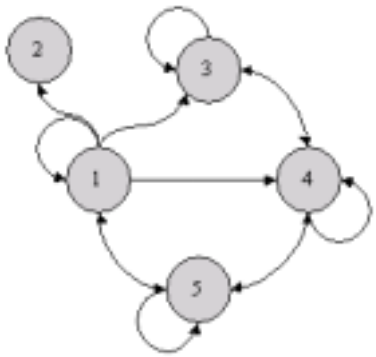

Figure 2: MC representing the first class.

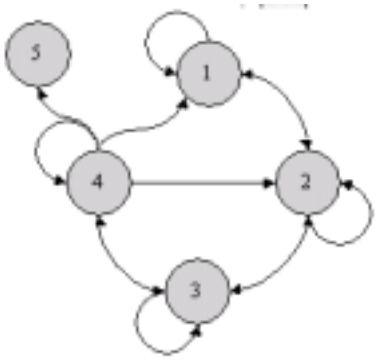

Figure 3: MC representing the second class.

sensors' values change simultaneously. The data include 11,118 values for each sensor. Our prior hyperparameters are computed by uniformly distributing the global prior precision, where $\alpha = 5$ and $\alpha' = 42$.

Figures 2 and 3 depict the MCs representing the two classes learned for the sensor `vis-a-x`, the horizontal location of an object in the visual field. The first class captures the sensor dynamics when the robot is not close to an object. 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, which 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 class 1, the confidence in these estimate is very low. The second class, 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 class 2, the transitions $4 \rightarrow 5$ and $4 \rightarrow 1$ were never observed, while state 4 was reached only once from state 3.

Our analysis can be extended to provide the robot with tools for recognizing the class 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 class membership by applying Bayes theorem. The first cluster is obtained by merging 30 episodes. Since the global precision adopted is $\alpha' = 42$, we can estimate that, conditional on the data, the probability of C_1 — i.e. that class membership is 1 and hence the object is not near the robot — is 0.7. Hence, the probability that $C = 2$ — i.e. that class 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 class 1.

Clustering by dynamics involves two levels of abstraction. First, the transitions in an episode are summarized in a transition matrix (MC); second, episodes are grouped into classes, and the MC representing a class is an averages of the constituent episode MCs. Both operations lose information. The *log-score* of a transition helps us evaluate these losses [Hand, 1997]. Let $s_{e,kij} = -\log \hat{p}_{kij}$ be the score of the transition $i \rightarrow j$ *observed* in an episode e. This score penalizes an episode MC by assigning large values to observed transitions when the MC assigns them small probabilities. We sum this score over all transitions in an episode, and sum again over all episodes, to get a score for the loss incurred by summarizing the time series of episode transitions into episode MCs. Now, instead of computing losses for episode e based on the episode MC, we can compute them based on the MC for the class to which e belongs. We let $s_{c,kij} = -\log \hat{p}_{kij}$ be the score assigned to the transition $i \rightarrow j$ observed in an episode that belongs to class C_k. As before, we sum $s_{c,kij}$ over the transitions within an episode, and sum again over episodes.

Finally, if episode e belongs to class C_k, we can ask how much predictive accuracy would be lost by using a *randomly selected* class to predict the episode transitions. This amounts to a test of whether classes retain any predictive power at all: if not, then a randomly selected class will incur the same losses as class C_k for episode e. The score $s_{r,kij} = -\log \hat{p}_{kij}$ is given to transition $i \rightarrow j$ *observed* in episode k but predicted by a random classification. Once again, these scores are summed over transitions in an episode and over episodes. We have now described three cumulative scores, s_e, s_c and s_r, which, for the dataset described earlier, have values $s_e = 120.5$, $s_c = 212.2$ and $s_r = 449.5$. The loss is least for episode MCs, intermediate for the MCs of our generated classifications and highest for randomly-select MCs. Apparently, the MC for episode e does a better job of predicting transitions in e than does the MC for class C_k to which e belongs, and both are better than using a randomly selected MC to make predictions.

Sign tests can tell us whether $s_e = 120.5$, $s_c = 212.2$ and $s_r = 449.5$ are significantly different. Let $s_{\delta,kij} = s_{e,kij} - s_{c,kij}$ be the difference in scores assigned to episode and cluster MCs for the transition $i \rightarrow j$ in episode k. Under the null hypothesis that the episode and cluster MCs make equally lossy predictions, half these differences should have a positive sign, half negative. In fact, 1567 differences are positive, 9950 are negative. We can compare predictions based on the generated classification with predictions based on randomly-selected MCs in the same way; 2799 differences are positive, 4445 are negative, and 3993 are zero. The sampling distribution for the number of positive differences is binomial and is well approximated by the normal distribution for this sample size. We find that episode MCs are significantly less lossy than the MCs representing the classes, which are themselves significantly less lossy than randomly-selected MCs.

5. Conclusions

This paper describes a new approach to discovering the dynamics of prototypical sensory experiences as a Bayesian unsupervised classification problem. The method represents the dynamic processes resulting from the interaction between the robot and its environment as MCs, and then groups these MCs into prototypical experiences. As described, the method uses first order MCs and univariate distributions but it can be easily extended to higher order MCs and multivariate distributions.

Acknowledgments

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. This research was developed while Paola Sebastiani and Marco Ramoni were visiting fellows at the Department of Computer Science, University of Massachusetts, Amherst.

References

[Charniak, 1993] Eugene Charniak. *Statistical Language Learning.* MIT Press, 1993.

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

[Hand, 1997] D.J. Hand. *Construction and Assessment of Classification Rules.* Wiley, New York, 1997.

[Johansson, 1973] Gunnar Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1973.

[Rabiner, 1989] L. R. Rabiner. A tutorial on Hidden Markov models and selected applications in speach recognition. *Proceedings of the IEEE*, 77:257–285, 1989.

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

[Ross, 1996] S.M. Ross. *Stochastic Processes.* Wiley, New York, 1996.

[Smyth, 1997] P. Smyth. Clustering sequences with hidden Markov models. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing*, volume 9, Cambridge, MA, 1997. MIT Press.