Unsupervised Clustering of Robot Activities: A Bayesian Approach

Paul Cohen, Marco Ramoni, Paola Sebastiani and John Warwick

University of Massachusetts in Amherst, MA, USA and The Open University, UK

Abstract

Our goal is for robots to learn conceptual systems sufficient for natural language and planning. The learning should be autonomous, without supervision. The first steps in building a conceptual system are to say some things are alike and others are different, based on how an agent interacts with them, and to organize similar things into classes or clusters. We use the BCD algorithm for clustering episodes experienced by our robots. The clusters contain episodes with similar dynamics, described by Markov chains.

Keywords: Adaptation and learning; Mobile agents; Autonomous robots.

1 Introduction

Picture yourself in a room illuminated only by a computer screen, onto which bit strings flash, two or three a second, for a few seconds at a stretch. After a few moments of this, the screen goes blank, until it starts up again with another sequence of bit strings. Your task is to make sense of the bit strings. Being human, you wonder what they mean. You try to work out how they might refer to your own experiences. You ask where they came from and what kind of process generated them. But your questions lead nowhere, so lacking a stronger model, you decide to analyze the bit strings as tokens generated by a very simple machine, a Markov chain. You treat each unique string as a token and tabulate the transition probabilities between them. And then you cluster the sequences of tokens into groups with similar transition probabilities. At the end of the day, you have your answer: Although the machine has generated several hundreds of sequences of bit strings, none identical, only five clusters emerge. Whatever process is generating the data, at some level of abstraction it seems to be doing only five qualitatively different things.

We have just described the world as experienced by a Pioneer 1 robot. As it does things in the laboratory — pushing a toy cup, passing another, picking up a block — its perceptual system produces propositions such as `(OBJECT A RED)` and `(APPROACH ROBOT A)`. By design, we know what these propositions mean but the robot does not. They might as well be structures of gensyms. Two or three times a second, the perceptual system produces a set of propositions to describe the current state. Every few seconds, the robot stops doing whatever it was doing and starts something else. In the current work, we mark these *episode boundaries* for the robot, although we are developing an algorithm to find episode boundaries automatically. Episodes, then, are time series representations, grounded in sensory data, of robot activities. The problem for our robot, and the focus of much of our research, is to learn enough about its activities and the objects in its environment to support planning and natural language dialog with humans [3, 15, 14, 13, 19, 22]. This paper describes an essential early step in the robot's conceptual development: Clustering episodes by their dynamics. Once the robot has identified clusters — once it knows that these episodes are similar and those are not — it can search for explanations, in particular, it can look for attributes of episodes that predict cluster membership.

It will be apparent from the previous example that we take the challenge of autonomy very seriously, but our robots will not learn much just by sitting in the lab with their power on. Some shaping is necessary, in the design of the robots' perceptual systems, in their activities, in the provision of algorithms — like the one we discuss here — for analyzing perceptual data. The "minimalist challenge" then is to show how *little* help an agent needs to learn about its world. Full autonomy clearly does not imply *tabula rasa* systems, but it does mean systems that learn primarily from interactions they have in their environments without supervision (i.e., without an omniscient teacher, oracle or critic). This paper describes one piece of the challenge, learning to group together similar activities.

More specifically, we let robots use a Bayesian algorithm for clustering episodes described by propositions produced by their perceptual system. In this work, dynamics are captured in first-order Markov chains (see [19, 14] for other approaches) and the clustering algorithms puts together episodes that are likely to be generated by the same process. The result is a partition of robots episodes into clusters describing different activities. Although a Markov chain is a very simple descriptions of a dynamic process, the algorithm, called Bayesian Clustering by Dynamics (BCD), has been applied successfully to cluster robot experiences based on sensory inputs [23, 25], simulated war games [24], as well as the behavior of stocks in market and the fugues of Bach. An intuitive explanation of the success of the algorithm is that describing a dynamic process as a Markov chain can be enough to capture the common dynamics of different robots episodes or, more generally, time series without need to resort to complex models as, for example, Hidden Markov models. Furthermore, an important novelty of the

BCD algorithm is its heuristic search that makes it very efficient.

After describing the algorithm, we will present the results of experiments with robots. We conclude this paper with a comparison of the BCD algorithm to other clustering techniques that capture other characteristics of dynamic processes.

2 Bayesian Clustering by Dynamics

The BCD algorithm is easily sketched: Given time series of tokens that represent states, construct a transition probability table for each series, then measure the similarity between each pair of tables to decide which tables try to cluster first, and finally group similar tables into clusters if their grouping increases a scoring metric. The clusters found by the BCD algorithm have the interesting property that they comprise a clustering with maximum posterior probability.

2.1 Estimating Markov Chains

Suppose we observe a time series $S = (x_0, x_1, x_2, ..., x_{i-1}, x_i, ..)$, where each x_i is one of the states $1, ..., s$ of a variable X. In the current work, x_i is a set of propositions generated by the robot's perceptual system at time i, such as `((MOVING-FORWARD R) (IS-RED A))`. The process generating the sequence S is a (first order) Markov chain if the conditional probability that the variable X visits state j at time t, given the sequence $(x_0, x_1, x_2, ..., x_{t-1})$, is only a function of the state visited at time $t-1$, [21]. Hence, we write $p(X_t = j|(x_0, x_1, x_2, ..., x_{t-1})) = p(X_t = j|x_{t-1})$, where X_t denotes the variable X at time t.

Markov chains can be represented as a probability distribution over the possible initial states of the chain and a table $P = (p_{ij})$ of transition probabilities, where $p_{ij} = p(X_t = j|X_{t-1} = i)$ is the probability of visiting state j given the current state i, so that

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

Given a time series generated from a Markov chain, we might estimate the probabilities of state transitions $X_t = j|X_{t-1} = i$ from the data as $p_{ij} = n_{ij}/n_i$, where $n_i = \sum_j n_{ij}$ and n_{ij} is the frequency of the transitions $X_t = j|X_{t-1} = i$ observed in the time series. Instead we prefer a Bayesian estimate in which prior information about transition probabilities can be taken into account. The derivation of this estimate is given in [23], here we simply give the result: The probability $\hat{p}_{ij}$ is estimated as

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

where $\alpha_i = \sum_j \alpha_{ij}$ and the so called prior hyper-parameter α_{ij} can be thought of as the prior frequency of the transition $X_t = j|X_{t-1} = i$, thus encoding prior knowledge about the process. This estimate is a posterior probability in the sense of being estimated from prior information α_{ij} about the transition $X_t = j|X_{t-1} = i$ and the observed frequency n_{ij} of the transition. Thus, α_i and n_i are the numbers of times the variable X visits state i in a process consisting of α and n transitions, respectively.

2.2 Clustering

The story so far has the robot engaging with its environment in episodes of a few seconds duration. Each episode is a time series S_i of sets of propositions, and each time series is transformed into a Markov chain as described above. Now, given the set of Markov chains, the BCD algorithm is ready to cluster the series in the set $S = (S_k)$. Clustering can be simply a matter of grouping objects together so that the average similarity of a pair of objects is high when they are in the same group and low when they are in different groups. Numerous clustering algorithms have been developed along this principle (see [6] for a survey).

The BCD algorithm is *agglomerative*, which means that, initially, there is one cluster for each Markov chain, then pairs of Markov chains are merged, iteratively. Merging two Markov chains yields another Markov chain until a stopping criterion is met. The BCD algorithm does not use a measure of similarity between Markov chains to decide if two Markov chains need to belong to the same cluster but it uses a different principle: Both the decision of whether grouping Markov chains and the stopping criterion in BCD are based on the posterior probability of the clustering, that is, the probability of the clustering conditional on the data observed. In other words, two Markov chains are merged in the same cluster if this operation increases the posterior probability of the clustering and the algorithm stops when the posterior probability of the clustering is maximum. If, as we do, the prior probabilities of the clusterings are assumed equal, then their posterior probabilities become proportional to a quantity called marginal likelihood.

In fact, BCD performs a hill-climbing search through the space of clusterings, so it yields a locally-maximum marginal likelihood clustering. A more precise term than "clustering" is "partition": A partition is a division of a set into mutually exclusive and exhaustive subsets. BCD's task is to find a maximum marginal likelihood partition of Markov chains. Said in yet another way, BCD solves a Bayesian model selection problem, where the model it seeks is the most probable partition of Markov chains given the data.

To solve the problem, BCD must know which Markov chains and clusters to agglomerate — it must know how to iteratively construct a partition — and it must be able to estimate the probability of the partition. We deal with these in turn.

The number of possible partitions grows exponentially with the number of Markov chains, so BCD cannot evaluate them all in its search for the most probable partition given the data. A heuristic method is required to make the search feasible. A good heuristic is merge or agglomerate similar Markov chains. What makes two Markov chains similar? Recall that each row of a transition probability table corresponds to a probability distribution over states at time t given a state at time $t-1$. Let P_1 and P_2 be tables of transition probabilities of two Markov chains. Because each table is a collection of s row conditional probability distributions, rows with the same index are probability distributions conditional on the same event. The measure of similarity that BCD uses is therefore an average of the Kullback-Liebler distances between row conditional distributions. Let p_{1ij} and p_{2ij} be the probabilities of the transition $X_t = j|X_{t-1} = i$ in P_1 and P_2. The Kulback-Liebler distance of the two probability distributions in row i is $D(p_{1i}, p_{2i}) = \sum_{j=1}^{s} p_{1ij} \log(p_{1ij}/p_{2ij})$. The average distance between P_1 and P_2 is then $D(P_1, P_2) = \sum_i D(p_{1i}, p_{2i})/s$.

Iteratively, BCD computes the set of pairwise distances between the transition probability tables, sorts the generated distances, merges the two closest Markov chains and evaluates the result. Note that the similarity measure is just used as a heuristic guide for the search process rather than a grouping criterion. The evaluation asks whether the resulting model M_c, in which two Markov chains are merged and replaced by the resulting Markov chain, is more probable than the model M_s in which these Markov chains are different, given the data S. If the probability $p(M_c|S)$ is larger than $p(M_s|S)$, BCD updates the set of Markov chains by replacing the two Markov chains with the cluster resulting from their merging. Then, BCD updates the set of ordered distances by removing all the ordered pairs involving the merged Markov chains, and by adding the distances between the new Markov chain and the remaining Markov chains in the set. The procedure repeats on the new set of Markov chains. If the probability $p(M_c|S)$ is not larger than $p(M_s|S)$, BCD 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 of this search is that merging similar Markov chains first should result in better models and increase the posterior probability sooner. Empirical evaluations of the methods in simulated data appear to support this intuition [17].

Now we will describe how BCD calculates the probability of a partition given the data, $p(M_c|S)$. Formally, this is done by regarding a partition as a hidden discrete variable C, where each state of C represents a cluster of Markov chains. The number c of states of C is unknown, but the number m of available Markov chains imposes an upper bound, as $c \leq m$. Each partition identifies a model M_c, and we denote by $p(M_c)$ its prior probability. 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 the variable C and the conditional distributions (p_{kij}) of $X_t|X_{t-1} = i, C_k$ — where C_k represents the cluster membership of the transition matrix of $X_t|X_{t-1}$ — using a well-known Bayesian method with conjugate Dirichlet priors [4, 23]. Let n_{kij} be the observed frequencies of transitions $X_t = j|X_{t-1} = i$ in cluster C_k, and let $n_{ki} = \sum_j n_{kij}$ be the number of transitions observed from state i in cluster C_k. We define m_k to be the number of time series that are merged into cluster 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 that is needed to compute the probability $p(S|M_c)$, which is the product of two components: $f(S, C)$ and $f(S, X_{t-1}, X_t, C)$.

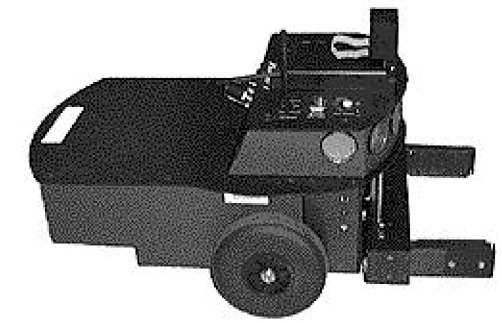
Figure 1: The Pioneer 1 robot.

Intuitively, the first quantity is the likelihood of the data, if we assume that we can partition the m Markov chains into c clusters, and it is computed as

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

The second quantity measures the likelihood of the data when, conditional on having c clusters, we uniquely assign each time series to a particular cluster. This quantity is given by

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

where $\Gamma(\cdot)$ denotes the Gamma function. Once created, the transition probability matrix of a cluster C_k — obtained by merging m_k time series — can be estimated as $\hat{p}_{kij} = (\alpha_{kij} + n_{kij})/(\alpha_{ki} + n_{ki})$.

We conclude this section by suggesting a choice of the hyper-parameters α_{kij}. We use uniform prior probabilities for all the transition 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 Markov chains 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 Markov chains 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 note that, since $\Gamma(x)$ is defined only for values greater than zero, the hyper-parameters α_{kij} must be non-negative.

Empirical evaluations have shown that the magnitude of the α value has the effect of zooming out differences between dynamics of different time series, so that, increasing the value of α yields an increasing number of clusters.

3 Clustering Robot Experiences

The robot in these experiments is a Pioneer 1 platform, depicted in Figure 1 with two independent drive wheels, a two degree-of-freedom paddle gripper, and several sensors. Visual sensors are provided by the Newton Labs Fast

RECEDE X Y	APPROACH X Y	
MOVING-BACKWARD R	MOVING-FORWARD R	
STOP R	FRONT-OF X Y	BEHIND X Y
LEFT-OF X Y	RIGHT-OF X Y	
IS-RED X	IS-ORANGE X	

Table 1: Example of primitive propositions describing the state of the world

Track vision system, which tracks colored regions and returns their location and size on the image plane. The robot also has seven sonars, shaft encoders for odometry on the drive wheels, and a bump sensor and IR sensors on the gripper. All told, the robot generates roughly 40 time series of real-valued data, sampled at 10Hz.

The question arises, should we try to cluster robot episodes by grouping together these multivariate sensory series, or should we first pass the sensor data to a perceptual system and cluster the perceptual experiences? We have done both. An earlier version of the BCD algorithm was tested with sensory data [25], and other algorithms for sensory data, based on dynamic time warping and delay coordinate embedding, have also been successful [14, 22, 19, 20]. In this work, we cluster time series of tokens returned by the perceptual system. One reason is that we overload the clustering algorithm when we force it to both resolve the ambiguity in sensory data and find similarities in time series.

The robot's perceptual system is a work in progress, but for our most recent experiments, it could describe the state of the world with up to 47 propositions at any instant. One example is given in Table 1 in which these propositions are obtained by binding objects in the robot's environment to the variables X and Y. Note that `R` is not a variable, it always denotes the robot.

Because up to 47 propositions may characterize the state of the world for the robot at each instant, the state space for the robot may be represented as a bit string of length 47. Each unique bit string represents a unique combination of propositions; for example, `LEFT-OF A B, IS-RED A, IS-ORANGE B` is one such string, in which three bits — corresponding to these propositions — have the value 1 and the rest have value 0. Although the state space for the robot is enormous, 2^{47}, in practice it encounters only a few of these states, and in this experiment encountered only 40. For example, in the first ten steps of an episode, one sees six unique states, described in Table 2.

Because the robot encountered only 40 unique states, BCD can represent each episode as a transition probability table of size 40×40. Notice that some states are physically impossible; for example, in the fifth state in Table 2, the robot is apparently receding from an object and stopped. The perceptual system is imperfect and has no "common sense" about the world, so it not infrequently constructs impossible state descriptions.

Episodes in the experiment were "set pieces" in which the robot executed a simple program in an environment controlled by us, such as moving forward past one object and bumping into another. Each episode lasted between two and eight seconds. Three replications, with different starting locations for the robot and objects in its environment, were run for each of the scenarios described in Table 3. Of course, the robot cannot group and differentiate these episodes based on what *we* call them, it must do so based on its perceptions during the episodes. BCD produced a partition of six clusters for these episodes that are displayed in Table 4. The numbers in parentheses refer to the replications of episodes; for instance, the first cluster contains all three replicates of `PUSH-C` and `APPROACH-C`, whereas the second cluster contains one of the three replicates of `PASS-RIGHT-C`. Clusters 3 and 4 each contain the replicates of just a single activity, whereas Clusters 5 and 6 contain several activities.

How should we evaluate this partition? Let us note, first, that it was produced by a single run of BCD, with no effort to tune the α parameter, or clean up the perceptual data, or to "help" BCD in any way. BCD did not produce 14 clusters (corresponding to the 14 scenarios in Table 3), each containing the three replicates of a single activity, but instead grouped some activities together. For example, Cluster 4 contains four activities in which the robot moved toward object A. Sometimes it stopped short of the object, sometimes it passed the object (on the left or right), and sometimes it pushed the object. It is not surprising, nor particularly disappointing, that BCD grouped these activities together, as they have similar dynamics: they all begin with approaching the object.

The disappointment is that BCD included `PASS-LEFT-C-THEN-PASS-RIGHT-A (2 3)` in Cluster 5, where they clearly do not belong. The story for Cluster 6 is similar: Three of the activities (and nine of the episodes) involve passing `A` on the left and then interacting with `C`, but one episode, `PASS-LEFT-C-THEN-PASS-RIGHT-A (1)` doesn't belong. The case can be made that the remaining two episodes, `PASS-RIGHT-C (1 2)`, have similar dynamics to the latter phase of `PASS-LEFT-A-THEN-PASS-RIGHT-C (1 2 3)`, so grouping them in Cluster 6 is not incorrect. As to Cluster 1, pushing C involves first approaching it, so grouping these activities together makes sense. Lastly, Cluster 2 is "pure" but for the inclusion of `PASS-RIGHT-C (1)`. In sum, we would have been happier had the activities in Clusters 5 and 6 not been grouped together, and we have identified four episodes (out of 42) that clearly do not belong in the clusters to which they were assigned, but on the whole, the partition above is satisfactory.

In a followup analysis, we ablated the state descriptions and re-ran BCD, to see how much the partition depended on particular propositions in the state descriptions. For exam-

```
((STOP R) (IS-RED A))
((STOP R) (IS-RED A))
((APPROACH A R) (STOP R) (IS-RED A))
((STOP R) (IS-RED A))
((RECEDE A R) (STOP R) (IS-RED A))
((IS-RED A))
((MOVING-FORWARD R) (IS-RED A))
((MOVING-FORWARD R) (IS-RED A))
((MOVING-FORWARD R) (IS-RED A))
((MOVING-FORWARD R) (IS-RED A))
```

Table 2: An example of propositions returned by the perceptual system of the robot.

APPROACH-A	APPROACH-C
PASS-RIGHT-A	PASS-RIGHT-C
PASS-RIGHT-A-THEN-PUSH-C	
PASS-RIGHT-C-THEN-PASS-RIGHT-A	
PASS-LEFT-A	PASS-LEFT-C
PASS-LEFT-A-THEN-PUSH-C	
PASS-LEFT-A-THEN-PASS-LEFT-C	
PASS-LEFT-A-THEN-PASS-RIGHT-C	
PASS-LEFT-C-THEN-PASS-RIGHT-A	
PUSH-A	PUSH-C

Table 3: Scenarios used in the experiment.

Cluster 1	PUSH-C (1 2 3) APPROACH-C (1 2 3)
Cluster 2	PASS-LEFT-C (1 2 3) PASS-RIGHT-C (1)
Cluster 3	PASS-RIGHT-A-THEN-PUSH-C (1 2 3)
Cluster 4	PASS-RIGHT-C-THEN-PASS-RIGHT-A (1 2 3)
Cluster 5	APPROACH-A (1 2 3) PASS-RIGHT-A (1 2 3) PASS-LEFT-A (1 2 3) PUSH-A (1 2 3) PASS-LEFT-C-THEN-PASS-RIGHT-A (2 3)
Cluster 6	PASS-LEFT-C-THEN-PASS-RIGHT-A (1) PASS-RIGHT-C (2 3) PASS-LEFT-A-THEN-PUSH-C (1 2 3) PASS-LEFT-A-THEN-PASS-LEFT-C (1 2 3) PASS-LEFT-A-THEN-PASS-RIGHT-C (1 2 3)

Table 4: Clusters produced by the BCD algorithm.

ple, we removed the propositions `IS-RED X` and `IS-ORANGE X` and re-wrote the affected state descriptions (so the state `((APPROACH R A)(IS-RED A))` becomes `(APPROACH R A)`). Similarly, we removed the propositions `MOVING-BACKWARD R` and `MOVING-FORWARD R`. In these analyses, BCD did not produce partitions of the episodes identical with the one above, but many of the clusters' substructures were maintained. For example, after removing propositions about color, BCD grouped together the elements in Clusters 1 and 3, above. It also formed a new cluster from `PASS-RIGHT-C (1 2 3)` and `PASS-LEFT-A (1 2 3)` episodes, which seems odd, although part of the explanation is that the robot identifies objects `A` and `C` by color, so with color terms gone, it confuses activities with objects `A` and `C`.

We have used BCD in other experiments with the robot, with similar results — it groups together instances of activities, and groups activities that share components such as approaching an object — and we have also used it to cluster simulated engagements in a wargame simulator [24], as well as time series of financial instruments, and Bach's fugues. In all these cases, we have been pleased with the results, but we recognize the need for more objective evaluation criteria. Generally it is difficult to say whether a partition of episodes is correct: Even if a gold standard partition exists for a given problem (and it doesn't in these experiments), we would need a metric that accounts for partially matching the gold standard. Had we a gold standard, we could use an approach, due to Schmill [14], which is to consider all possible pairs of episodes and ask whether the members of the pair were put in the same cluster or different clusters by BCD and the same cluster or different clusters by the gold standard. A different approach would be to assess whether a partition leads to good or poor results when it is used for some purpose, such as classification or another predictive task, using for example a decision theoretic approach or a scoring system. As yet, we have not put the robot's clusters to use in the life of the robot, but we shall.

4 Future and Related Work

We are currently implementing two extensions to BCD. First, the BCD algorithm clusters *univariate* time series, but a multivariate version is in the works. Suppose one has a multivariate time series of k variable each of which takes, say, v values. It's trivial to recode this series as one in which a single variable takes v^k values. The difficulty is computational: the transition probability table for the univariate case will have $(v^k)^2$ probabilities to estimate, and will tend to be very sparse unless the time series are quite long. In contrast, the transition probability tables for the k variables in the multivariate case could be quite dense, yielding good probability estimates, with shorter series. The second extension to BCD addresses a limitation we have had to impose on the autonomy of the robot. BCD clusters episodes, but we have to identify the episodes for the algorithm. Consequently, instead of letting the robot roam around the lab continuously, we have it execute one "set piece", then another, and call these episodes. One way to give the robot back some of the autonomy we stole from it would be to have the robot itself label episode boundaries; for example, as the robot switched from one plan to another it could flag the transition. Then BCD could work as it does now, but the episodes would be collected from continuous activity. The alternative we are pursuing is for BCD to find episode boundaries for itself. The intuition is simply that different episodes have different dynamics, so for every consecutive window of data BCD can ask whether it has the same or different dynamics than the previous window. BCD learns new Markov chains when none of its previous Markov chains explain a window of data well.

At first glance it may appear that BCD and Hidden Markov Models are similar technologies, and indeed we have used Hidden Markov Models for some robot learning tasks [5], but they are quite different. An Hidden Markov Model is a state machine with transition probabilities between states and each state has a probability distribution over the tokens it emits [16]. Hidden Markov Models are trained with time series (univariate or multivariate, continuous or categorical), which means the probability distributions within states and the transition probabilities between states are estimated. One must specify the number of states in advance although the algorithm in [5] dynamically splits Hidden Markov Model states in accordance with a minimum description length principle. Might one use Hidden Markov Model technology to find maximum likelihood partitions of time series, as BCD does? This problem would have to be transformed into one of finding the probabilities of emitting tokens and transition probabilities in a model of a specific number of states. An obvious choice would be to fit one Hidden Markov Model with n states to each episode with n unique states, then cluster the Hidden Markov Models, but it is unclear what advantage this holds over our current method of estimating Markov chains and clustering them. Another idea is to have each state represent a cluster of Markov chains. The difficulty is that, *within* an Hidden Markov Model state, one can only model the *marginal* probabilities of tokens, not the conditional probability of a token given the previous token. These conditional probabilities are modeled as state transition probabilities in Hidden Markov Models, which means that an episode must be modeled as a sequence of Hidden Markov Model states, not as a single state.

The Bayesian modeling-based approach used in BCD is similar to that used, for example, by Raftery [1, 7] to cluster static data. Recent work [18, 26] attempted to extend the idea to dynamic processes without, however, succeeding in finding a closed form solution as the one we have identified. For example, Ridgway [18] proposed using Markov Chain Monte Carlo methods, notorious for their slowness. Furthermore, an important novelty of our method is its heuristic search that makes the algorithm feasible. Methodology aside, BCD is similar in some respects to other algorithms for clustering time series developed in our lab. To assess the dissimilarity of a pair of multivariate, real-valued time series, Oates applies dynamic time warping to force one series to fit the other as well as possible; the residual lack of fit is a measure of dissimilarity, and with this, Oates can cluster episodes [14]. Rosenstein solves the problem by first detecting events in time series, then measuring the root mean squared difference between values in two series in a window around an event [19, 20]. It is worth noting that these methods and BCD handle time very differently. In Rosenstein's method, two time series are compared moment by moment for a fixed interval. In Oates's approach, one series is stretched and compressed within intervals to make it fit the other as well as possible. The former method keeps time rigid, the latter makes time elastic. If the duration of a sequence within a series is important to the identity of the series — if clustering should respect durations — the former method is probably preferrable to the latter. BCD is even more extreme because it transforms a time series, in which durations might be important, into a table of state transitions, which is inherently atemporal. For instance, one cannot tell by looking at a transition table whether a transition $X_t = j|X_{t-1} = i$ occurred before or after a transition $X_{t'} = j|X_{t'-1} = i$. We are beginning a study of the relative strengths and weaknesses of these methods on several kinds of time series.

5 Conclusion

Our goal is for robots to learn conceptual systems sufficient for natural language and planning, without supervision. By conceptual systems, we mean organizations of concepts that denote activities and objects in the robots' world; and knowledge about how activities unfold, and the roles that objects play in activities. With Lakoff and others [11, 10, 12, 9, 27, 8, 28, 2] we assert that the first step in building a conceptual system is to say some things are alike and others are different, based on how we interact with them, and to organize similar things into classes or clusters. BCD organizes the robot's activities into clusters. It searches heuristically for the organization having maximum probability, conditional on the data. The next step, currently underway, is to organize the objects in the robot's environment into clusters based on their roles in activities. Once we have clusters of activities and objects, we will apply standard classification algorithms such as `c4.5` to find the attributes of activities and objects that predict cluster membership.

Acknowledgments

This research is supported by DARPA/AFOSR under contracts Nos F49620-97-1-0485 and N66001-96-C-8504. 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

[1] J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821, 1993.

[2] J. A. Coelho and R. A. Grupen. A control basis for learning multifingered grasps. *Journal of Robotic Systems*, 14(7):545–557, 1997.

[3] P. R. Cohen. Dynamic maps as representations of verbs. In *Proceedings of the Thirteenth European Conference on Artificial Intelligence*, pages 145–149, 1998.

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

[5] L. Firoiu, T. Oates, and P. R. Cohen. Learning regular languages from positive evidence. In *Proceedings of the Twentieth Annual Meeting of the Cognitive Scien ce Society*, pages 350–355, 1998.

[6] D. Fisher. Conceptual clustering. In W. Klosgen and J. Zytkow, editors, *Handbook of Data Mining and Knowledge Discovery.* Oxford University Press, 1985.

[7] C. Fraley and A. E. Raftery. How many clusters? Which clustering methods? Answers via model-based cluster analysis. Technical Report 329, Department of Statistics, University of Washington, 1998.

[8] D. Gentner and A. B. Markman. Structural alignment in comparison: No difference without similarity. *Psychological Science*, 5:152–158, 1994.

[9] G. J. Gibbs and H. L. Colston. The cognitive psychological reality of image schemas and their transforms. *Cognitive Linguistics*, 6–4:347–378, 1995.

[10] M. Johnson. *The Body in the Mind.* University of Chicago Press, 1987.

[11] G. Lakoff. *Women, Fire, and Dangerous Things.* University of Chicago Press, 1984.

[12] J. M. Mandler. How to build a baby: II. Conceptual primitives. *Psychological Review*, 99(4):587–604, 1992.

[13] T. Oates and P. R. Cohen. Learning planning operators with conditional and probabilistic effects. In *Proceedings of the AAAI Spring Symposium on Planning with Incomplete Information for Robot Problems*, pages 86–94, 1996.

[14] T. Oates, M. D. Schmill, and P. R. Cohen. Identifying qualitatively different experiences: Experiments with a mobile robot. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, 1999.

[15] T. Oates P. R. Cohen, M. S. Atkin and C. R. Beal. NEO: Learning conceptual knowledge by sensorimotor interaction with an environment. In *Proceedings of the First International Conference on Autonomous Agents*, pages 170–177, 1997.

[16] L. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.

[17] M. Ramoni, P. Sebastiani, P. R. Cohen, J. Warwick, and J. Davis. Bayesian clustering by dynamics. Technical report, Knowledge Media Institute, The Open University, 1999.

[18] G. Ridgeway and S. Altschuler. Clustering finite discrete Markov Chains. In *Proceedings of the Joint Statistical Meetings, Section on Physical and Engineering Sciences.* ASA Press, 1998.

[19] M. Rosenstein and P. R. Cohen. Concepts from time series. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 739–745. AAAI Press, 1998.

[20] M. T. Rosenstein and P. R. Cohen. Continuous categories for a mobile robot. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 1999.

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

[22] M. D. Schmill, T. Oates, and P. R. Cohen. Learned models for continuous planning. In *Proceedings of Uncertainty 99: The Seventh International Workshop on Artificial Intelligence and Statistics*, pages 278–282, 1999.

[23] P. Sebastiani, M. Ramoni, and P. Cohen. Bayesian analysis of sensory inputs of a mobile robot. In *Proceedings of the 5th Workshop on Case Studies in Bayesian Statistics.* Springer-Verlag, 1999. To appear.

[24] P. Sebastiani, M. Ramoni, P. Cohen, J. Warwick, and J. Davis. Discovering dynamics using Bayesian clustering. In *Proceedings of the Third International Symposium on Intelligent Data Analysis*, pages 199–209. Springer, New York, NY, 1999.

[25] P. Sebastiani, M. Ramoni, and P. R. Cohen. Unsupervised classification of sensory input in a mobile robot. In *Proceedings of the IJCAI Workshop on Neural, Symbolic and Reinforcement Methods for Sequence Learning*, pages 23–28. 1999.

[26] P. Smyth. Probabilistic model-based clustering of multivariate and sequential data. In *Proceedings of Artificial Intelligence and Statistics 1999*, pages 299–304. Morgan Kaufman, San Mateo CA, 1999.

[27] E. S. Spelke, K. Breinlinger, J. Macomber, and K. Jacobsõn. Origins of knowledge.. *Psychological Review*, 99:605–632, 1992.

[28] E. Thelen and L. Smith. *A Dynamic Systems Approach to the Development of Cognition and Action.* MIT Press, Cambridge, MA, 1994.