HMM-Based Semantic Learning for a Mobile Robot

Kevin M. Squire, *Member, IEEE,* and Stephen E. Levinson, *Fellow, IEEE,*

***Abstract*— We are developing a intelligent robot and attempting to teach it language. While there are many aspects of this research, for the purposes here the most important are the following ideas. Language is primarily based on semantics, not syntax, which is still the focus in speech recognition research these days. To truly learn meaning, a language engine cannot simply be a computer program running on a desktop computer analyzing speech. It must be part of a more general, embodied intelligent system, one capable of using associative learning to form concepts from the perception of experiences in the world, and further capable of manipulating those concepts symbolically. This paper explores the use of hidden Markov models (HMMs) in this capacity. HMMs are capable of automatically learning and extracting the underlying structure of continuous-valued inputs and representing that structure in the states of the model. These states can then be treated as symbolic representations of the inputs. We show how a model consisting of a cascade of HMMs can be embedded in a small mobile robot and used to learn correlations among sensory inputs to create symbolic concepts, which will be used for decision making and eventually be manipulated linguistically.**

***Index Terms*— hidden Markov model, developmental robotics, hierarchical model, semantic learning, online learning.**

I. Introduction

COGNITIVE development has been studied in various environments—on the playground by the psychologist, under the microscope by the neuroscientist, and in the armchair by the philosopher. Our study occurs in a robotics lab, where we attempt to embody cognitive models in steel and silicon.

How did we choose this particular habitat? First and foremost, we are scientists and engineers, which immediately suggests forming theories and building things to test them. The particular question we are examining is one of the most fascinating questions that has been asked in the last century: Can machines think?

Alan Turing raised this very question back in 1950. He introduced the idea of a machine engaging in “pure thought” and communicating to the world via teletype writer. As an answer to The Question, he suggested that when the machine’s discourse (via teletype) was indistinguishable from a human’s, we could say that the machine was thinking. He goes on at the end of the paper to suggest that initially, machines could perhaps learn to compete with men at some purely intellectual task, such as chess, but then, he suddenly presents an alternative approach for creating machine intelligence:

> It can also be maintained that it is best to provide the machine with the best sense organs that money can buy, and then teach it to understand and speak English. This process could follow the normal teaching of a child. Things would be pointed out and named, etc. [1, p. 460]

Most artificial intelligence research has followed the former proposal. We believe the latter method holds more promise.

As scientists, we start with a hypothesis. Our hypothesis forms a constructive theory of mind, and can be summarized as follows. We believe that human intelligence, and hence language, is primarily semantic. We believe that the mind forms semantic concepts through the association of events close together in time, or events or cues close together in space, or both. We further believe that an integrated sensory-motor system is necessary to ground these concepts and allow the mind to form a semantic representation of reality—there is no such thing as a disembodied mind.

To test our hypothesis, we are developing a robotic platform, complete with basic sensory-motor and computing capabilities. The sensory-motor components are functionally equivalent to their human or animal counterparts, and include binaural hearing, stereo vision, tactile sense, and basic proprioceptive control. On top of these components, our group is implementing various processing and learning models, with the intention of creating and aiding semantic understanding. Our goal is to produce a robot that will learn to understand and carry out simple tasks in response to natural language requests.

At this point in time, we have already developed a robust base system and conducted a number of experiments on the way to our goal of a language-learning robot. In particular, we have developed the basic hardware and software framework necessary for our work, have run numerous experiments to study ideas in learning, memory and behavior. The primary contribution of this paper is an associative semantic memory based on hidden Markov models (HMMs) and built as part of the robot’s cognitive system.

Various researchers [2]–[15] study aspects of cognition using robotics; see [16] for a recent survey. One key aspect of our project different from most other work is our focus on language learning and interaction as a basis for higher level learning.

A. Robot Design and Construction

The flow diagram depicted in Fig. 1 has guided the design of both the physical and cognitive aspects of our robots. This diagram shows the flow of cognition through the body (somatic system), mind (noetic system), and outside world.

The sensory and motor systems form the basis of the somatic system–the body. As mentioned above, we believe that interaction with the world through a sensory-motor system is a prerequisite for cognition. Ideally, we would like our robot to perceive the world using a human-like set of senses, and have the ability to explore and manipulate its environment.

Manuscript received Somemonth 00, 2005; revised Anothermonth 32, 2005.

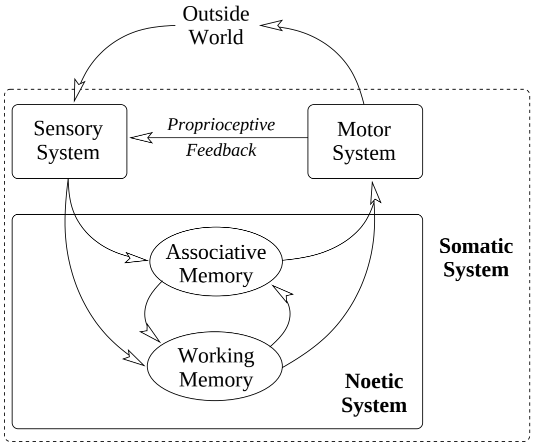


Fig. 1. Flow of Cognition. The design of our robotic system was guided by this diagram.

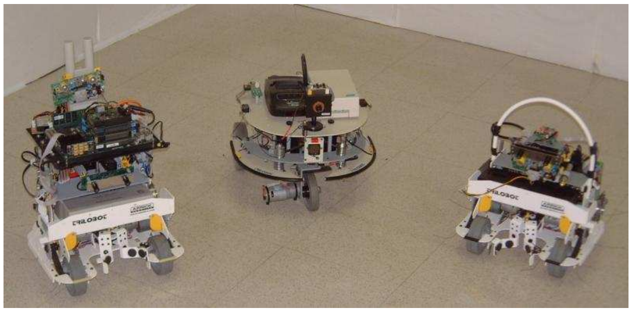


Fig. 2. Robots: Illy, Alan, and Norbert. Alan is an older model Trilobot.

As suggested by Fig. 1, cognition has a physical component (i.e., the brain is part of the somatic system), but its function, although constrained by the underlying physical component, is often considered separately. Taking a cue from psychologists, we model this functional aspect of cognition as an associative memory for long-term storage, and a working memory for immediate information and decision making.

To realize this system, we chose to work with Arrick Robotics' Trilobot, whose basic anthropomorphic capabilities are complete enough to suit our purposes. In particular, the robot can move freely over level surfaces, can move its head, and can use its arm to manipulate common objects, allowing relatively complex behaviors. For sensory input, the robot has a number of touch and other sensors available. We have added cameras and microphones for stereo vision and hearing, and an on-board computer processing and wireless networking for distributed processing. For this processing, we have developed a robust distributed computing framework to manage the actual data transmission and processing, with various modules for sensory data processing, learning, decision making, and control.

We currently have three such robots, whom we have named *Alan*, *Illy* and *Norbert* (see Fig. 2). We are continuing to develop and refine them.

B. Associative Learning and Memory

Through evolution and in our early childhood development, we first learn to understand the world by associating sensory-motor events and cues [17]. Some examples include learning what happens when one touches a hot burner, learning to associate the sight and smell of fire, and learning to associate a word with an event or some other co-occurring cue. Here, we will describe a model of associative semantic learning.

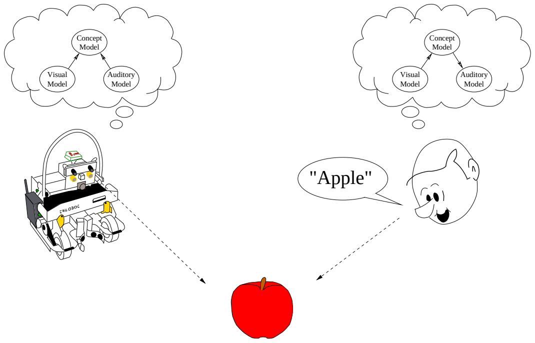


Fig. 3. Associative learning scenario. The robot is attempting to learn the boy's model of the world.

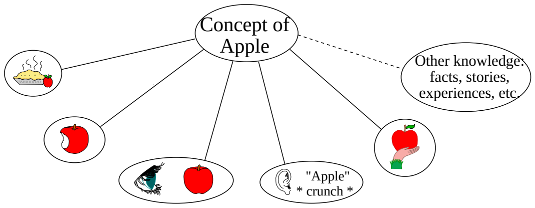


Fig. 4. The concept of *apple*, as a relationship among sensory inputs and other knowledge.

One necessary condition for effective communication between two people (or even a person and a robot) is that they share a similar model of the world—that they can experience the world in similar ways and understand a similar set of concepts. One way of posing the problem of robotic language acquisition, then, is that a robot should to learn a model of the world through verbal interactions with a human, as suggested by Fig. 3. This figure shows an interaction between two subjects, a boy and a robot, each with his own cognitive model of the world. The immediate goal of the robot is to learn the cognitive model the boy is using to comprehend the immediate environment. As the robot learns the boy's model of the world, it can use that model when making decisions.

For our purposes, we will assume that the boy's cognitive model contains the boy's semantic knowledge of the world, encoded as *concepts*. A recurring theme in our research is the idea that our understanding of the world is intimately related to our senses, and so, as suggested by Fig. 4, concepts such as *apple* are learned and recognized through corresponding sensory inputs and other related knowledge. In order to form concepts similar to the boy's, the robot must therefore learn associations among related sensory inputs from multiple senses.

Fig. 5 shows this idea in an abstract, simplified manner. Each submodel in this figure is a classifier. Classes in the visual model correspond to aspects of the different objects in its environment, including such things as colors, shapes, textures, or types of motion. Classes in the audio model represent

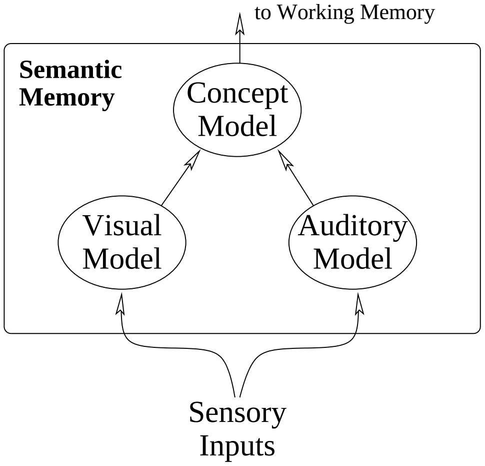


Fig. 5. Visual/auditory concept learning. The visual and auditory models form discrete representations (classifications) of corresponding sensory inputs, and the concept model learns common co-occurances of these representations.

unique audio cues, including speech. For the concept model, classes represent concepts and are formed from frequently co-occurring audio and visual inputs. Learning in all submodels is unsupervised.

As an example, suppose that the world contains apples, oranges, and bananas. As the robot explores, its visual model will be presented repeatedly with features from the various objects (such as shape and color information), and will form classes corresponding to the visual aspects of apples, oranges, and bananas. Additionally, auditory features from words or sounds corresponding to the objects will be presented to the auditory model, and it will form classes corresponding to commonly repeated sounds or words, such as “apple,” “round,” or (perhaps) non-speech related sounds. The classifications of the audio and visual inputs by these models will be presented to the concept model, which will form classes for commonly co-occuring inputs, corresponding to, e.g., shapes, colors, or object names. We have implemented the associative memory model presented here as a cascade of HMMs.

The rest of this paper proceeds as follows. Since they are the basis for our model, we introduce HMMs in Section II. We also include an online training algorithm for HMMs in this section. Section III describes our HMM-cascade model in abstract form, and Section IV describes how we use the model for concept learning, including simulation results. The actual implementation of the memory model on our robot required some design changes. We describe these changes and other practical matters in Section V, and give some experimental results from our robot runs in Section VI. Section VII gives some conclusions and future directions.

II. Hidden Markov Models

A. Description

An HMM is a discrete-time stochastic process with two components, $\{X_n, Y_n\}$, where (i) $\{X_n\}$ is a finite-state Markov chain, and (ii) given $\{X_n\}$, $\{Y_n\}$ is a sequence of conditionally independent random variables. The conditional distribution of Y_k depends on $\{X_n\}$ only through X_k. The name *hidden Markov model* arises from the assumption that $\{X_n\}$ is not observable, and so its statistics can only be ascertained from $\{Y_n\}$.

HMMs have many interesting features that can be exploited for concept learning. As noted previously, concepts are formed from the correlation in time among events. HMMs by construction have a notion of sequence, and have proven quite effective at learning time series and spatial models in such areas as speech processing [18] and computational biology [19]–[21]. This characteristic of HMMs provides a useful starting point for learning time-based correlations.

Another property of HMMs useful for learning concepts is their ability to discover structure in input data. Cave and Neuwirth [22] demonstrated this capability by training a low-order ergodic HMM on text. They found that the states of the model represented broad categories of letters, discovering some of the underlying structure of the text. Poritz [23] developed a similar model for speech data, and Ljolje and Levinson [24] created a speech recognizer based on this type of model. Our hierarchical model exploits this natural capability of HMMs to discover structure in order to learn higher level concepts.

Finally, in addition to their familiar role as recognizers, HMMs can be used in a generative capacity. In particular, when placed in a hierarchy, we can drive the various HMMs to produce sequences of states and corresponding output, roughly simulating sequences of thoughts.

Some characteristics of HMMs are not as useful for our work, however. Two of the most common methods used for HMM parameter estimation, the Baum-Welch method and methods based on the Viterbi algorithm, both require off-line processing of large amounts of data [18]. For our goal of learning concepts in real time using a robot, these methods are not very useful. We would much prefer an iterative or online training procedure.

There are generally two approaches researchers have used to implement online training for HMMs. The first approach minimizes the prediction error of the model via recursive methods [25]–[27]. The second approach is to recursively maximize the Kullback-Leibler information between the estimated model and true model, or equivalently, to recursively maximize the likelihood of the estimated model for an observation sequence [27]–[32]. The recursive maximum-likelihood estimation (RMLE) algorithm presented in Section II-C is based on [32].

B. Signal Model

An HMM is a discrete-time stochastic process with two components, $\{X_n, Y_n\}$, defined on probability space $(\Omega, \mathcal{F}, P)$. Let $\{X_n\}_{n=1}^{\infty}$ be a discrete-time first-order Markov chain with state space $R = \{1, \ldots, r\}$, r a fixed known constant. The model starts in a particular state $i = 1, \ldots, r$ with probability $\pi_i = P(X_1 = i)$. Define $\boldsymbol{\pi} \in \Pi$ by $\boldsymbol{\pi} = \{\pi_i\}$, where Π is the set of length-r stochastic vectors. For $i, j = 1, \ldots, r$, the transition probabilities of the Markov chain are given by

$$a_{ij} = P(X_n = j | X_{n-1} = i). \tag{1}$$

Let $\mathbf{A} = \{a_{ij}\}$. Then $\mathbf{A} \in \mathcal{A}$, where $\mathcal{A}$ is the set of all $r \times r$ stochastic matrices.

In an HMM, $\{X_n\}$ is not visible, and its statistics can only be ascertained from a corresponding observable stochastic process, $\{Y_n\}$. The process $\{Y_n\}$ is a probabilistic function of $\{X_n\}$, i.e., given X_n, Y_n takes values from some space E according to a conditional probability distribution. The corresponding conditional density of Y_n is generally assumed to belong to a parametric family of densities $\{b(\cdot;\theta) : \theta \in \Theta\}$, where the density parameter θ is a function of X_n, and Θ is the set of valid parameters for the particular conditional density assumed by the model. The conditional density of Y_n given $X_n = j$ can be written $b(\cdot;\theta_j)$, or simply $b_j(\cdot)$ when the explicit dependence on θ_j is understood.

Example 2.1: (Gaussian observation density): Suppose the observation density for each state in an HMM is described by a univariate Gaussian distribution. Then parameter set $\Theta = \{(\mu, \sigma) \in \mathbb{R} \times (0, \infty)\}$, $\theta_j \in \Theta$, and $\{Y_n\} = \{y_n\}$ is a sequence of continuously valued, conditionally independent outputs on $\mathbb{R}$, each with probability distribution

$$b(y_n;\theta_j) = b\left(y_n;\mu_j,\sigma_j\right) = \frac{1}{\sqrt{2\pi}\sigma_j}\exp\left[-\frac{(y_n-\mu_j)^2}{2\sigma_j^2}\right] \tag{2}$$

for $X_n = j$.

Example 2.2: (Finite-alphabet observation density): Suppose observations Y_n are drawn from a finite set of symbols $\mathcal{V} = \{v_k\}$, $k = 1, \ldots, s$. Then $\Theta = \{(b_1, \ldots, b_s) | \sum_{k=1}^{s} b_k = 1,\ b_k \geq 0\}$ is the set of length-s stochastic vectors, $\theta_j \in \Theta$, and $\{Y_n\} = \{y_n\}$ is a sequence of symbols draw from a finite alphabet, each y_n having probability

$$b(y_n;\theta_j) = b_{jk}|_{y_n = v_k} \tag{3}$$

for $X_n = j$.

For simplicity, the previous two examples and the following discussion assume Y_n to be scalar valued, although the formulation easily generalizes to vector-valued observations.

Define the HMM parameter space as $\Phi = \Pi \times \mathcal{A} \times \Theta$. The model $\boldsymbol{\varphi} \in \Phi$ is then defined as

$$\boldsymbol{\varphi} \;=\; \{\pi_1, \ldots, \pi_r, a_{11}, a_{12}, \ldots, a_{rr}, \theta_1, \ldots, \theta_r\}. \tag{4}$$

The model parameters for a particular model are accessed via coordinate projections, e.g., $a_{ij}(\boldsymbol{\varphi}) = a_{ij}$. In some cases, we will not be concerned with estimating $\boldsymbol{\pi}$. In that case, $\Phi = \mathcal{A} \times \Theta$, and $\boldsymbol{\varphi}$ changes accordingly. Note that the literature occasionally describes other model parameterizations (see, e.g., [26], [28]).

Let p be the length of $\boldsymbol{\varphi}$. When estimating model parameters, let $\boldsymbol{\varphi}^* \in \Phi$ be the fixed set of "true" parameters of the model we are trying to estimate.

For a vector or matrix $\mathbf{v}$, $\mathbf{v}'$ represents its transpose. Define the r-dimensional column vector $\mathbf{b}(y_n;\boldsymbol{\varphi})$ and $r \times r$ matrix $\mathbf{B}(y_n;\boldsymbol{\varphi})$ by

$$\mathbf{b}(y_n;\boldsymbol{\varphi}) = [b_1(y_n;\theta_1(\boldsymbol{\varphi})), ..., b_r(y_n;\theta_r(\boldsymbol{\varphi}))]' \tag{5}$$

and

$$\mathbf{B}(y_n;\boldsymbol{\varphi}) = \mathrm{diag}[b_1(y_n;\theta_1(\boldsymbol{\varphi})), ..., b_r(y_n;\theta_r(\boldsymbol{\varphi}))]. \tag{6}$$

Vector $\mathbf{b}(y_n;\boldsymbol{\varphi})$ and matrix $\mathbf{B}(y_n;\boldsymbol{\varphi})$ give the observation density evaluated at y_n for each state (in model $\boldsymbol{\varphi}$), as a vector and diagonal matrix, respectively.

Using the definitions above, it can be shown (see, e.g., [33]) that the likelihood of the sequence of observations $\langle y_1, \ldots, y_n\rangle$ for model $\boldsymbol{\varphi}$ is given by

$$p_n(y_1, \ldots, y_n;\boldsymbol{\varphi}) = \boldsymbol{\pi}(\boldsymbol{\varphi})'\mathbf{B}(y_1;\boldsymbol{\varphi})\prod_{k=2}^{n}\mathbf{A}(\boldsymbol{\varphi})\mathbf{B}(y_k;\boldsymbol{\varphi})\mathbf{1}_r, \tag{7}$$

where $\mathbf{1}_r$ refers to the r-length vector of ones.

C. Recursive Maximum-Likelihood Estimation of HMM Parameters

Maximum-likelihood estimation (MLE) is formally defined as follows. For observation sequence $\langle y_1, \ldots, y_n\rangle$, find

$$\hat{\boldsymbol{\varphi}} = \arg\max_{\boldsymbol{\varphi}\in\Phi} p_n(y_1, \ldots, y_n;\boldsymbol{\varphi}), \tag{8}$$

where $\hat{\boldsymbol{\varphi}}$ is the most likely estimate of the true underlying parameters $\boldsymbol{\varphi}^*$. The recursive maximum-likelihood estimation (RMLE) algorithm defined here is an iterative, stochastic gradient solution to this problem based on prediction, or forward filters.

For the results of this section to hold, it is necessary to assume various conditions on periodicity, continuity, and ergodicity for the model. See [32] for details.

Define the prediction filter $\mathbf{u}_n(\boldsymbol{\varphi})$ as

$$\mathbf{u}_n(\boldsymbol{\varphi}) = [u_{n1}(\boldsymbol{\varphi}), \ldots, u_{nr}(\boldsymbol{\varphi})]', \tag{9}$$

where

$$u_{ni}(\boldsymbol{\varphi}) = P(X_n = i | y_1, \ldots, y_{n-1}) \tag{10}$$

is the probability of being in state i at time n given all *previous* observations. Using this filter, the likelihood $p_n(y_1, \ldots, y_n;\boldsymbol{\varphi})$ can be written as

$$p_n(y_1, \ldots, y_n;\boldsymbol{\varphi}) = \prod_{k=1}^{n}\mathbf{b}(y_k;\boldsymbol{\varphi})'\mathbf{u}_k(\boldsymbol{\varphi}). \tag{11}$$

The value of $\mathbf{u}_n(\boldsymbol{\varphi})$ can be calculated recursively as

$$\mathbf{u}_{n+1}(\boldsymbol{\varphi}) = \frac{\mathbf{A}(\boldsymbol{\varphi})'\mathbf{B}(y_n;\boldsymbol{\varphi})\mathbf{u}_n(\boldsymbol{\varphi})}{\mathbf{b}(y_n;\boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} \tag{12}$$

when initialized by $\mathbf{u}_1(\boldsymbol{\varphi}) = \boldsymbol{\pi}(\boldsymbol{\varphi})$.

Let $\mathbf{w}_n^{(l)}(\boldsymbol{\varphi}) = (\partial/\partial\varphi_l)\mathbf{u}_n(\boldsymbol{\varphi})$ be the partial derivative of $\mathbf{u}_n(\boldsymbol{\varphi})$ with respect to (wrt) the lth component of $\boldsymbol{\varphi}$. Each $\mathbf{w}_n^{(l)}(\boldsymbol{\varphi})$ is an r-length column vector, and

$$\mathbf{w}_n(\boldsymbol{\varphi}) = (\mathbf{w}_n^{(1)}(\boldsymbol{\varphi}), \mathbf{w}_n^{(2)}(\boldsymbol{\varphi}), \ldots, \mathbf{w}_n^{(p)}(\boldsymbol{\varphi})) \tag{13}$$

is an $r \times p$ matrix. Taking the derivative of $\mathbf{u}_{n+1}(\boldsymbol{\varphi})$ from Equation (12),

$$\begin{aligned}\mathbf{w}_{n+1}^{(l)}(\boldsymbol{\varphi}) &= \frac{\partial\mathbf{u}_{n+1}(\boldsymbol{\varphi})}{\partial\varphi_l}\\ &= R_1(y_n, \mathbf{u}_n(\boldsymbol{\varphi}), \boldsymbol{\varphi})\mathbf{w}_n^{(l)}(\boldsymbol{\varphi}) + R_2^{(l)}(y_n, \mathbf{u}_n(\boldsymbol{\varphi}), \boldsymbol{\varphi}), \end{aligned} \tag{14}$$

where

$$R_1(y_n, \mathbf{u}_n(\boldsymbol{\varphi}), \boldsymbol{\varphi}) = \mathbf{A}(\boldsymbol{\varphi})' \left[I - \frac{\mathbf{B}(y_n; \boldsymbol{\varphi})\mathbf{u}_n(\boldsymbol{\varphi})\mathbf{1}_r'}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} \right] \frac{\mathbf{B}(y_n; \boldsymbol{\varphi})}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} \qquad (15)$$

$$R_2^{(l)}(y_n, \mathbf{u}_n(\boldsymbol{\varphi}), \boldsymbol{\varphi}) = \mathbf{A}(\boldsymbol{\varphi})' \left[I - \frac{\mathbf{B}(y_n; \boldsymbol{\varphi})\mathbf{u}_n(\boldsymbol{\varphi})\mathbf{1}_r'}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} \right] \frac{[\partial \mathbf{B}(y_n; \boldsymbol{\varphi})/\partial \varphi_l]\mathbf{u}_n(\boldsymbol{\varphi})}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} + \frac{[\partial \mathbf{A}(\boldsymbol{\varphi})'/\partial \varphi_l]\mathbf{B}(y_n; \boldsymbol{\varphi})\mathbf{u}_n(\boldsymbol{\varphi})}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})}. \qquad (16)$$

Using these equations, we can recursively calculate $\mathbf{w}_n(\boldsymbol{\varphi})$ at every iteration.

For observation sequence $\langle y_1, \ldots, y_n \rangle$, we would like to maximize $p_n(y_1, \ldots, y_n; \boldsymbol{\varphi})$. Equivalently, we can maximize $\log p_n(y_1, \ldots, y_n; \boldsymbol{\varphi})$. Define the log-likelihood of observations $\langle y_1, \ldots, y_n \rangle$ as

$$\ell_n(\boldsymbol{\varphi}) = \frac{1}{n+1} \log p_n(y_1, ..., y_n; \boldsymbol{\varphi}). \qquad (17)$$

Using Equation (11), we can rewrite this as

$$\ell_n(\boldsymbol{\varphi}) = \frac{1}{n+1} \sum_{k=1}^{n} \log[\mathbf{b}(y_k; \boldsymbol{\varphi})'\mathbf{u}_k(\boldsymbol{\varphi})]. \qquad (18)$$

To estimate the set of optimal parameters $\boldsymbol{\varphi}^*$, we want to find the maximum of $\ell_n(\boldsymbol{\varphi})$, which we will attempt via recursive stochastic approximation. For each parameter l in $\boldsymbol{\varphi}$, at each time n, we take $(\partial/\partial\varphi_l)$ of the most recent term inside the summation in Equation (18), to form an "incremental score vector"

$$\mathbf{S}(\tilde{Y}_n; \boldsymbol{\varphi}) = \left(S^{(1)}(\tilde{Y}_n; \boldsymbol{\varphi}), ..., S^{(p)}(\tilde{Y}_n; \boldsymbol{\varphi}) \right)' \qquad (19)$$

with

$$S^{(l)}(\tilde{Y}_n; \boldsymbol{\varphi}) = \frac{\partial}{\partial \varphi_l} \log[\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})] = \frac{\mathbf{b}(y_n; \boldsymbol{\varphi})'[(\partial/\partial\varphi_l)\mathbf{u}_n(\boldsymbol{\varphi})] + [(\partial/\partial\varphi_l)\mathbf{b}(y_n; \boldsymbol{\varphi})]'\mathbf{u}_n(\boldsymbol{\varphi})}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} = \frac{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{w}_n(\boldsymbol{\varphi}) + [(\partial/\partial\varphi_l)\mathbf{b}(y_n; \boldsymbol{\varphi})]'\mathbf{u}_n(\boldsymbol{\varphi})}{\mathbf{b}(y_n; \boldsymbol{\varphi})'\mathbf{u}_n(\boldsymbol{\varphi})} \qquad (20)$$

where

$$\tilde{Y}_n \triangleq (Y_n, \mathbf{u}_n(\boldsymbol{\varphi}), \mathbf{w}_n(\boldsymbol{\varphi})). \qquad (21)$$

The RMLE algorithm takes the form

$$\boldsymbol{\varphi}_{n+1} = \Pi_G \left(\boldsymbol{\varphi}_n + \epsilon_n \mathbf{S}(\tilde{Y}_n; \boldsymbol{\varphi}_n) \right), \qquad (22)$$

where ϵ_n is a sequence of step sizes satisfying $\epsilon_n \geq 0$, $\epsilon_n \to 0$ and $\sum_n \epsilon_n = \infty$, G is a compact and convex set (here, $G \subseteq \Phi$, the set of all valid parameter sets $\boldsymbol{\varphi}$), and Π_G is a projection onto set G. The purpose of the projection is generally to ensure valid probability distributions and maintain all necessary conditions. Note that Equation (22) is a gradient update rule, with constraints. Equations (16) and (20) can both be simplified for each type of parameter in $\boldsymbol{\varphi}$.

Krishnamurthy and Yin [32] prove convergence of this algorithm for HMMs with continuous observations, as well as autoregressive models with Markov regime. Their proof is easily extended to the case of HMMs with discrete observations. In addition, several numerical examples of RMLE training are given in [32] and [34].

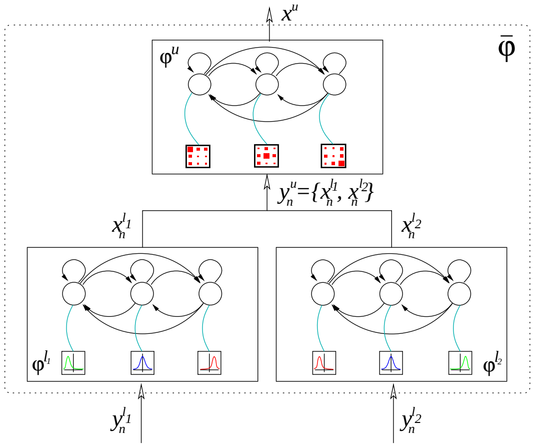


Fig. 6. Cascade of Hidden Markov Models. Models $\boldsymbol{\varphi}^{l_1}$, $\boldsymbol{\varphi}^{l_2}$, and $\boldsymbol{\varphi}^u$ are all HMMs.

III. HMM Cascade Model Description

Formally, let the topology of an HMM cascade model be as shown in Fig. 6; i.e., let our cascade model $\bar{\boldsymbol{\varphi}} = \{\boldsymbol{\varphi}^{l_1}, \boldsymbol{\varphi}^{l_2}, \boldsymbol{\varphi}^u\}$, where each component model $\boldsymbol{\varphi}^{l_j}$ and $\boldsymbol{\varphi}^u$ are HMMs defined according to the description in Section II-A. Let $\{X_n^{l_1}, Y_n^{l_1}\}$, $\{X_n^{l_2}, Y_n^{l_2}\}$, and $\{X_n^u, Y_n^u\}$ be the state and observation sequences corresponding respectively to $\boldsymbol{\varphi}^{l_1}$, $\boldsymbol{\varphi}^{l_2}$, and $\boldsymbol{\varphi}^u$. In this model, observations $Y_k^{l_j}$ of lower models $\boldsymbol{\varphi}^{l_j}$ are generally assumed to be continuous. The observations Y_k^u of upper model $\boldsymbol{\varphi}^u$ are the concatenated state sequences of the lower level models; i.e., $Y_k^u = (X_k^{l_1}, X_k^{l_2})$, and $\boldsymbol{\varphi}^u$ models the joint distribution of $X_k^{l_1}$ and $X_k^{l_2}$ for each state $j = 1, \ldots, r^u$, where r^u is the number of states in $\boldsymbol{\varphi}^u$. (To simplify calculations, we assume $X_k^{l_1}$ and $X_k^{l_2}$ to be independent, though this is not strictly necessary.)

A. Recursive maximum-likelihood estimation

Even though the individual component models are generative, it is impossible to generate data with this model with sufficient statistics to identify all model parameters. To see this, suppose we use upper model $\boldsymbol{\varphi}^u$ to generate state pair sequences $\{y_n^{u,1}, y_n^{u,2}\}$, and use these as the states of the lower models; i.e., let $x_n^{l_1} = y_n^{u,1}$ and $x_n^{l_2} = y_n^{u,2}$. In this situation, there is no direct dependence between $x_n^{l_j}$ and $x_{n-1}^{l_j}$, because we do not use the state transition matrix $\mathbf{A}(\boldsymbol{\varphi}^{l_j})$. On the other hand, if we use the state transition matrix to generate the next state, then there is no dependence on the upper model. We therefore cannot assume that input data was generated from a model with identical structure.

To alleviate this problem, we will make a slight modification of our original model for generative purposes, and then assume our proposed cascade model approximates this modified model. The modification we need is to make the states $x_n^{l_j}$ of the lower models dependent both on $x_{n-1}^{l_j}$ and on $y_n^{u,j}$. A graphical dynamic Bayesian network (DBN) showing this relationship is shown in Fig. 7. To generate this dependence,

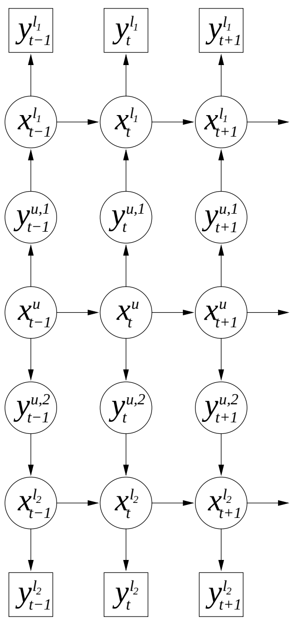


Fig. 7. A dynamic Bayesian network (DBN) model showing the dependence among output and state variables assumed by our cascade HMM. The cascade HMM cannot generate data with these dependencies, but this DBN can be fully implemented using a cascaded switching HMM.

we define a modification of an HMM called a switching HMM, whose name refers to its structural similarities to switching state-space models [35].

A switching HMM is a discrete-time stochastic process with two components, $\{X_n, Y_n\}$, defined on probability space $(\Omega, \mathcal{F}, P)$. Let $\{X_n\}_{n=1}^{\infty}$ be a discrete-time process with state space $R = \{1, \ldots, r\}$. Unlike an HMM, in a switching HMM the dynamics of X_n are determined by a *set* of Markov chains $\{\mathbf{A}^m(\lambda)\}$, $m = 1, \ldots, s$, with each chain having order r, and the transition probabilities for each chain defined as usual. An external discrete signal $q_n = 1, \ldots, s$, determines which Markov chain to use for the transition from X_{n-1} to X_n. As in an HMM, the process $\{Y_n\}$ is a probabilistic function of $\{X_n\}$, as we have defined previously. Let λ be the vector of parameters for this model. The topology of this model is shown in Fig. 8.

Proceeding, let the topology of our generating model be as shown in Fig. 9, and call this model a cascaded switching HMM. Define $\tilde{\varphi} = \{\varphi^u, \lambda^{l_1}, \lambda^{l_2}\}$, where φ^u is a finite-alphabet observation HMM as defined in Section II-A, and λ^{l_1} and λ^{l_2} are switching HMMs as defined above. We note that $\tilde{\varphi}$ can be formulated as a single HMM with an extended state space, although as with other hierarchical variations of HMMs [36], [37], the structure we have chosen is important for modeling our specific domain of study.

We will approximate $\tilde{\varphi}$ with a cascade model $\hat{\tilde{\varphi}} = \{\hat{\varphi}^u, \hat{\varphi}^{l_1}, \hat{\varphi}^{l_2}\}$, as shown in Fig. 10. Note that, unlike the cascaded switching HMM, the proposed cascade of HMMs cannot be expressed as a single HMM and obviously does not exactly match the model which generated the data. Nonetheless, the cascade of HMMs can still learn important information about the original model, and under appropriate conditions

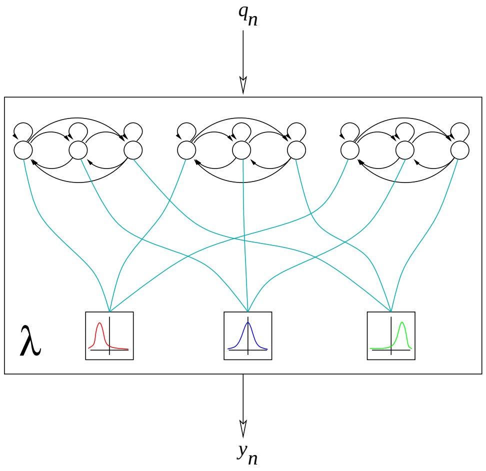


Fig. 8. A switching HMM. Each Markov chain in the model has the same number of states, with the same state in each chain corresponding to the same observation probability density function. Input q_n chooses which Markov chain to use for the transition from x_{n-1}.

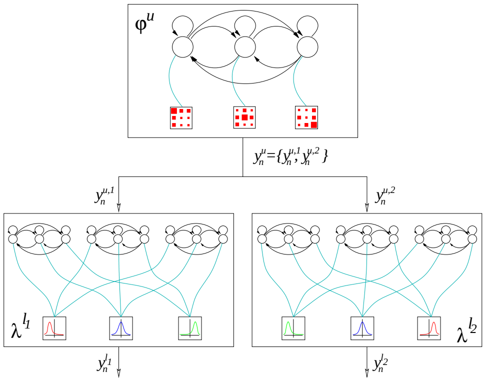


Fig. 9. A cascaded switching HMM. As a generator, an HMM φ^u outputs a discrete pair $\{y_n^{u,1}, y_n^{u,2}\}$, the components of which become the switches for a pair of switching HMMs λ^{l_1} and λ^{l_2}, selecting which Markov chain is used to determine the next transition.

(described below) learn a set of states in $\hat{\varphi}^u$ corresponding to those in φ^u. It also has the important benefit of being an easier model to learn, since we can learn each submodel of $\hat{\varphi}$ independently using the RMLE algorithm described in the previous section.

Comparing the two models $\tilde{\varphi}$ and $\hat{\varphi}$, we note that (1) in the cascaded switching HMM, state transitions are determined by a set of transition probability matrices $\{\mathbf{A}^m(\lambda^{l_1})\}$, which we attempt to model by a single transition probability matrix $\mathbf{A}(\hat{\varphi}^{l_1})$ in the cascade HMM, and (2) in the cascaded switching HMM, the joint observation densities in φ^u represent the selection of Markov chains in the switching HMMs, whereas in the cascade HMM, the joint distribution in $\hat{\varphi}^u$ correspond to the actual states in the lower models. We suggest that generally this joint distribution over states will be sufficient to identify states in the original HMM φ^u. However, we note that not all cascaded switching HMMs $\tilde{\varphi}$ will be identifiable. An example of a switching HMM that is not identifiable by our cascade model can be constructed (1) by selecting a particularly simple form for φ^u, such that each state deterministically selects

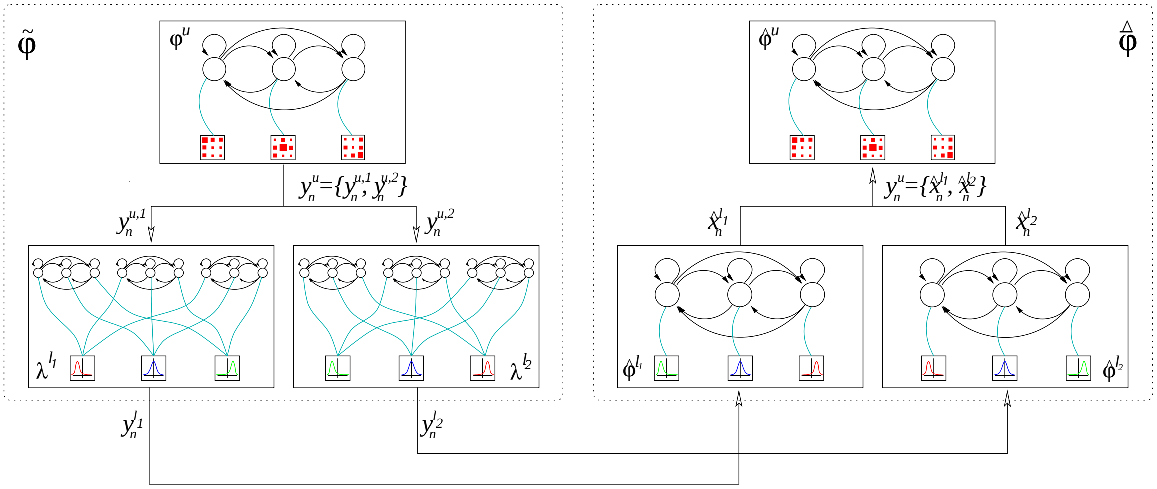


Fig. 10. Monte Carlo simulation for learning a cascaded switching HMM $\tilde{\varphi}$ using a cascade HMM $\hat{\tilde{\varphi}}$. The model on the left is a generative model, producing data for the model on the right to learn.

a single Markov chain in each of the switching HMMs, and then (2) by considering transition probability matrices in $\{\mathbf{A}^m(\lambda^{l_j})\}$, $j = 1, 2$, whose stationary distributions are identical, but whose actual transitions differ.

Proceeding, for the following analysis, assume that the number of states and the form of the density function in each HMM and switching HMM in the original model $\tilde{\varphi}$ are known, and that we are attempting with $\hat{\tilde{\varphi}}$ to learn a set of first-order transition probabilities and observation density parameters representing the original data.

Consider state-observation sequence pair $\{x_n^{l_1}, y_n^{l_1}\}$. There exists an HMM that represents the first-order statistics of this sequence, i.e., one that exactly matches the first-order transition probabilities and observation densities of this sequence. The model $\hat{\varphi}^{l_1}$ in our cascade structure will converge to this model when updated using the recursive maximum-likelihood algorithm presented in Section II-C. The same applies to $\hat{\varphi}^{l_2}$.

Next, consider the estimated composite state sequence $\{\hat{x}_n^{l_1}, \hat{x}_n^{l_2}\}$ recognized by the models $\hat{\varphi}^{l_1}$ and $\hat{\varphi}^{l_2}$. We will assume that, as models $\hat{\varphi}^{l_1}$ and $\hat{\varphi}^{l_2}$ converge, this sequence will be representative of the true state sequences in the switching HMMs λ^{l_1} and λ^{l_2} which generated the data. As above, we note that there then exists an HMM that can represent the first-order statistics of this sequence, which, again, we can learn through recursive maximum-likelihood estimation. Each state in model $\hat{\varphi}^u$ will correspond to a unique state in φ^u if the joint distribution of states in λ^{l_1} and λ^{l_2} is unique for each state in φ^u.

The main purpose of this section was to introduce the cascade of HMMs and cascaded switching HMMs, and justify recursive maximum-likelihood learning for the HMM-cascade model. See [34] for numerical simulations of the Monte Carlo simulation described above and shown in Fig. 10.

IV. HMM-Cascade Model for Concept Learning

Analysis of our model in Section III assumed that the data being analyzed came from the same underlying source. In fact, for concept learning, it is highly unlikely that the data was produced in this manner. A more likely scenario comes from Fig. 11, which is derived from Fig. 3. In this scenario, both the robot and the person have a model of the world, which here is represented by a cascade of HMMs. We assume that each model structurally allows the recognition of visual and auditory information present in the world (the lower level models), and further, that concepts can be inferred and understood from the sequence of discrete classifications of this auditory and visual information (using the upper level model).

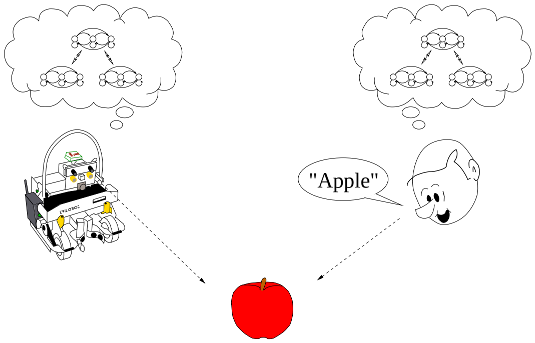


Fig. 11. Concept learning scenario using a cascade of HMMs. This model corresponds to Fig. 3, with the generic models replaced by HMMs.

It is assumed that the boy's model of the world is better or more complete than the robot's model and, therefore, that the goal of the robot is to learn the boy's model of the world. To reach this goal, the robot must try to garner information about each of the boy's submodels. To learn the boy's visual submodel, the robot will use visual data obtained from the world and assume that the boy's model was learned from similar information. For learning the boy's auditory submodel, the robot will use the boy's own "speech," and to learn the boy's concept model, the robot will attempt to find a relationship between what the boy says and what the world presents visually.

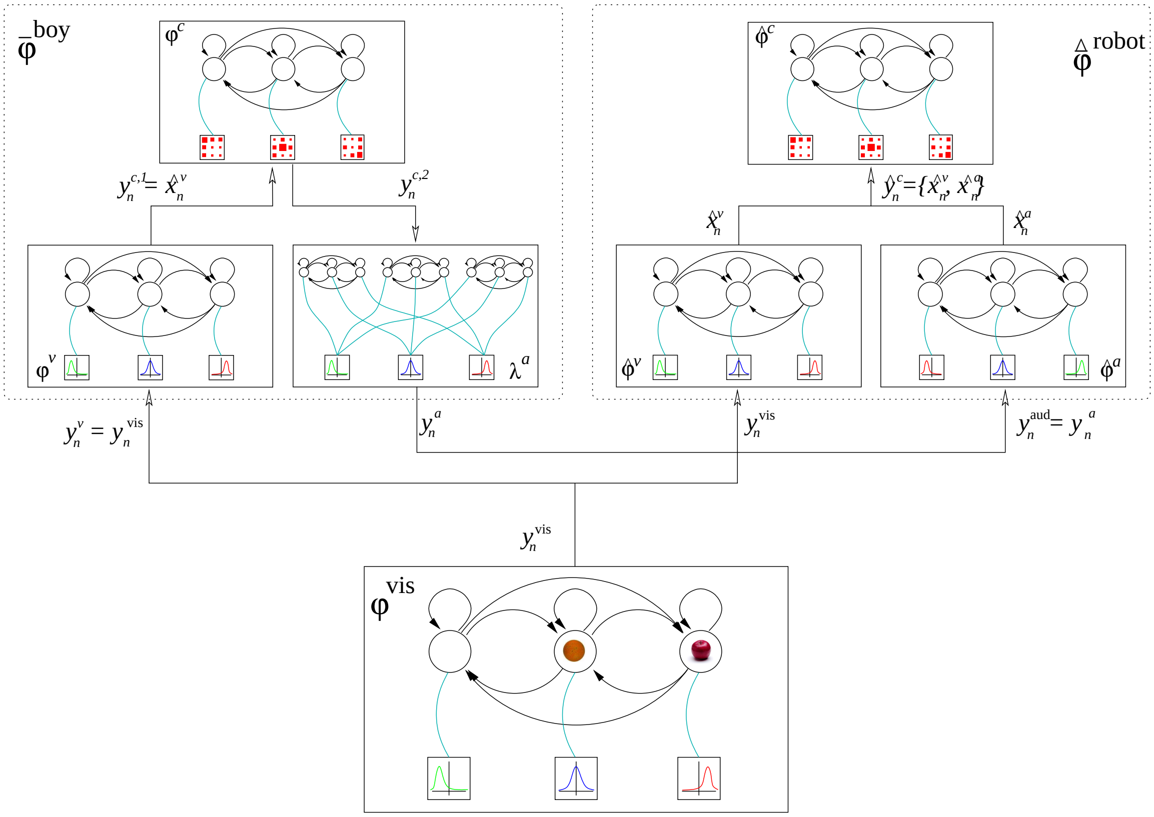


Fig. 12. Model topology for robot concept learning. The topology of this diagram corresponds to the scenario presented in Fig. 11. The lower model $\boldsymbol{\varphi}^{\text{vis}}$ is a model of the world producing visual outputs. The upper left model $\bar{\boldsymbol{\varphi}}^{\text{boy}}$ recognizes this visual input and produces auditory output. The upper right model $\hat{\boldsymbol{\varphi}}^{\text{robot}}$ accepts as input both the visual input produced by $\boldsymbol{\varphi}^{\text{vis}}$ and the auditory input produced by $\bar{\boldsymbol{\varphi}}^{\text{boy}}$, and trains its various submodels.

A. Model scenario

Fig. 12 shows the model topology of the scenario we envision. This scenario proceeds as follows:

1) The model $\boldsymbol{\varphi}^{\text{vis}}$ produces a stream of states $\{x_n^{\text{vis}}\}$ and corresponding visual features $\{y_n^{\text{vis}}\}$. The visual features $\{y_n^{\text{vis}}\}$ are accessible by both the boy and the robot. The stream of states may include such states as $x_n^{\text{vis}} = APPLE$ and $x_n^{\text{vis}} = NOTHING$.
2) The boy uses $\boldsymbol{\varphi}^v$ to recognize this visual stream, producing estimated state sequence $\{\tilde{x}_n^v\}$.
3) Using only the visual partial of the joint audio-visual state pdfs in concept model $\boldsymbol{\varphi}^c$, the boy "thinks" of the concept related to the visual input (i.e., chooses the most likely state $\tilde{x}_n^c$ concept state in $\boldsymbol{\varphi}^c$ corresponding to $\tilde{x}_n^v$).
4) The boy may choose, at random times, to "speak his mind." At these times, he uses the auditory observation pdf from state $\tilde{x}_n^c$ to produce $y_n^{c_a}$. This output becomes the switch for switching HMM λ^a, which produces output stream $\{y_n^{\text{aud}}\} = \{y_n^a\}$. It is assumed that the switch is "on" long enough to produce meaningful output from λ^a. At other times, the model λ^a produces "silence" (i.e., $x_n^a = SILENCE$, and y_n^a represents this state appropriately).
5) The robot simultaneously recognizes and learns (clusters) class information from visual input stream $\{y_n^{\text{vis}}\}$ with HMM $\hat{\boldsymbol{\varphi}}^v$, and auditory input stream $\{y_n^{\text{aud}}\}$ with HMM $\hat{\boldsymbol{\varphi}}^a$. These models produce estimated state sequences $\{\hat{x}_n^v\}$ and $\{\hat{x}_n^a\}$, respectively.
6) When both $\hat{x}_n^v$ and $\hat{x}_n^a$ have meaningful information (i.e., $\hat{x}_n^a \neq SILENCE$ and $\hat{x}_n^v \neq NOTHING$), model $\hat{\boldsymbol{\varphi}}^c$ both:
   a) updates (learns) using these inputs, (i.e., it clusters common co-occurrences), and
   b) estimates $\hat{x}^c$, its "thoughts" about the pair of inputs.
7) At other times, when only one of $\hat{x}_n^v$ and $\hat{x}_n^a$ have meaningful information, $\hat{\boldsymbol{\varphi}}^c$ uses only the partial pdf associated with that input to estimate $\hat{x}^c$, and the model is not updated.

When actually run on the robot, estimated state information from all of the robot models may be used by other programs (e.g., the controller) to make decisions.

B. Simulation results

Using the scenario outlined above, we ran a Monte Carlo simulation of the composite model. This section outlines those results. The following parameters were used for the fixed models $\boldsymbol{\varphi}^{\text{vis}}$ and $\bar{\boldsymbol{\varphi}}^{\text{boy}} = \{\boldsymbol{\varphi}^c, \lambda^a, \boldsymbol{\varphi}^v\}$. Let $\boldsymbol{\varphi}^{\text{vis}}$ be an HMM with Gaussian observations. Define its transition probability matrix as

$$\mathbf{A}(\boldsymbol{\varphi}^{\text{vis}}) = \begin{bmatrix} 0.90 & 0.05 & 0.05 \\ 0.04 & 0.95 & 0.01 \\ 0.04 & 0.01 & 0.95 \end{bmatrix},$$

and its Gaussian observation density parameters as

$$\mu(\boldsymbol{\varphi}^{\text{vis}}) = [\ 0 \quad 7 \quad 9\ ]'$$

and

$$\sigma^2(\boldsymbol{\varphi}^{\text{vis}}) = [\ 1.0 \quad 0.7 \quad 0.6\ ]'.$$

Let the boy's visual model $\boldsymbol{\varphi}^v$ be a learned version of $\boldsymbol{\varphi}^{\text{vis}}$, i.e., $\boldsymbol{\varphi}^v \approx \boldsymbol{\varphi}^{\text{vis}}$.

For the boy's auditory switched HMM λ^a, let the set of transition probability matrices $\{\mathbf{A}^m(\lambda^a)\}$, $1 \leq m \leq 3$, be defined as

$$\mathbf{A}^1(\lambda^a) = \begin{bmatrix} 0.94 & 0.02 & 0.02 & 0.02 \\ 0.94 & 0.02 & 0.02 & 0.02 \\ 0.94 & 0.02 & 0.02 & 0.02 \\ 0.94 & 0.02 & 0.02 & 0.02 \end{bmatrix},$$

$$\mathbf{A}^2(\lambda^a) = \begin{bmatrix} 0.05 & 0.45 & 0.45 & 0.05 \\ 0.01 & 0.90 & 0.08 & 0.01 \\ 0.01 & 0.70 & 0.28 & 0.01 \\ 0.05 & 0.45 & 0.45 & 0.05 \end{bmatrix},$$

and

$$\mathbf{A}^3(\lambda^a) = \begin{bmatrix} 0.05 & 0.05 & 0.45 & 0.45 \\ 0.05 & 0.05 & 0.45 & 0.45 \\ 0.05 & 0.05 & 0.10 & 0.80 \\ 0.01 & 0.01 & 0.08 & 0.90 \end{bmatrix},$$

and let the Gaussian parameters $\mu(\lambda^a)$ and $\sigma^2(\lambda^a)$ be

$$\mu(\lambda^a) = \begin{bmatrix} 0.0 & 3.0 & 5.0 & 7.0 \end{bmatrix}'$$

and

$$\sigma^2(\lambda^a) = \begin{bmatrix} 1.0 & 0.4 & 0.5 & 0.6 \end{bmatrix}'.$$

Finally, let the boy's concept HMM $\boldsymbol{\varphi}^c$ be defined by

$$\mathbf{A}(\boldsymbol{\varphi}^c) = \begin{bmatrix} 0.90 & 0.05 & 0.05 \\ 0.08 & 0.90 & 0.02 \\ 0.08 & 0.02 & 0.90 \end{bmatrix},$$

$$\tilde{b}^v(\boldsymbol{\varphi}^c) = \begin{bmatrix} 0.98 & 0.01 & 0.01 \\ 0.02 & 0.90 & 0.08 \\ 0.02 & 0.08 & 0.90 \end{bmatrix},$$

and

$$b^a(\boldsymbol{\varphi}^c) = \begin{bmatrix} 0.96 & 0.02 & 0.02 \\ 0.10 & 0.90 & 0.00 \\ 0.10 & 0.00 & 0.90 \end{bmatrix}.$$

Note that $\tilde{b}^v(\boldsymbol{\varphi}^c)$ represents a distribution over the states of $\boldsymbol{\varphi}^v$, whereas $b^a(\boldsymbol{\varphi}^c)$ represents a distribution over the selection of Markov chains $\{\mathbf{A}^m(\lambda^a)\}$. These two distributions are not and cannot be used simultaneously.

As in the cascade model simulation in Section III, assume we know the number of states and type of distribution for each of the models, so that $\hat{\bar{\boldsymbol{\varphi}}}^{\text{robot}} = \{\hat{\boldsymbol{\varphi}}^c, \hat{\boldsymbol{\varphi}}^a, \hat{\boldsymbol{\varphi}}^v\}$ has (approximately) the correct topology to learn the given models. As in the previous section, means and variances for the observation densities of $\hat{\boldsymbol{\varphi}}^a$ and $\hat{\boldsymbol{\varphi}}^v$ were initialized using k-means initialization on the first 1000 outputs of the generative model, Gaussian noise with zero mean and standard deviation one was then added to the initial means, and noise with zero mean and standard deviation 0.5 was added to the variances. For recursive maximum-likelihood training, we let learning rate $\varepsilon_k = \frac{0.006}{k^{0.2}}$, where $k = min(n, 1000)$, and n is the iteration number.

TABLE I

AVERAGE CLASSIFICATION ACCURACY FOR LEARNED HMM-CASCADE $\hat{\boldsymbol{\varphi}}$ OVER 50 SIMULATION RUNS. THE NUMBER IN PARENTHESIS IS STANDARD DEVIATION.

	Maximum-Likelihood Classification	Viterbi Classification
$\hat{\boldsymbol{\varphi}}^a$	90.1%(3.7%)	89.9%(4.2%)
$\hat{\boldsymbol{\varphi}}^v$	97.9%(1.6%)	99.1%(2.4%)
$\hat{\boldsymbol{\varphi}}^c$	98.4%(1.1%)	98.8%(1.1%)

Fig. 13 shows the progression of a training run for model $\hat{\boldsymbol{\varphi}}^c$, where each subfigure shows the progression of the parameter values through time. As can be seen from the graphs, most of the parameters converge quite rapidly. Parameters which converge more slowly, such as those in Fig. 13(c), are those that are tracking changes in lower models which have not yet converged.

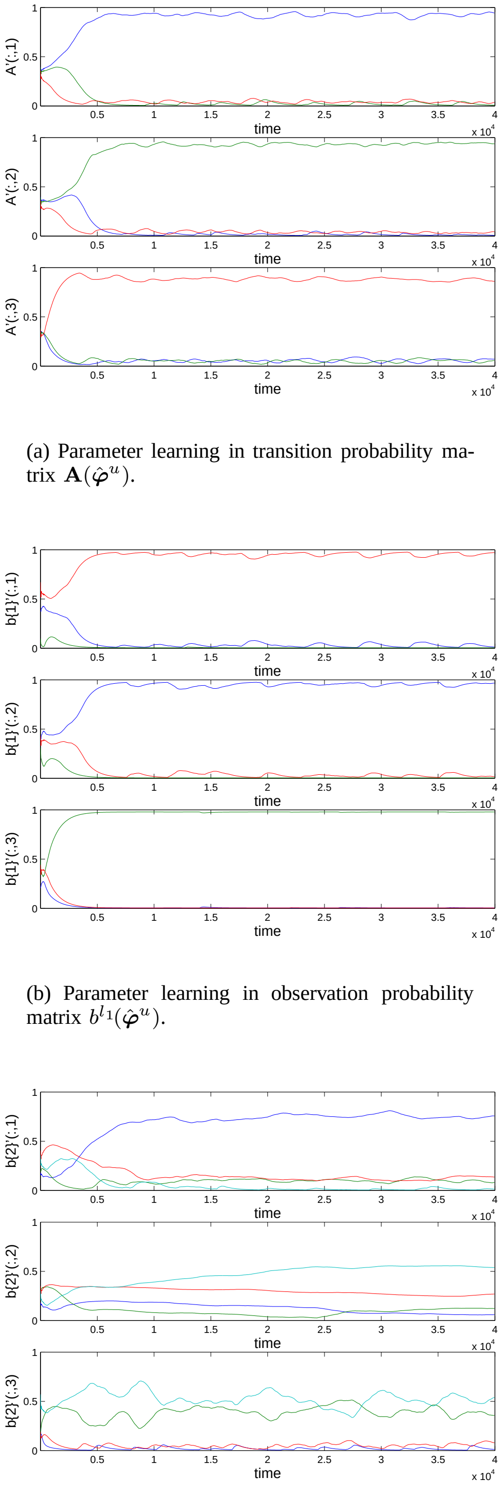
We repeated this experiment for 50 training episodes of 50 000 iterations each, and compared the state sequences of each submodel in $\hat{\bar{\boldsymbol{\varphi}}}$ to sequences from $\bar{\boldsymbol{\varphi}}$ using both maximum-likelihood and viterbi estimates of the states. These comparisons are summarized in Table I. The cascade model showed a high degree of robustness.

V. ROBOTIC EXPERIMENTS

The basic component of the robotic experiment testing our model is similar to the simulated scenario presented in the previous section: the robot and a person are looking at the same object, the person names the object or some aspect of it, and the robot, over time, learns the association between that word or phrase and the visual features of the object. Here we describe a scenario which incorporates this experiment.

In our scenario, our robot is wandering around a benign environment, and is instinctually motivated to look for "interesting" things. We expect the following behaviors:

1) It will be attracted to objects, especially ones that it has not seen before, or not seen recently; it will "play" with these objects, attempting to first pick them up, then knock them over.
2) It will be attracted by loud noises, turning toward them and assuming, e.g., that someone wants to get its attention.
3) Using our proposed cascade model, it will
   a) learn to recognize the visual objects in its environment,
   b) learn to recognize distinct words spoken to it, and
   c) learn the concepts associated with the various words and objects.



(a) Parameter learning in transition probability matrix $\mathbf{A}(\hat{\boldsymbol{\varphi}}^u)$.

(b) Parameter learning in observation probability matrix $b^{l_1}(\hat{\boldsymbol{\varphi}}^u)$.

(c) Parameter learning in observation probability matrix $b^{l_2}(\hat{\boldsymbol{\varphi}}^u)$.

Fig. 13. Parameter learning for model $\hat{\boldsymbol{\varphi}}^c$. Although the learned model cannot be directly compared to the original model, these graphs show that the model parameters converge.

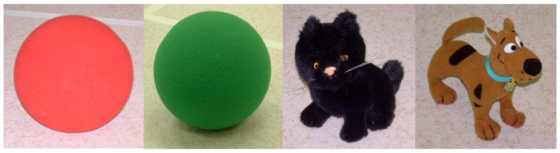


Fig. 14. Objects used in our robot demonstration.

TABLE II
LIST OF WORDS USED IN OUR ROBOT DEMONSTRATION.

animal
ball
cat
dog
green ball
red ball

4) Also using our HMM cascade, it will demonstrate that it recognizes these concepts by
   a) recognizing a word, choosing a corresponding concept, and finding an object which also matches that concept, and
   b) recognizing an object and saying the name of a concept corresponding to that object.

The behaviors listed in numbers one and two above were first demonstrated by McClain [38]. The demonstration described here builds on his work and on the work of others, including

- sound source localization research by Li and Levinson [39], [40],
- speech feature extraction and synthesis research by Kleffner [41], and
- visual feature extraction by Lin [42].

The specific objects we are using in this demonstration are shown in Fig. 14, and the list of words and phrases we say are listed in Table II. These words were chosen to test the learning concepts for specifically named objects (such as *cat*) as well as concepts for general categories (such as *animal*). Although not necessary, the concepts we initially learn correspond directly to the words and phrases listed in Table II.

Because they pertain directly to our work, autonomous exploration and speech and visual feature extraction are briefly discussed below. This discussion is followed by a description of the implementation of our HMM cascade model for our robots in Section V-C.

A. Finite state machine controller

The central component of the above experiment is a finite state machine (FSM) controller developed by McClain [38] as a part of an autonomous exploration mode for our robot . This controller continuously evaluates the state of the robot and its environment, and uses this information to make decisions and produce specific types of behavior. For our experiment, we modified the state machine and its related programs to use information from our associative memory when making decisions, as well as to facilitate learning in our model. A description of each state is as follows:

1) Explore: look around for something interesting.

   a) If we see an interesting object (such as one we have not seen before), go to state 2.
   b) If we hear an interesting (i.e., loud) sound, go to state 6.
2) Found object: an object is visible.
   a) If it is far away, approach it, study what it looks like, and stay in state 2.
   b) If it is near, go to state 3.
3) Learn name: learn the name or feature of an object.
   a) If we hear something, repeat it and try to associate it with this object; stay in state 3.
   b) After a period of silence, go to state 4.
4) Play 1: play with the object.
   a) Approach and attempt to pick up the object; go to state 5.
5) Play 2: play with the object.
   a) Try to knock the object over; go back to state 1.
6) Interact: listen for known sounds.
   a) If we hear the name of an object we know, look for it; go to state 7.
   b) If we hear something we do not know, beep and stay in state 6.
   c) If we do not hear anything for a short period, go back to state 1.
7) Search: look for a particular object.
   a) If we have not found the object, keep looking, and stay in state 7.
   b) If we have not found the object after a long time, give up and go to state 1.
   c) If we find the desired object, say the name (if we know it), and go to state 2.

The role our HMM cascade associative memory plays changes depending on the state of the model. We describe these roles below in Section V-C.

B. Sensory inputs

We are running this experiment on a real robot with real sensory inputs, so in addition to the FSM controller, our associative memory needs features extracted from live speech and visual inputs. For speech data analysis, we are extracting energy, voicing confidence, and a set of log-area ratios (LARs) from a 16-kHz audio stream. This processing is based on work developed by Kleffner [41] for speech imitation for the robot. Typically, around 8-12 LARs plus pitch and voicing information can be used to synthesize a very accurate reproduction of the speech signal. For our work, we are currently extracting three LARs, log energy, and voicing confidence on consecutive 20-ms segments of audio, giving us a stream of length-five feature vectors at 50 Hz. Despite the short length of this feature vector, these features are very representative of the speech signal; using only the three extracted LARs and voicing information, we can still synthesize speech that is intelligible.

For visual data analysis, the current experiment is using a robust segmentation and feature extraction algorithm developed by Lin [42]. The segmentation portion of this work is based on loopy belief propagation [43]. After image segmentation, the feature extractor presents a length-10 visual feature vector for each object in an image, consisting of

1) a normalized length-eight color histogram,
2) the first moment of the object shape, and
3) the height/width ratio of the object.

These features are calculated at a rate of about 2 sets per second. Descriptions of the audio feature extraction and the visual segmentation and feature extraction algorithms appear in [34, Appendix B].

C. HMM cascade model setup

For the auditory HMM in the HMM-cascade, we created a simple word recognizer, with one state (classification) per word. For the visual HMM $\hat{\varphi}^{\mathrm{vis}}$, we used a four state HMM to recognize features from the objects show in Fig. 14. For visual features, we chose color histogram, moment, and width to height ratio. For each object, we obtained a feature vector for 200 images of that object taken from multiple perspectives. These vectors were quantized before being used to initialize the densities. Transition probabilities were initialized uniformly, and the model was then trained for 10 epochs on the same data using RMLE. We only used 10 epochs to train with because the initial density estimates were close to their optimal values.

The concept model, $\hat{\varphi}^{c}$, is a discrete, finite-observation HMM, with observations covering the joint state spaces of the audio and visual models. Since the word HMM ran slower than the visual HMM, the classification output of the auditory model was upsampled to match the classification rate of the faster visual model, and timestamps were used to align the signals before presentation to the concept HMM. We initialized the concept model with six states, corresponding to the six words/phrases in our word list in Table II. The transition probabilities were initialized uniformly, and the observation probabilities were initialized by hand to bias them slightly toward the desired concepts. For example, for the state we chose to corresponding to “ball,” the observation probabilities corresponding to the visually recognized red and green balls were given slightly higher probabilities than the observation probabilities corresponding to the cat and the dog, and the observation probability corresponding to the word “ball” was given a slightly higher value than those probabilities corresponding to other words.

Depending on the mode of the finite state machine, certain parts of the model are inactive. Specifically, referring to the FSM described in Section V-A, when in states 1, 2, 4, 5, and 7, auditory input is ignored: the object HMM recognizes visual inputs, and the concept model uses the marginal density corresponding to the states of the object HMM to determine its state. In state 6, where the robot is listening for speech input, the opposite happens: visual input is ignored, the auditory model attempts to recognize spoken words, and the state of the word model alone determines the state of the concept model. Finally, in state 3, both audio and visual inputs are present. All models are active, and recognition and learning is done in the concept model with both inputs. Note that learning is possible in both the auditory and visual models in any state where the model of interest is active. For the experiment here, we chose not to enable learning in these models.

D. Implementation Issues

There are a few miscellaneous issues we must deal with in our experiment, depending on the current state of the FSM. The first issue is that multiple objects may be present within a scene. When this happens, each visual object is presented in sequence to the visual HMM. In this way, the transition probabilities in the visual HMM would come to represent information about the spatial relationship between various objects, in that objects that are close to one another will frequently be presented to the HMM sequentially.

Depending on the state of the FSM, one of these objects is identified as a target object. For example, in state 2, the target object would be the object first identified as "interesting" in state one. In subsequent iterations, the robot will remember and attempt to track this target object, e.g., so that it can be played with later.

Because we have stereo cameras, we additionally must handle correspondence. Currently, at every iteration the model recognizes objects in each image separately, and then correspondence is determined using the recognition labels (i.e., the recognized states of $\hat{\varphi}^v$) for each image. Objects which appear in only one of the images are currently ignored. We do not currently handle the situation where there are multiple objects of the same visual class present.

A final potential issue is object occlusion, where only a portion of an object appears in an image. As of right now, this has not been a serious issue. In the case where an object is misclassified because it is occluded in one image, but fully visible in the other, correspondence is not drawn between the two objects. If the robot is looking for this object, it will eventually find it when it moves or turns its head. As the robot approaches an object, the bottom of the object may also be cut off; in this case, we lower its head. Even in the case of a partially occluded object, the recognition has generally proven robust enough to do proper recognition. This issue will likely become more important as we increase the number of objects.

VI. Robotic Experiment Results

Our goal in this experiment is to show that the concept model $\hat{\varphi}^{\text{con}}$ can be learned from a set of real inputs. As described above, we initialized and trained the auditory and visual models off-line using recorded auditory and visual features, respectively. Note that, even though the training occurred off-line, we used recursive maximum-likelihood estimation to learn the model parameters, so this training could be done online.

For the concept model, we initialized the model as described in Section V-C above, i.e., we initially set all of the transition probabilities equal, and by hand initialized the discrete observation probabilities so that they would have a slight bias to particular concepts.

The model was then trained using RMLE during the simulation run. Specifically, when the FSM entered state 3, the robot would sit in front of a target object. The visual model $\hat{\varphi}^v$ would continuously recognize this object, and the auditory model $\hat{\varphi}^a$ would recognize words that were spoken into a close-talk microphone. When a word was spoken and recognized, the state $\hat{x}^a$ of model $\hat{\varphi}^a$ corresponding to that word and the state $\hat{x}^v$ of model $\hat{\varphi}^v$ were presented to the concept model, and the model was updated according to the RMLE algorithm. To speed up training, each input pair was presented 10 times each time the a word was recognized. This process was repeated multiple times for each object as the robot wandered around and played with its toys. Fig. 15 shows a picture taken during this training.

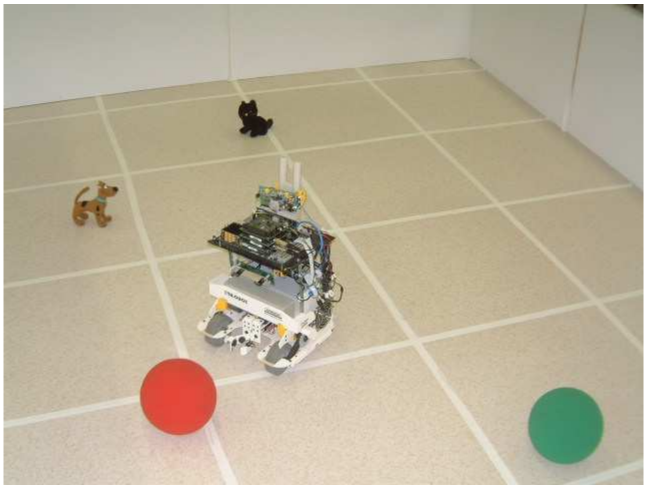


Fig. 15. Illy learning about various objects. In this scenario, as Illy approaches various objects, she stops and waits for a verbal description consisting of short words or phrases. Over time, she associates these spoken words with the object.

TABLE III
Trained transition probabilities for the concept HMM. These values were initialized uniformly (i.e., all values started at $\frac{1}{6}$).

	animal	ball	cat	dog	green ball	red ball
animal	0.4211	0.0856	0.1670	0.1840	0.0699	0.0723
ball	0.0723	0.4760	0.0597	0.0721	0.1659	0.1540
cat	0.1931	0.1017	0.3717	0.1479	0.0925	0.0931
dog	0.2023	0.0911	0.1307	0.4115	0.0776	0.0867
green ball	0.1082	0.2142	0.1002	0.1051	0.3321	0.1401
red ball	0.1105	0.1951	0.1026	0.1186	0.1434	0.3298

For the training run shown here, we ran the robot for about 30 min. The final trained transition and observation probabilities are shown in Tables III, IV, and V.

A. Discussion

The actual results shown are an indication of the long-term capabilities of the model. Specifically, the model parameters had not converged after 30 min, but the parameter values did move in a direction which indicated convergence to a useful state.

Trained transition probabilities in Table III indicate (1) a general affinity for "thinking" of the same object at consecutive time steps (as indicated by high diagonal values), and (2) a slightly smaller but discernible relationship between related classifications (e.g., the animal state was more likely to be followed by a cat or dog state than any of the other states). These transition probabilities strongly reflect the order of words presented to the model, which is reasonable considering (1) we only trained the model when a word was present, (2) we often repeated the same word consecutively, and (3) when we did not repeat a word consecutively, we often spoke another word related to the same object.

For both auditory and visual inputs, the observation probabilities for each state in the concept HMM were biased slightly at the beginning of training toward a particular outcome. For the observation probabilities learned through the end of training, those probabilities referring to visible objects (in Table IV) are the more interesting of the two. For example, the concept *ball* initially corresponded to a visual representation of the red or green ball with probabilities 0.3 and 0.3, and to a visual representation of the cat or dog with probabilities 0.2 and 0.2. The trained values of these states showed a stronger proclivity to all initial biases. Taking the *ball* example again, the final observation probabilities for the red ball and green ball were approximately 0.43 and 0.42, respectively, and the observation probabilities for the visual representations of cat and dog went down accordingly (see Table IV). The same was true for other observation probabilities.

The results here indicate that our HMM cascade model can learn a set of concepts using features extracted from live auditory and visual inputs measured by a mobile robot exploring its environment. This learned information can then be used by the robot's controller module to make important behavioral decisions.

VII. Conclusion

This paper has discussed the use of our cascade of HMMs as an associative memory. First, simulation results representing a real-world scenario indicated that this model is viable for learning associations among concurrent stationary regions of multiple input streams, where each of these stationary regions are modeled by a state in a hidden Markov model. Next, a live version of this scenario was run on the robot, whereby features were extracted from auditory and visual streams, classified by a HMM, and these classifications then used as input to a concept HMM for training and additional classification.

The robotic implementation of our HMM cascade model presented here is a proof of concept for an important idea. Specifically, we are able to take noisy, real-world analog inputs, convert them to symbols (by classifying them using rather crude features), and present them to a controller for use in making important decisions (for example, whether to approach and play with a particular toy, or look for another). In other words, our robot is making symbolic decisions based on discrete representations of the real world around it. In addition, when classifying and learning about real world outputs, the model learns to associate related auditory and visual information with the same (symbolic) concept. The model is learned online using a robust maximum-likelihood estimator.

The work described in this paper explored the case where each concept in our concept HMM corresponded to exactly one word, though potentially multiple visual objects. One of our future experiments is to learn concepts which could refer to both multiple words and multiple objects. Another plan is to grow the cascade model as new visual objects or words are presented to the robot.

References

[1] A. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, 1950.

[2] R. Brooks, C. Breazeal, M. Marjanovic, B. Scassellati, and M. Williamson, "The Cog project: Building a humanoid robot," in *Computation for Metaphors, Analogy and Agents*, C. Nehaniv, Ed. Berlin: Springer-Verlag, 1998, pp. 52–87.

[3] P. Varshavskaya, "Behavior-based early language development on a humanoid robot," in *Proc. 2nd Int. Workshop on Epigenetic Robotics*, 2002.

[4] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[5] H. Kozima and H. Yano, "A robot that learns to communicate with human caregivers," in *Proc. 1st Int. Workshop on Epigenetic Robotics*, Lund, Sweden, 2001.

[6] J. Weng, Y. Zhang, and Y. Chen, "Developing early senses about the world: 'Object permanence' and visuoauditory real-time learning," in *Proc. INNS/IEEE Int. Joint Conf. Neural Networks*, vol. 4, July 2003, pp. 2710–2715.

[7] N. Almássy, G. M. Edelman, and O. Sporns, "Behavioral constraints in the development of neuronal properties: a cortical model embedded in a real world device," *Cerebral Cortex*, vol. 8, pp. 346–361, 1998.

[8] O. Sporns and W. H. Alexander, "Neuromodulation in a learning robot: Interactions between neural plasticity and behavior," in *Proc. INNS/IEEE Int. Joint Conf. Neural Networks*, vol. 4, July 2003, pp. 2789–2794.

[9] A. K. Seth, J. L. McKinstry, G. M. Edelman, and J. L. Krichmar, "Visual binding, reentry and neuronal synchrony in a physically situated brain-based device," in *Proc. 3rd Int. Workshop on Epigenetic Robotics*, 2003.

[10] G. Stojanov, "Petitagé: A case study in developmental robotics," in *Proc. 1st Int. Workshop on Epigenetic Robotics*, Lund, Sweden, 2001.

[11] K. Fischer and R. Moratz, "From communicative strategies to cognitive modelling," in *Proc. 1st Int. Workshop on Epigenetic Robotics*, Lund, Sweden, 2001.

[12] L. Hugues and A. Drogoul, "Shaping of robot behaviors by demonstration," in *Proc. 1st Int. Workshop on Epigenetic Robotics*, Lund, Sweden, 2001.

[13] P. R. Cohen, C. Sutton, and B. Burns, "Learning effects of robot actions using temporal associations," in *Proc. 2nd Int. Conf. on Development and Learning*, Cambridge, MA, June 2002, pp. 96–101.

[14] I. Fasel, G. O. Deák, J. Triesch, and J. Movellan, "Combining embodied models and empirical research for understanding the development of shared attention," in *Proc. 2nd Int. Conf. on Development and Learning*, Cambridge, MA, June 2002, pp. 21–27.

[15] R. A. Grupen, "A developmental organization for robot behavior," in *Proc. 3rd Int. Workshop on Epigenetic Robotics*, 2003.

[16] M. Lungarella and G. Metta, "Beyond gazing, pointing, and reaching: A survey of developmental robotics," in *Proc. 3rd Int. Workshop on Epigenetic Robotics*, 2003.

[17] D. R. Shanks, *The Psychology of Associative Learning*. Cambridge: Cambridge University Press, 1995.

[18] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice Hall PTR, 1993.

[19] P. Boufounos, S. El-Difrawy, and D. Ehrlich, "Hidden Markov models for DNA sequencing," in *Workshop on Genomic Signal Processing and Statistics (GENSIPS 2002)*, Oct. 2002.

[20] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler, "Hidden Markov models in computational biology: Applications to protein modeling," *J. of Molecular Biology*, vol. 235, pp. 1501–1531, 1994.

[21] D. Kulp, D. Haussler, M. G. Reese, and F. H. Eeckman, "A generalized hidden Markov model for the recognition of human genes in DNA," in *Proc. 4th Int. Conf. Intell. Syst. Molecular Bio.*, 1996, pp. 134–142.

[22] R. L. Cave and L. P. Neuwirth, "Hidden Markov models for English," in *Proc. of the Symposium on the Applications of Hidden Markov Models to Text and Speech*. Princeton, NJ: IDA-CRD, Oct. 1980, pp. 16–56.

[23] A. B. Poritz, "Linear predictive hidden Markov models and the speech signal," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Processing*, 1982, pp. 1291–1294.

[24] A. Ljolje and S. E. Levinson, "Development of an acoustic-phonetic hidden Markov model for continuous speech recognition," *IEEE Trans. Signal Processing*, vol. 29, no. 1, pp. 29–39, 1991.

[25] A. Arapostathis and S. I. Marcus, "Analysis of an identification algorithm arising in the adaptive estimation of Markov chains," *Math Control Signals Systems*, vol. 3, no. 1, pp. 1–29, 1990.

[26] I. B. Collings, V. Krishnamurthy, and J. B. Moore, "On-line identification of hidden Markov models via recursive prediction error techniques," *IEEE Trans. Signal Processing*, vol. 42, no. 12, pp. 3535–3539, Dec. 1994.

[27] F. LeGland and L. Mével, "Recursive estimation in hidden Markov models," in *Proc. 36th IEEE Conf. Decision Contr.*, San Diego, CA, Dec. 1997.

[28] U. Holst and G. Lindgren, "Recursive estimation in mixture models with Markov regime," *IEEE Trans. Inform. Theory*, vol. 37, no. 6, pp. 1683–1690, Nov. 1991.

[29] V. Krishnamurthy and J. B. Moore, "On-line estimation of hidden Markov model parameters based on the Kullback-Leiber information measure," *IEEE Trans. Signal Processing*, vol. 41, no. 8, pp. 2557–2573, Aug. 1993.

[30] T. Rydén, "On recursive estimation for hidden Markov models," *Stochastic Processes and their Applications*, vol. 66, pp. 79–96, 1997.

[31] F. LeGland and L. Mével, "Recursive identification of HMM's with observations in a finite set," in *Proc. 34th IEEE Conf. Decision Contr.*, New Orleans, Dec. 1995, pp. 216–221.

[32] V. Krishnamurthy and G. G. Yin, "Recursive algorithms for estimation of hidden Markov models and autoregressive models with Markov regime," *IEEE Trans. Inform. Theory*, vol. 48, no. 2, pp. 458–476, Feb. 2002.

[33] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *The Bell System Technical Journal*, vol. 62, no. 4, pp. 1035–1074, Apr. 1983.

[34] K. Squire, "A robotic framework for semantic learning," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2004, (forthcoming).

[35] Z. Ghahramani and G. Hinton, "Variational learning for switching state-space models," *Neural Computation*, vol. 12, pp. 831–864, 2000.

[36] Z. Ghahramani and G. E. Hinton, "Factorial hidden Markov models," *Machine Learning*, vol. 29, pp. 245–273, 1997.

[37] S. Fine, Y. Singer, and N. Tishby, "The hierarchical hidden Markov model: Analysis and applications," *Machine Learning*, vol. 32, no. 1, pp. 41–62, 1998.

[38] M. McClain, "The role of exploration in language acquisition for an autonomous robot," M.S. thesis, University of Illinois at Urbana-Champaign, 2003.

[39] D. Li and S. E. Levinson, "A robust linear phase unwrapping method for dual-channel sound source localization," in *Int. Conf. on Robot. Automat.*, Washington D.C., May 2002.

[40] ——, "A Bayes-rule based hierarchical system for binaural sound source localization," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Processing*, Hong Kong, Apr. 2003.

[41] M. Kleffner, "A method of automatic speech imitation via warped linear prediction," M.S. thesis, University of Illinois at Urbana-Champaign, 2003.

[42] R.-S. Lin (personal communication), 2004.

[43] K. P. Murphy, Y. Weiss, and M. I. Jordan, "Loopy belief-propagation for approximate inference: An empirical study," in *Proc. 15th Conf. Uncertainty in Artif. Intell.*, K. B. Laskey and H. Prade, Eds., San Mateo, CA, 1999.

TABLE IV

TRAINED OBSERVATION PROBABILITIES USED BY THE CONCEPT HMM FOR VISIBLE OBJECTS. THE HORIZONTAL AXIS REFERS TO THE CONCEPT CLASS, AND THE VERTICAL AXIS REFERS TO THE CLASSIFIED VISUAL OBJECT.

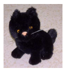
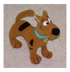
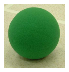
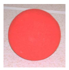
	animal	ball	cat	dog	green	red
					ball	ball
	0.4086	0.0761	0.5412	0.3193	0.0996	0.1041
	0.3987	0.0744	0.2665	0.5077	0.0974	0.1027
	0.0957	0.4215	0.0963	0.0603	0.5232	0.2777
	0.0970	0.4280	0.0960	0.1127	0.2799	0.5155

TABLE V

TRAINED OBSERVATION PROBABILITIES USED BY THE CONCEPT HMM FOR WORDS. THE HORIZONTAL AXIS REFERS TO THE CONCEPT CLASS, AND THE VERTICAL AXIS REFERS TO THE CLASSIFIED SPOKEN WORD.

					green	red
	animal	ball	cat	dog	ball	ball
"animal"	0.6728	0.0530	0.1977	0.2134	0.0576	0.0605
"ball"	0.0660	0.7088	0.0572	0.0509	0.2166	0.1896
"cat"	0.0769	0.0268	0.5738	0.0575	0.0385	0.0398
"dog"	0.1112	0.0735	0.1022	0.6194	0.0647	0.0882
"green ball"	0.0396	0.0699	0.0369	0.0306	0.5550	0.0722
"red ball"	0.0334	0.0681	0.0322	0.0282	0.0676	0.5496