Classification of time series with hidden Markov models: Unsupervised learning and self-organization

V. Breuer and G. Radons
Institut für Theoretische Physik der Christian-Albrechts-Universität Kiel, Olshausenstrasse 40, 24098 Kiel, Germany
(Received 18 August 1995)

Unsupervised learning and feature recognition with hidden Markov models (HMM's) is investigated. The well-known Baum-Welch algorithm is utilized to tune the free parameters of the HMM. The local state probability distribution of the model controls the recognition. For a simple problem we show analytically the self-organization of a feature detector. In a numerical simulation we present a detector for two-dimensional textures that perceives, recognizes, and restores disturbed textures.

PACS number(s): 87.10.+e, 02.50.Ga, 42.30.Sy

I. INTRODUCTION

A general problem in analyzing complex data is the detection of significant structures (features). The possibility of parsing a given data sample into such features provides an effective reduction of the data preserving the relevant information, and hence is a big step forward for the understanding of a code or an underlying process.

In many applications, the segmentation of the data into units with different features is known *a priori*. In this case, classifiers can be trained by (supervised) learning from examples. For complex time series, such as, e.g., in speech recognition [1–3], for measurements of ion channel currents [4], for neuronal [5–8], medical [9], and molecular biological [10,11] applications, or for the analysis of DNA sequences [12], hidden Markov models were applied as maximal likelihood classifiers [13].

Supervised learning is no longer applicable if the "meaning" of each particular time series in the training set is unknown. In a self-organizing way, one has to detect significant structures which are appropriate to subdivide the training set into subsets with equal structure. Tools appropriate for such problems are vector-quantization (VQ) methods [14]. VQ methods exhibit a competition among classifiers, e.g., the "winner take all" learning in Kohonen networks [15]: the adaptation of a classifier (perceptron) to a new piece of data is weighted with the actual distance. For data with few degrees of freedom those vector quantization methods were applied very successfully. With increasing dimensionality (e.g., long time series) and for a continuous signal source (unsegmented data), the application of VQ methods becomes more difficult. Recently, hidden Markov models (HMM's) were used to detect different modes of neuronal activity [8].

We investigate unsupervised feature detection with hidden Markov models. This approach remains applicable for high dimensional data, for an unknown number of features, and for unsegmented data.

In Sec. II we briefly review the principles of HMM's. Subsequently, in Sec. III we show analytically the self-organization of a HMM to a feature detector. Here the data are divided into segments containing only one feature and the structure of the HMM is given *a priori*. Subsequently, in Sec. IV we show in a numerical simulation the emergence of domains of states in a HMM that represent different features in the unsegmented training data.

II. BASICS OF HIDDEN MARKOV MODELS

A hidden Markov model is an abstract object consisting of a given number of states z_i, $i = 1,\ldots,N$, and transitions between these states. Transitions occur with probabilities a_{ij}, i.e., a_{ij} is the conditional probability $p(j \mid i)$ for making a transition to state z_j, if the system is in state z_i. The a_{ij} have the property $\sum_{j=1}^{N} a_{ij} = 1$ for all i, which means that some transition occurs with probability 1. Therefore they can be considered as elements of a stochastic matrix A, the transition matrix.

So far this defines a simple Markov process because the probability for the next state z_j depends only on the current state z_i. Hidden Markov models are characterized by the additional ingredient that on every transition $z_i \to z_j$ one defines a probability distribution $a_{ij}(y)$ for emitting a symbol y of some alphabet $\mathbf{O}$. Some symbol is generated with certainty on every state z_i which means that $\sum_{\{y\}} \sum_{\{j\}} a_{ij}(y) = 1$ for every i. The meaning of the symbols depends on the actual application. This explains why HMM's are often called probabilistic functions of Markov processes.

In order to generate symbol sequences with such models one also has to specify an initial probability distribution $\pi^0 = (\pi_1^0,\ldots,\pi_N^0)$ over the states z_i. A symbol sequence of length T, $\sigma^1\sigma^2\cdots\sigma^T$ with $\sigma^t \in \mathbf{O}$, $t = 1,\ldots,T$, is generated as follows. One randomly selects an initial state according to the distribution π^0, e.g., state z_i with probability π_i^0. Then, one emits a symbol σ^1 and jumps to another state, say z_j with probability $a_{ij}(\sigma^1)$. Subsequently one emits another symbol σ^2 and jumps to another state z_k with probability $a_{jk}(\sigma^2)$, and so on.

The probability $p(\sigma^1 \cdots \sigma^T)$ for generating a sequence $\sigma^1 \cdots \sigma^T$ is obtained by summing the probabilities from all paths through the automaton which are compatible with this sequence. It can be calculated as

$$p(\sigma^1 \cdots \sigma^T) = \pi^0 A(\sigma^1)\ A(\sigma^2)\ \cdots A(\sigma^{T-1})\ A(\sigma^T)\hat{\eta} \quad (1)$$

where we introduced the matrices $A(\sigma)$, $\sigma \in \mathbf{O}$, with elements $a_{ij}(\sigma)$. The multiplication with the vector $\hat{\eta} = (1, 1, \ldots, 1)^T$ provides a summation over all states, which means that the system is allowed to be in any of the N states z_j while it emits the last symbol σ^T. In the speech recognition literature the evaluation of expression (1) is called the forward-backward algorithm, which reflects that the product can be calculated iteratively from left to right (forward) or from right to left (backward).

To summarize, a hidden Markov model is defined by a set of parameters which includes the initial probability distribution π^0 and the set of transition matrices $A(y)$ with y from the alphabet $\mathbf{O}$. These parameters are conveniently collected in one vector denoted $\vec{\lambda}$, that is,

$$\vec{\lambda} = (\pi^0, \{A(y)\})\ . \quad (2)$$

In view of (1) and (2) one can identify a given HMM also with a parametrized probability distribution over all symbol sequences $\sigma^1\sigma^2 \cdots \sigma^T$ of arbitrary length T. In Sec. IV we also consider HMM's with input [16–18]. In these models the transition matrices $A(y)$ are replaced by input-dependent matrices $A(y|x)$, with x being some symbol of an input alphabet.

In general it is a hard problem to find those model parameters that reflect most accurately the probability distribution of a given set of symbol sequences. A powerful optimization method is the recursive reestimation of the model parameters by their expectation values (with respect to the given data), starting with some first-guess initialization. This procedure is called the Baum-Welch algorithm in the literature. This algorithm guarantees a stepwise decrease of the Kullback-Leibler distance between the probability distributions over symbol sequences of the data and the model.

III. UNSUPERVISED LEARNING

The Baum-Welch algorithm for the reestimation of the parameters of a HMM implies a competition among possible paths through the state graph of the HMM to become most probable with respect to the training data. This can be motivated by plausibility arguments [17].

The Baum-Welch reestimation formulas for probability distributions $P(\sigma)$ of symbol strings σ can be derived in a straightforward manner from those for single symbol strings [2,1,18]. For any such distribution the Baum-Welch reestimation formulas are given by

$$\bar{a}_{ij}(y) = \frac{\sum_\sigma \frac{P(\sigma)}{p(\sigma|M)} \sum_t \pi_i^{t-1,\sigma} a_{ij}(y) \eta_j^{t,\sigma} \delta_{\sigma^t,y}}{\sum_\sigma \frac{P(\sigma)}{p(\sigma|M)} \sum_t \pi_i^{t-1,\sigma} \eta_i^{t-1,\sigma}},$$

$$\bar{\pi}_i^0 = \sum_\sigma \frac{P(\sigma)}{p(\sigma|M)} \pi_i^0 \eta_i^{0,\sigma}. \quad (3)$$

Here π_i^0 and $a_{ij}(y)$ [respectively the vector π^0 and the matrices $A(y)$] are the parameters of the HMM. They are the probability to start in state z_i and the compound probability for the generation of symbol y during a transition from state z_i to z_j, respectively. The new estimates of the parameters are indicated by an overbar. The vector quantities $\pi^{t,\sigma}$ and $\eta^{t,\sigma}$ are the forward-backward probabilities:

$$\pi^{t,\sigma} = \pi^0 \prod_{\tau=1}^{t} A(\sigma^\tau), \quad (4)$$

$$\eta^{t,\sigma} = \left[\prod_{\tau=t+1}^{|\sigma|} A(\sigma^\tau)\right] \hat{\eta}, \quad \hat{\eta} = (1, \ldots, 1). \quad (5)$$

$p(\sigma|M)$ is the probability for the HMM M to generate the sequence of symbols σ [i.e., $p(\sigma|M) = \pi^{t,\sigma}\eta^{t,\sigma}\ \forall t$].

Consider now a probability distribution of binary sequences that can (in principle) be obtained from the following coin-tossing procedure. From an urn containing K different sorts κ of coins choose one and perform N subsequent tosses. If the number of coins of sort κ in the urn is n_κ and the probability to obtain "0" by tossing the coin is $q^{(\kappa)}$ ($1 - q^{(\kappa)}$ to obtain "1") then $P(\sigma)$ is given by

$$P(\sigma) = \sum_{\kappa=1}^{K} Q(\kappa) P^{(\kappa)}(\sigma) \quad (6)$$

with

$$P^{(\kappa)}(\sigma) = \prod_{t=1}^{N} [q^{(\kappa)}\delta_{\sigma^t,0} + (1 - q^{(\kappa)})\delta_{\sigma^t,1}]$$

and

$$Q(\kappa) = \frac{n_\kappa}{\sum n_{\kappa'}}.$$

With $\varrho(\sigma)$ being the fraction of symbols $y = 0$ in σ, one can write

$$P(\sigma) = \sum_{\kappa=1}^{K} Q(\kappa)(q^{(\kappa)})^{N\varrho(\sigma)}(1 - q^{(\kappa)})^{N[1-\varrho(\sigma)]} \quad (7)$$

Note that the probability $P(\varrho)$ to observe a sequence with $\varrho(\sigma) = \varrho$ becomes locally maximal for values of $\varrho = q^{(\kappa)}$:

$$P(\varrho) = \sum_{\kappa=1}^{K} Q(\kappa) P^{(\kappa)}(\varrho),$$

$$P^{(\kappa)}(\varrho) = \binom{N}{N\varrho} (q^{(\kappa)})^{N\varrho}(1 - q^{(\kappa)})^{N(1-\varrho)}. \quad (8)$$

These local maxima correspond to the different features in the sequences generated by this process.

Data distributed according to $P(\sigma)$ serve to train a HMM. We consider HMM's consisting of isolated states, i.e., transitions between different states $z_i \to z_j$ are forbidden. The transition matrices are diagonal in this case.

$$a_{ij}(y) = \left\{ \begin{array}{cc} 0 & \text{for } i \neq j \\ a_{ii} & \text{for } y = 0 \\ 1 - a_{ii} & \text{for } y = 1 \end{array} \right\} \quad \text{for} \quad i = j. \tag{9}$$

The probability $p(\sigma|M)$ for such a HMM M to generate σ is given by

$$p(\sigma|M) = p[\varrho(\sigma)|M] = \sum_i \pi_i^0 [a_{ii}^{\varrho(\sigma)}(1 - a_{ii})^{[1-\varrho(\sigma)]}]^N. \tag{10}$$

With the substitution

$$p_i[\varrho(\sigma)] \stackrel{def}{=} a_{ii}^{\varrho(\sigma)}(1 - a_{ii})^{[1-\varrho(\sigma)]}, \tag{11}$$

one gets

$$p(\sigma|M) = \sum_i \pi_i^0 p_i[\varrho(\sigma)]^N. \tag{12}$$

Using (12) and the relation $\pi_i^{t-1,\sigma} a_{ii}(\sigma^t)\eta_i^{t,\sigma} = \pi_i^0 p_i[\varrho(\sigma)]^N$ we get for the Baum-Welch formulas

$$\bar{a}_{ii}(y) = \frac{\sum_\sigma \frac{P(\sigma)}{p(\sigma|M)} \sum_t \pi_i^{t-1,\sigma} a_{ii}(y) \eta_i^{t,\sigma} \delta_{\sigma^t,y}}{\sum_\sigma \frac{P(\sigma)}{p(\sigma|M)} \sum_t \pi_i^{t-1,\sigma} \eta_i^{t-1,\sigma}} \tag{13}$$

$$= \frac{1}{\sum_\sigma P(\sigma) \frac{\pi_i^0 p_i[\varrho(\sigma)]^N}{\sum_j \pi_j^0 p_j[\varrho(\sigma)]^N} \sum_t 1} \sum_\sigma P(\sigma) \frac{\pi_i^0 p_i[\varrho(\sigma)]^N}{\sum_j \pi_j^0 p_j[\varrho(\sigma)]^N} \sum_t \delta_{\sigma^t,y}, \tag{14}$$

$$\stackrel{y=0}{\Rightarrow} \bar{a}_{ii} = \frac{1}{\sum_\sigma P(\sigma) \frac{\pi_i^0 p_i[\varrho(\sigma)]^N}{\sum_j \pi_j^0 p_j[\varrho(\sigma)]^N}} \sum_\sigma P(\sigma) \frac{\pi_i^0 p_i[\varrho(\sigma)]^N}{\sum_j \pi_j^0 p_j[\varrho(\sigma)]^N} \varrho(\sigma), \tag{15}$$

and

$$\bar{\pi}_i^0 = \sum_\sigma P(\sigma) \frac{\pi_i^0 p_i[\varrho(\sigma)]^N}{\sum_j \pi_j^0 p_j[\varrho(\sigma)]^N}. \tag{16}$$

The terms in the sums over sequences σ only depend on $\varrho(\sigma)$. One can write

$$\bar{a}_{ii} = \frac{\sum_{\kappa=1}^K \sum_{N\varrho=0}^N Q(\kappa) P^{(\kappa)}(\varrho) \frac{\pi_i^0 p_i(\varrho)^N}{\sum_j \pi_j^0 p_j(\varrho)^N} \varrho}{\sum_{\kappa=1}^K \sum_{N\varrho=0}^N Q(\kappa) P^{(\kappa)}(\varrho) \frac{\pi_i^0 p_i(\varrho)^N}{\sum_j \pi_j^0 p_j(\varrho)^N}}, \tag{17}$$

$$\bar{\pi}_i^0 = \sum_{\kappa=1}^K \sum_{N\varrho=0}^N Q(\kappa) P^{(\kappa)}(\varrho) \frac{\pi_i^0 p_i(\varrho)^N}{\sum_j \pi_j^0 p_j(\varrho)^N}. \tag{18}$$

In the limit of sequences of infinite length the distribution $P(\varrho)$ is given by

$$\lim_{N\to\infty} P(\varrho) = \sum_{\kappa=1}^K Q(\kappa) \delta(\varrho - q^{(\kappa)}), \tag{19}$$

and hence the reestimation formulas become

$$\lim_{N\to\infty} \bar{a}_{ii} = \frac{1}{\sum_\kappa Q(\kappa) \frac{\pi_i^0 p_i(q^{(\kappa)})^N}{\sum_j \pi_j^0 p_j(q^{(\kappa)})^N}} \times \sum_\kappa Q(\kappa) \frac{\pi_i^0 p_i(q^{(\kappa)})^N}{\sum_j \pi_j^0 p_j(q^{(\kappa)})^N} q^{(\kappa)}, \tag{20}$$

$$\lim_{N\to\infty} \bar{\pi}_i^0 = \sum_\kappa Q(\kappa) \frac{\pi_i^0 p_i(q^{(\kappa)})^N}{\sum_j \pi_j^0 p_j(q^{(\kappa)})^N}. \tag{21}$$

After rearranging the states of the model such that $a_{11} \leq a_{22} \leq \cdots$, the quotients in (20) and (21) can be expressed in terms of the Heaviside step function $\Theta(x)$:

$$\lim_{N\to\infty} \frac{\pi_i^0 p_i(\varrho)^N}{\sum_j \pi_j^0 p_j(\varrho)^N} = \frac{\Theta\left(\varrho - \hat{\varrho}_{(i,i-1)}\right) - \Theta\left(\varrho - \hat{\varrho}_{(i,i+1)}\right)}{1 + \sum_{j\neq i} \delta_{a_{ii},a_{jj}}}, \tag{22}$$

with $\hat{\varrho}_{(i,j)}$ being the solution of the equation

$$\frac{p_i(\hat{\varrho}_{(i,j)})}{p_j(\hat{\varrho}_{(i,j)})} = 1 \quad \Rightarrow \quad \hat{\varrho}_{(i,j)} = \frac{1}{1 - \frac{\ln\left(\frac{a_{ii}}{a_{jj}}\right)}{\ln\left(\frac{1-a_{ii}}{1-a_{jj}}\right)}}. \tag{23}$$

The quantities $\hat{\varrho}_{(0,1)}$ and $\hat{\varrho}_{(|Z_M|,|Z_M|+1)}$ are defined as 0 and 1, respectively. The sum $\sum_{j\neq i} \delta_{a_{ii},a_{jj}}$ in (22) is the degree of degeneracy $\deg(i)$ of state z_i in the HMM.

For every i there exists a subinterval $I(i) =]\hat{\varrho}_{(i,i-1)}, \hat{\varrho}_{(i,i+1)}[\in [0,1]$, in which the right-hand side of (22) is nonzero. In other words, the probability $p_i(\varrho)$ dominates all other probabilities $p_{j\neq i}(\varrho)$ if ϱ is chosen from $I(i)$. Inserting (22) into (20) and (21), one gets

$$\bar{a}_{ii} = \frac{\sum_{\kappa} q^{(\kappa)} Q(\kappa) \left[\Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right) - \Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right)\right]}{\sum_{\kappa} Q(\kappa) \left[\Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right) - \Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right)\right]}, \tag{24}$$

$$\bar{\pi}_i^0 = \frac{1}{1+\deg(i)} \sum_{\kappa} Q(\kappa) \left[\Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right) - \Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right)\right]. \tag{25}$$

From Eqs. (24) one obtains the transition probabilities a_{ii} by calculating the average over all $q^{(\kappa)} \in I(i)$ weighted with the *a priori* probability $Q(\kappa)$ for coin κ. According to (25), the initial probabilities π_i^0 for nondegenerate states are given by the sum of all *a priori* probabilities $Q(\kappa)$ of coins κ, for which $q^{(\kappa)}$ is an element of $I(i)$.

If there is none of the $q^{(\kappa)}$ within the interval $I(i)$, (24) is no longer valid and one must utilize Eq. (20). It is convenient to write (20) in the form

$$\bar{a}_{ii} = \frac{1}{\sum_{\kappa} \frac{1}{1+\sum_{j\neq i}[\pi_j^0 p_j(q^{(\kappa)})^N / \pi_i^0 p_i(q^{(\kappa)})^N]}} \sum_{\kappa} \frac{1}{1+\sum_{j\neq i}[\pi_j^0 p_j(q^{(\kappa)})^N / \pi_i^0 p_i(q^{(\kappa)})^N]} q^{(\kappa)} \tag{26}$$

$$\overset{=}{N\to\infty} \frac{1}{\sum_{\kappa} \frac{1}{\max_{j\neq i}[\pi_j^0 p_j(q^{(\kappa)})^N / \pi_i^0 p_i(q^{(\kappa)})^N]}} \sum_{\kappa} \frac{1}{\max_{j\neq i}[\pi_j^0 p_j(q^{(\kappa)})^N / \pi_i^0 p_i(q^{(\kappa)})^N]} q^{(\kappa)} \tag{27}$$

$$= \frac{1}{\sum_{\kappa}[\pi_i^0 p_i(q^{(\kappa)})^N / \pi_{j^*}^0 p_{j^*}(q^{(\kappa)})^N]} \sum_{\kappa} \frac{\pi_i^0 p_i(q^{(\kappa)})^N}{\pi_{j^*}^0 p_{j^*}(q^{(\kappa)})^N} q^{(\kappa)}. \tag{28}$$

The index $j^*(\kappa) \neq i$ determines the interval $I(j^*)$ in which $q^{(\kappa)}$ resides. With $\kappa_\mu = \arg\max_\kappa [\pi_i^0 p_i(q^{(\kappa)})^N / \pi_{j^*}^0 p_{j^*}(q^{(\kappa)})^N]$, one can write

$$\bar{a}_{ii} \overset{=}{N\to\infty} \left[1 \Big/ \frac{\pi_i^0 p_i(q^{\kappa_\mu})^N}{\pi_{j^*}^0 p_{j^*}(q^{\kappa_\mu})^N}\right] \frac{\pi_i^0 p_i(q^{\kappa_\mu})^N}{\pi_{j^*}^0 p_{j^*}(q^{\kappa_\mu})^N} q^{\kappa_\mu} = q^{\kappa_\mu}. \tag{29}$$

Hence, in the limit of sequences of infinite length, the Baum-Welch reestimation formulas for the "new" parameters can be written in compact form as

$$\bar{a}_{ii} = \begin{cases} \dfrac{\sum_{\kappa} q^{(\kappa)} Q(\kappa) \left(\Theta\left[q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right] - \Theta\left[q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right]\right)}{\sum_{\kappa} Q(\kappa) \left(\Theta\left[q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right] - \Theta\left[q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right]\right)} & \text{if } \exists\kappa \text{ such that } q^{(\kappa)} \in I(i) \\ q^{\kappa_\mu},\ \kappa_\mu = \arg\max_{\kappa} \left[\dfrac{p_i(q^{(\kappa)})}{p_{j^*}(q^{(\kappa)})}\right] & \text{else,} \end{cases} \tag{30}$$

$$\bar{\pi}_i^0 = \frac{1}{1+\deg(i)} \sum_{\kappa} Q(\kappa) \left[\Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i-1)}\right) - \Theta\left(q^{(\kappa)} - \hat{\varrho}_{(i,i+1)}\right)\right], \tag{31}$$

using step functions and the maximal-argument function, whose arguments $\hat{\varrho}_{(i,i\pm1)}$ and $p_i(q^{(\kappa)})$, depend on the "old" parameters according to (23) and (11), respectively.

From (30) it is evident that the values of the reestimated parameters $\bar{a}_{ii}$ are identical to a particular $q^{(\kappa)}$ or to the average over some of the $q^{(\kappa)}$. Furthermore, after the convergence of the iteration (which is always guaranteed [1]), each interval $I(i)$ contains at least one $q^{(\kappa)}$. Each of the states z_i thus has learned (without any supervision) one feature of the distribution $P(\sigma)$, i.e., the "tossing statistic" of a particular coin, or at least a mixture of such features.

Additionally, if the number of states is known to be equal to the number K of different features, and if the resulting HMM is nondegenerate (the degree of degeneracy depends on the initialization of the HMM), every feature is represented by one state of the HMM and its *a priori* probability is equal to the initial probability of this state. Such a HMM is an optimal model of the process.

A given sequence can easily be classified by calculating the actual state distribution $\Pi(t)$ of the HMM, i.e., the vector of probabilities to be in one of the states z_i at time

t given the sequence σ. $\Pi(t)$ can be expressed in terms of the forward-backward probabilities:

$$\Pi_i(t) = \frac{\pi_i^{t,\sigma}\eta_i^{t,\sigma}}{p(\sigma|M)}. \tag{32}$$

For isolated states, the actual state distribution does not explicitly depend on t. In this case, the maximal component of Π is equal to the probability of the most likely path. The index of this component also determines the most likely state to generate σ and thus can be utilized to classify the feature.

IV. TEXTURE RECOGNITION WITH HMMS

In the last section we showed the ability of HMM's to learn the features of a mixture of Bernoulli processes. To show that unsupervised learning with HMM's is possible even for more complex data, we applied HMM's to a texture recognition task.

In our numerical simulation, we used a square-shaped region of 100×100 pixels. The pixels are colored black and white. There are two domains of different textures. Within a circle around the center, the texture consists of alternating black and white vertical stripes. In the outer region, there is a checkerboard texture. These textures are additionally perturbed by independently changing the color of each pixel with a probability of p =0.3 (Fig. 1).

We generated training data for the HMM by the following procedure. Starting at an arbitrary pixel, we performed random walks (each 10 000 steps, imposing periodic boundaries) through the textured region. With equal probability a step to the right or downwards is chosen and the color of the actual square is stored with binary coding for direction and color (0=down, 1=right and 0=black, 1=white, respectively), we get pairs of sequences determining the path and the color of the pixels along the path. Using the Baum-Welch algorithm, we trained an input-dependent HMM [16–18] with these pairs of sequences, where the movement direction is the input and the actual color is the output of the HMM.

The resulting HMM exhibits a structure that mirrors the nature of the data. Figure 2 shows the transition probabilities of the HMM for all combinations of input and output. There are two pairs of states (z_1, z_4) and (z_2, z_3), between which state transitions have very low probability, whereas the probabilities of transitions between the states of one pair are orders of magnitude higher. The pair (z_1, z_4) describes the situation within the (disturbed) checkerboard texture: for every movement direction (input), the state of the HMM alters between z_1 and z_4. The color (output) also alternates, but with a probability of 0.3 (small arrow) a "wrong" color is detected. The other pair (z_2, z_3) corresponds to the striped texture: for a horizontal movement, the relations are similar, but for a step down, the HMM remains in its actual state (z_2 or z_3). While moving vertically, one of the colors is detected with higher probability than the other ("wrong") color.

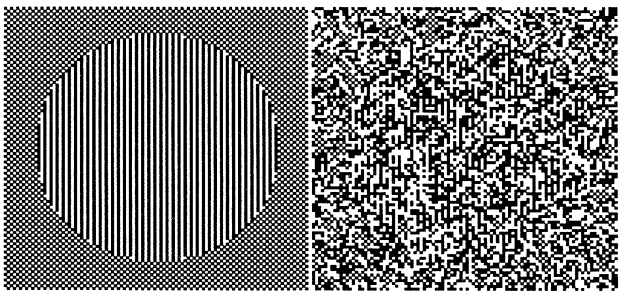

FIG. 1. Unperturbed texture in a square of 100×100 pixels (left) and perturbed texture (right), where each pixel was flipped with a probability $p = 0.3$.

The trained HMM can now be used to detect the domains of the different textures. This can be done by performing a new walk (random or not) through the region. For a given trajectory and sequence of detected colors, one has to calculate the actual state distribution $\Pi(t)$ for every step, given by (32). The components of this distribution can be utilized to recognize the textures (or the relative position within a texture) along the trajectory. To visualize the result of the recognition, we used a color code of the state distribution (Fig. 3). A pure color (red, blue, green, or yellow) corresponds to the probability 1 to be in one particular state. A high color saturation thus indicates a distribution with little entropy, i.e., a relative position within a texture is recognized with high accuracy. On the other hand, a gray color corresponds to high entropy and hence indicates ambiguity of the recognition. In Fig. 3 the different texture domains can now easily be distinguished by their colors. Note that, in contrast to our analytical example, no information about the number of features in the data was assumed.

V. CONCLUSION

For a long time hidden Markov models have been used very successfully as classifiers for complex signals. The adjustment of the model parameters typically has to be performed by supervised learning from examples.

Motivated by the competitive nature of the Baum-Welch algorithm, we investigated the applicability of HMM's to unsupervised learning. For a HMM initialized with isolated states, we have shown analytically that the states adapt to different features in the training data.

In a numerical experiment, we successfully applied a HMM to texture detection and recognition in a noisy image: the different texture domains were identified correctly and the original image could be restored. All numerical algorithms we used were derived from standard tools. We used arbitrarily initialized HMM's and observed the emergence of a topology of weakly connected subsets of states (submodels) with strong interconnections. Those submodels represent the different features in the data. Features can be recognized by calculating the local state probability distribution for the trained HMM.

We propose unsupervised hidden Markov learning as an alternative to neural networks approaches and other vector-quantization methods.

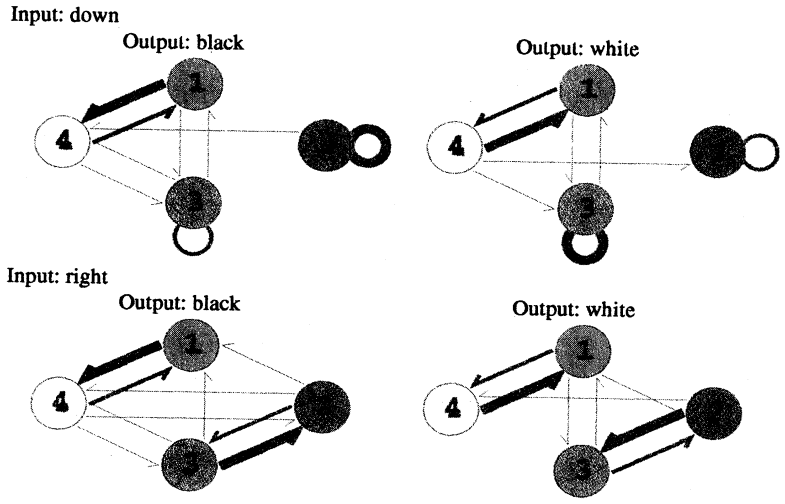

FIG. 2. The hidden Markov model after training. The thickness of the arrows indicates the transition probabilities between states for the different combinations of input (movement) and output (color). Transitions between the pairs of states (z_1, z_4) and (z_2, z_3) have only small probabilities.

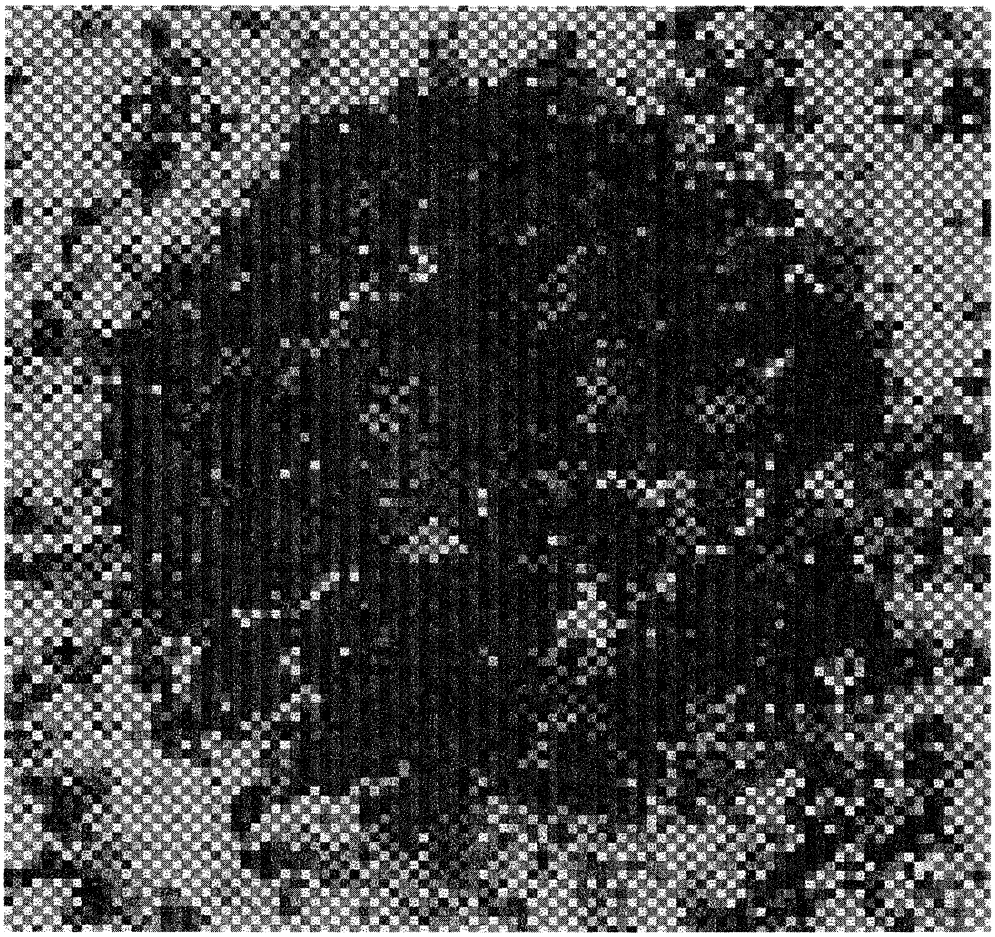

FIG. 3. Recognition of the perturbed textures. For every step on a trajectory through the square, the state distribution of the trained hidden Markov model is represented by a color code. Each color is a mixture of the pure colors of the states (see Fig. 2) with respect to their particular weight. The two texture domains can now be separated by their colors (red and yellow for the checkerboard texture and blue and green for the striped texture).

[1] L. R. Rabiner, Proc. IEEE **77**, 257 (1989).
[2] L. R. Bahl, F. Jelinek, and R. L. Mercer, IEEE Trans. Pattern Anal. Machine Intell. **PAMI-5**, 179 (1983).
[3] X. D. Huang, Y. Ariki, and M. A. Jack, *Hidden Markov Models for Speech Recognition* (Edinburgh University Press, Edinburgh, 1990).
[4] S. H. Chung, V. Krishnamurthy, and J. B. Moore, Philos. Trans. R. Soc. London B **334**, 357 (1991).
[5] I. Gat, N. Tishby, in *Advances in Neural Information Processing Systems,* edited by J. E. Moody, S. J. Hanson, and R. P. Lippmann (Morgan Kaufmann, San Mateo, CA, 1993), Vol. 5, p. 945.
[6] G. Radons, J. D. Becker, B. Dülfer, and J. Krüger, Biol. Cybern. **71**, 359 (1994).
[7] G. Radons and J. D. Becker, in *Mustererkennung 1993*, edited by S.J. Pöppl and H. Handels (Springer, Berlin, 1993), p. 498.
[8] M. Abeles, H. Bergman, I. Gat, I. Meilijson, E. Seidelmann, N. Tishby, and E. Vaadia, Proc. Natl. Acad. Sci. U.S.A. **92**, 8616 (1995).
[9] B. G. Leroux and M. L. Puterman, Biometrics **48**, 545 (1992).
[10] P. Baldi, Y. Chauvin, T. Hunkapiller, and M. A. McClure, Proc. Natl. Acad. Sci. U.S.A. **91**, 1059 (1994).
[11] A. Krogh, M. Brown, I. S. Mian, K. Sjolander, and D. Haussler, J. Mol. Biol. **235**, 1501 (1994).
[12] G. A. Churchill, Bull. Math. Biol. **51**, 79 (1989).
[13] R. O. Duda und P. E. Hart, *Pattern Classification and Scene Analysis* (John Wiley & Sons, New York, 1973).
[14] R. M. Gray, IEEE Acoust. Speech Signal Process. Mag. **1**, 4 (1984).
[15] T. Kohonen, Biol. Cybern. **43**, 59 (1982).
[16] L. A. Zadeh and E. Polak, *System Theory* (McGraw-Hill, New York, 1969).
[17] V. Breuer, Ph.D. thesis, University of Kiel, 1995.
[18] V. Breuer, G. Radons, IEEE Trans. Signal Process. **43**, 1313 (1995).

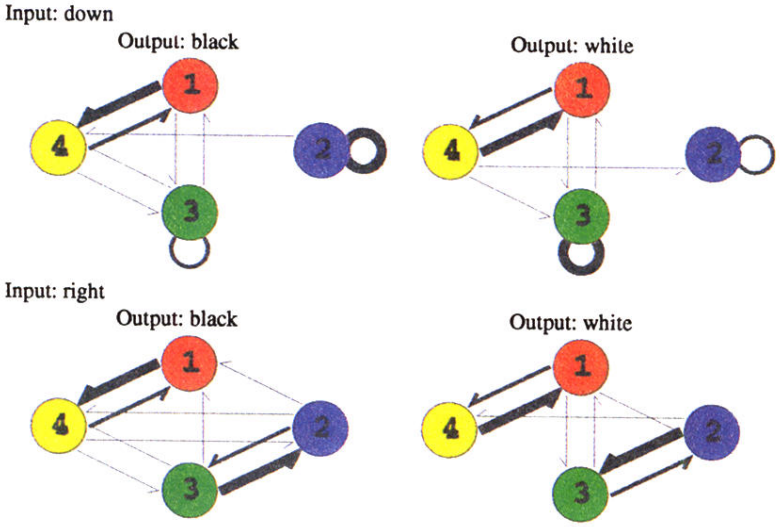

FIG. 2. The hidden Markov model after training. The thickness of the arrows indicates the transition probabilities between states for the different combinations of input (movement) and output (color). Transitions between the pairs of states (z_1, z_4) and (z_2, z_3) have only small probabilities.

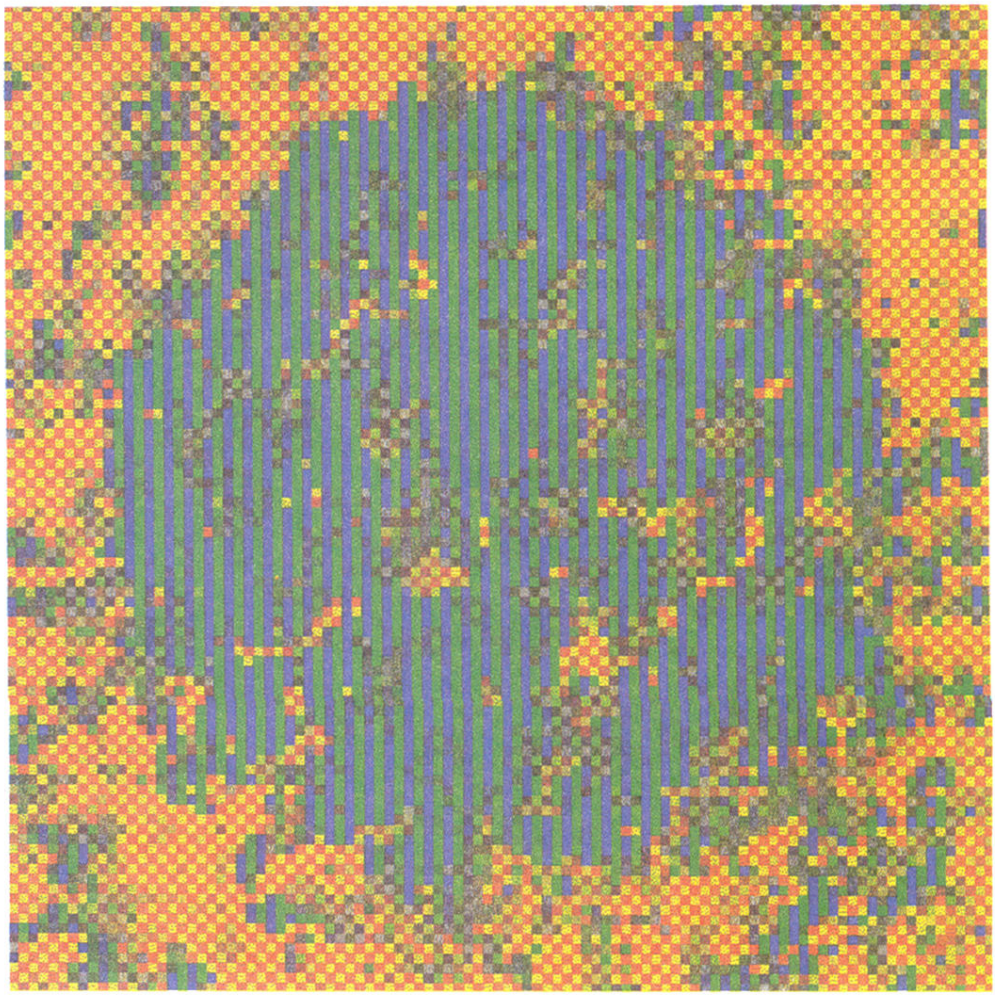

FIG. 3. Recognition of the perturbed textures. For every step on a trajectory through the square, the state distribution of the trained hidden Markov model is represented by a color code. Each color is a mixture of the pure colors of the states (see Fig. 2) with respect to their particular weight. The two texture domains can now be separated by their colors (red and yellow for the checkerboard texture and blue and green for the striped texture).