Learning Self-Organizing Maps as a Mixture Markov Models

Mustapha Lebbah, Younès Bennani and Nicoleta Rogovschi

***Abstract*—This paper describes a new algorithm to learn Self-Organizing map as Markov Mixture Models. Our model realizes an unsupervised learning using unlabelled evolutionary data sets, namely those that describe sequential data. The new formalism that we present is valid for all structure of graphical model. We use E-M (Expectation-Maximisation) standard algorithm to maximize the likelihood. The graph structure is integrated in the parameter estimation of Markov model using a neighborhood function to learn a topographic clustering of not i.i.d data set. The new approach provides a self-organizing Markov model using an original learning algorithm.**

***Index Terms*—Self-Organizing clustering, Markov Models, Sequential data, Expectation-Maximisation, graphical model**

I. Introduction

Since many years, temporal and spatial sequences have been the subject of investigation in many fields, such as statistics, pattern recognition, web mining, and bioinformatic. The easiest way to treat sequential data would be simply to ignore the sequential aspects and treat the observations as independent and identically distribution (i.i.d) in the first stage. For many applications, the i.i.d assumption will be a poor one. Often in many application the treatment is decomposed in two steps; the first one is the clustering task with i.i.d assumption. In second stage the result of clustering is used to learn a probabilistic model by relaxing the i.i.d. assumption, and one of the simplest ways to do this is to consider a Markov model.

Hidden Markov Models (HMMs) are the most well-known and practically used extension of Markov model. They offer a solution to this problem introducing, for each state, an underlying stochastic process that is not known (hidden) but could be inferred through the observations it generates. In fact the probabilistic graphical modelling motivates different graphical structures based on the HMM [1], [2]. Another variant of the HMM worthy of mention is the factorial hidden Markov model [3], in which there are multiple independent Markov chains of latent variables, and the distribution of the observed variable at a given time step is conditional on the states of all of the corresponding latent variables at that same time step. Many related models, such as hybrids of HMMs with artificial neural networks [4], [5], [6]. Clearly, there are many possible probabilistic structures that can be constructed according to the needs of particular applications. Graphical models provide a general formalism for motivating, describing, and analysing such structures. Therefore, it will be very important to have algorithms able to infer from a data set of sequences not only the probability distributions but also the topological structure of the model, i.e., the number of states and the transitions interconnecting them. Unfortunately, this task is very difficult and only partial solutions are today available [7], [8], [9]. In order to overcome the limitations of HMMs, in [9] the author proposes a novel and an original machine learning paradigm, which is titled topological HMM, that embeds the nodes of an HMM state transition graph in Euclidian space. This approach models the local structure of HMM and extract their shape by defining a unit of information as a shape formed by a group of symbols of a sequence.

Others attempts have been made for combining HMMs and SOMs (Sel-Organizing Map of Kohonen) to form hybrid models that contain the clustering power of SOM with the sequential time series aspect of HMMs [6]. In many of these hybrid architectures, SOM models are used as front-end processors for vector quantization, and HMMs are then used in higher processing stages. In [10], [11], a vector sequence is associated with a node of SOM using DTW (dynamic time warping) model. Others works exist and differ in the manner of combination [12], [13], [14]. In [14] the authors propose an original combined model which is the offspring of a crossover between the SOM algorithm and the HMM theory. The model's core consists of a novel unified/hybrid SOM-HMM algorithm where each cell of SOM map presents an HMM. The model is coupled with a sequence data training method, that blends together the SOM unsupervised learning and the HMM dynamic programming algorithms. Of course, there is a lot more litterature on HMMs and their applications than can be covered here, but this survey wants to be representative of the issues addressed here. However, in the other the organization process are not integrated explicitly in HMM approach.

The aim of this paper is to built a new model for automating and self-organizing the construction of a statistical generative model of a data set of spatial sequences. In our model 3M-SOM (Self-Organizing Maps as a Mixture Markov Models), we consider that we have one Markov chain forming a grid. The generation of the observed variable at a given time step is conditional on the neighborhood states at that same time step. Thus, a high proximity implies a high probability to contribute to generation. This proximity is quantified using neighborhood function. The same principle is used by Kohonen algorithm for i.i.d data set [15]. In our case we focus about not i.i.d observations. We use Expectation-Maximization (EM) algorithm to maximize the likelihood. The formalism that we present is valid for all structure of graph model. In our case we prefer to define the HMM architecture as map (grid).

This paper is organized as follows. In section II we present the

Mustapha Lebbah,Younès Bennani and Nicoleta Rogovschi are with LIPN-UMR 7030 - CNRS, Université Paris 13. 99, av. J-B Clément F-93430 Villetaneuse e-mail: firstname.second-name@lipn.univ-paris13.fr.

model we propose 3M-SOM. In section III we discuss the self-organizing process integrated in HMM. Finally, conclusions and some future works are provided.

II. Self-organizing Markov Model

We assume that the HMM architecture is a lattice $\mathcal{C}$, which has a discrete topology (discrete output space) defined by an undirect graph. Usually, this graph is a regular grid in one or two dimensions. We denote the number of cells (nodes,state) in $\mathcal{C}$ as K. For each pair of cells (c,r) on the graph, the distance $\delta(c,r)$ is defined as the length of the shortest chain linking cells r and c. The architecture of 3M-SOM model is inspired from probabilistic topographic clustering of i.i.d observations using a self-organizing map model of Kohonen [16], [17], [18].

A. Mixture model and Self-Organizing

We assume that each element $\mathbf{x}_n$ of sequence observation $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n, ..., \mathbf{x}_N\}$ is generated by the following process: We start by associating to each cell (state) $c \in \mathcal{C}$ a probability $p(\mathbf{x}_n/c)$ where $\mathbf{x}_n$ is a vector in the data space. Next, we pick a cell c^* from $\mathcal{C}$ according to the prior probability $p(c^*)$. For each cell c^*, we select an associated cell $c \in \mathcal{C}$ following the conditional probability $p(c/c^*)$. All cells $c \in \mathcal{C}$ contribute to the generation of an element $\mathbf{x}_n$ with $p(\mathbf{x}_n/c)$ according to the proximity to c^* described by the probability $p(c/c^*)$. Thus, a high proximity to c^* implies a high probability $p(c/c^*)$, and therefore the contribution of state c to the generation of $\mathbf{x}_n$ is high.

Let us introduce a K-dimensional binary random variable as latent variable $\mathbf{z}_n$ and $\mathbf{z}_n^*$ having a 1-of-K representation in which a particular element z_{nk} and z_{nk}^* is equal to 1 and all other elements are equal to 0. Each component z_{nk}^* and z_{nk} indicate a couple of state responsible for the generation of an element of the observation. Using this notation we can rewrite:

$$p(\mathbf{x}_n/c) \equiv p(\mathbf{x}_n/z_{nc} = 1) \equiv p(\mathbf{x}_n/\mathbf{z}_n)$$

and

$$p(c/c^*) = p(z_{nc} = 1/z_{nc^*}^* = 1) \equiv p(z_{nc}/z_{nc^*}^*) \equiv p(\mathbf{z}_n/\mathbf{z}_n^*)$$

is assumed to be known. To introduce the self-organizing process in the mixture model learning, we assume that $p(z_{nc}/z_{nc^*}^*)$ can be defined as:

$$p(z_{nc}/z_{nc^*}^*) = \frac{\mathcal{K}^T(\delta(c,c^*))}{\sum_{r\in\mathcal{C}} \mathcal{K}^T(\delta(r,c^*))},$$

where $\mathcal{K}^T$ is a neighbourhood function depending on the parameter T (called temperature): $\mathcal{K}^T(\delta) = \mathcal{K}(\delta/T)$, where $\mathcal{K}$ is a particular kernel function which is positive and symmetric ($\lim_{|x|\to\infty} \mathcal{K}(x) = 0$). Thus $\mathcal{K}$ defines for each state $z_{nc^*}^*$ a neighbourhood region in the graph $\mathcal{C}$. The parameter T allows the control of the size of the neighbourhood influencing a given cell on the map $\mathcal{C}$. As with the Kohonen algorithm for i.i.d observations, we decrease the value of T between two values T_{max} and T_{min}.

For the better understanding we have used similar notations as in the book [19, chap. 13]. We denote the set of all latent variables by $\mathbf{Z}^*$ and $\mathbf{Z}$, with a corresponding row $\mathbf{z}_n^*$ and $\mathbf{z}_n$ associated to each sequence element $\mathbf{x}_n$. Now assume that, for each sequence observation in X, corresponds the coupe of latent variable $\mathbf{Z}$ and $\mathbf{Z}^*$. We denote by $\{\mathbf{X}, \mathbf{Z}, \mathbf{Z}^*\}$ the complete data set, and we refer to the observed data $\mathbf{X}$ as incomplete.

The set of all model parameters is denoted by θ, the likelihood function is obtained from the joint distribution by marginalizing over the latent variables $\mathbf{Z}^*$ and $\mathbf{Z}$

$$p(\mathbf{X};\theta) = \sum_{\mathbf{Z}^*} \sum_{\mathbf{Z}} p(\mathbf{X}, \mathbf{Z}, \mathbf{Z}^*; \theta) \tag{1}$$

Because the joint distribution $p(\mathbf{X}, \mathbf{Z}, \mathbf{Z}^*; \theta)$ does not factorize over n, we cannot treat each of the summations over $\mathbf{z}_n^*$ and $\mathbf{z}_n$ independently.

An important concept for probability distributions over multiple variables is that of conditional independence [20]. We assume that the conditional distribution of $\mathbf{X}$, given $\mathbf{Z}^*$ and $\mathbf{Z}$, is such that it does not depend on the value of $\mathbf{Z}^*$. Often this assumtion is used for graphical model, so that $p(\mathbf{X}/\mathbf{Z}, \mathbf{Z}^*) = p(\mathbf{X}/\mathbf{Z})$. Thus the joint distribution of the sequence observations is equal to:

$$p(\mathbf{X}, \mathbf{Z}^*, \mathbf{Z}) = p(\mathbf{Z}^*)p(\mathbf{Z}/\mathbf{Z}^*)p(\mathbf{X}/\mathbf{Z})$$

thus we can rewrite the marginal distribution as

$$p(\mathbf{X};\theta) = \sum_{\mathbf{Z}^*} p(\mathbf{Z}^*) \sum_{\mathbf{Z}} p(\mathbf{Z}/\mathbf{Z}^*)p(\mathbf{X}/\mathbf{Z}) \tag{2}$$

We note that

$$p(\mathbf{X}/\mathbf{Z}^*) = \sum_{\mathbf{Z}} p(\mathbf{Z}/\mathbf{Z}^*)p(\mathbf{X}/\mathbf{Z}) \tag{3}$$

B. Cost function and optimization

Considering a map $\mathcal{C}$ as Markov model, we allow the probability distribution of $\mathbf{z}_n^*$ to depend on the state of the previous latent variable $\mathbf{z}_{n-1}^*$ through a conditional distribution $p(\mathbf{z}_n^*|\mathbf{z}_{n-1}^*)$. Because the latent variables are K-dimensional binary variables, this conditional distribution corresponds to a table of probabilities that we denote by $\mathbf{A}$. The elements of $\mathbf{A}$ are known as transition probabilities denoted by $A_{jk} = p(\mathbf{z}_{nk}^* = 1/\mathbf{z}_{n-1,j}^* = 1)$, with $\sum_k A_{jk} = 1$. So the matrix $\mathbf{A}$ has maximum of $K(K-1)$ independent parameters. In our case the number of transitions are limited by the grid (map). We can then write the conditional distribution explicitly in the form

$$p(\mathbf{z}_n^*/\mathbf{z}_{n-1}^*, \mathbf{A}) = \sum_{k=1}^{K} \sum_{j=1}^{K} A_{jk}^{z_{n-1,j}^* z_{nk}^*}$$

All of the conditional distributions governing the latent variables share the same parameters $\mathbf{A}$.

The initial latent state $\mathbf{z}_1^*$ is special in that it does not have a parent cell, and so it has a marginal distribution $p(\mathbf{z}_1^*)$ represented by a vector of probabilities π with elements $\pi_k = p(\mathbf{z}_{1k}^* = 1)$, so that $p(\mathbf{z}_1^*|\pi) = \prod_{k=1}^{K} \pi^{z_{1k}^*}$, where $\sum_k \pi_k = 1$.

The model parameters are completed by defining the conditional distributions of the observed variables $p(\mathbf{x}_n/\mathbf{z}_n;\phi)$, where ϕ is a set of parameters governing the distribution which is known as emission probabilities in HMM model.

Because $\mathbf{x}_n$ is observed, the distribution $p(\mathbf{x}_n/\mathbf{z}_n,\phi)$ consists, for a given value of ϕ, of a vector of K numbers corresponding to the K possible states of the binary vector $\mathbf{z}_n$. We can represent the emission probabilities in the form

$$p(\mathbf{x}_n/\mathbf{z}_n;\phi)=\prod_{k=1}^{K}p(\mathbf{x}_n;\phi_k)^{z_{nk}}$$

The joint probability distribution over sequence observed variables and both latent $\mathbf{Z}$ and $\mathbf{Z}^*$ is then given by

$$p(\mathbf{X},\mathbf{Z}^*,\mathbf{Z};\theta)=p(\mathbf{Z}^*;\mathbf{A})\times p(\mathbf{Z}/\mathbf{Z}^*)\times p(\mathbf{X}/\mathbf{Z};\phi)$$

$$\begin{aligned}p(\mathbf{X},\mathbf{Z}^*,\mathbf{Z};\theta) &= \left[p(\mathbf{z}_1^*|\pi)\prod_{n=2}^{N}p(\mathbf{z}_n^*/\mathbf{z}_{n-1}^*;\mathbf{A})\right]\\ &\times \left[\prod_{i=1}^{N}p(\mathbf{z}_i/\mathbf{z}_i^*)\right]\\ &\times \left[\prod_{m=1}^{N}p(\mathbf{x}_m/\mathbf{z}_m;\phi)\right] \qquad (4)\end{aligned}$$

$\theta=\{\pi,\mathbf{A},\phi\}$ denotes the set of parameters governing the model. Most of our discussion of the self organizing Markov model will be independent of the particular choice of the emission probabilities. It's not obvious to maximize the likelihood function, because we obtain complex expressions with no closed-form solutions. Hence, we use the expectation maximization algorithm to find parameters for maximizing the likelihood function. EM algorithm starts with some initial selection for the model parameters, which we denote by θ^{old}. In the E step, we take these parameter values and find the posterior distribution of the latent variables $p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X},\theta^{old})$. We then use this posterior distribution to evaluate the expectation of the logarithm of the complete-sequence data likelihood function (4), as a function of the parameters θ, to give the function $Q(\theta,\theta^{old})$ defined by:

$$Q(\theta,\theta^{old})=\sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})\ln p(\mathbf{X},\mathbf{Z}^*,\mathbf{Z};\theta)$$

$$\begin{aligned}Q(\theta,\theta^{old}) &= \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})\ln p(\mathbf{Z}^*;\pi,\mathbf{A})\\ &+ \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})\ln p(\mathbf{X}/\mathbf{Z};\phi)\\ &+ \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})\ln p(\mathbf{Z}/\mathbf{Z}^*)\end{aligned}$$

$$\begin{aligned}Q(\theta,\theta^{old}) &= Q_1(\pi,\theta^{old})+Q_2(\mathbf{A},\theta^{old})\\ &+ Q_3(\phi,\theta^{old})+Q_4 \qquad (5)\end{aligned}$$

where

$$Q_1(\pi,\theta^{old})=\sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{k=1}^{K}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})z_{1k}^*\ln\pi_k$$

$$Q_2(\mathbf{A},\theta^{old})=\sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{n=2}^{N}\sum_{k=1}^{K}\sum_{j=1}^{K}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})z_{n-1,j}^*z_n^*\ln(A_{jk})$$

$$Q_3(\phi,\theta^{old})=\sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{n=1}^{N}\sum_{k=1}^{K}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})z_{nk}\ln\left(p(\mathbf{x}_n;\phi_k)\right)$$

$$Q_4=\sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})\ln p(\mathbf{Z}/\mathbf{Z}^*)$$

At this point, we introduce some notation. We will use $\gamma(\mathbf{z}_n^*,\mathbf{z}_n)$ to denote the marginal posterior distribution of a latent variable $\mathbf{z}_n^*$ and $\mathbf{z}_n$, and $\xi(\mathbf{z}_{n-1}^*,\mathbf{z}_n^*)=p(\mathbf{z}_{n-1}^*,\mathbf{z}_n^*/\mathbf{X},\theta^{old})$ to denote the joint posterior distribution of successive latent variables, so that

$$\gamma(\mathbf{z}_n^*,\mathbf{z}_n)=p(\mathbf{z}_n^*,\mathbf{z}_n|\mathbf{X};\theta^{old})$$

thus

$$\gamma(\mathbf{z}_n^*)=\sum_{\mathbf{z}}p(\mathbf{z}_n^*,\mathbf{z}_n|\mathbf{X};\theta^{old})$$

$$\gamma(\mathbf{z}_n)=\sum_{\mathbf{z}^*}p(\mathbf{z}_n^*,\mathbf{z}_n|\mathbf{X};\theta^{old})$$

$$\begin{aligned}\gamma(z_{nk}^*) &= \mathbf{E}[z_{nk}^*]\\ &= \sum_{\mathbf{z}^*}\sum_{\mathbf{z}}\gamma(\mathbf{z}_n^*,\mathbf{z}_n)z_{nk}^*\\ &= \sum_{\mathbf{z}^*}\gamma(\mathbf{z}_n^*)z_{nk}^*\end{aligned}$$

We observe that the objective function (5) $Q(\theta,\theta^{old})$ is defined as a sum of four terms. The first term $Q_1(\pi,\theta^{old})$ depends on initial probabilities; the second term $Q_2(\mathbf{A},\theta^{old})$ depends on transition probabilities $\mathbf{A}$; the third term $Q_3(\phi,\theta^{old})$ depends on ϕ, and the forth term is constant. Maximizing $Q(\theta,\theta^{old})$ with respect to $\theta=\{\pi,\mathbf{A},\phi\}$ can be performed separately.

1) Maximization of $Q_1(\pi,\theta^{old})$: Initial probabilities:

$$\begin{aligned}Q_1(\pi,\theta^{old}) &= \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{k=1}^{K}p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{old})z_{1k}^*\ln\pi_k\\ &= \sum_{\mathbf{Z}^*}\sum_{k=1}^{K}p(\mathbf{Z}^*/\mathbf{X};\theta^{old})z_{1k}^*\ln\pi_k\\ &= \sum_{k=1}^{K}\gamma(z_{1k}^*)\ln\pi_k\end{aligned}$$

The update parameter is computed as follows:

$$\pi_k = \frac{\gamma(z_{1k}^*)}{\sum_{j=1}^{K}\gamma(z_{1j}^*)} \qquad (6)$$

2) Maximization of $Q_2(\mathbf{A}, \theta^{old})$: Probability transitions :

$$Q_2(\mathbf{A}, \theta^{old}) =$$
$$\sum_{\mathbf{Z}^*} \sum_{\mathbf{Z}} \sum_{n=2}^{N} \sum_{k=1}^{K} \sum_{j=1}^{K} p(\mathbf{Z}^*, \mathbf{Z}/\mathbf{X}; \theta^{old}) z^*_{n-1,j} z^*_n \ln(A_{jk})$$
$$= \sum_{\mathbf{Z}^*} \sum_{n=2}^{N} \sum_{k=1}^{K} \sum_{j=1}^{K} p(\mathbf{Z}^*/\mathbf{X}; \theta^{old}) z^*_{n-1,j} z^*_n \ln(A_{jk})$$
$$= \sum_{n=2}^{N} \sum_{k=1}^{K} \sum_{j=1}^{K} \xi(z^*_{n-1,j} z^*_n) \ln(A_{jk})$$

The update parameter is computed as follows:

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z^*_{n-1,j}, z^*_{nk})}{\sum_{l=1}^{K} \sum_{n=2}^{N} \xi(z^*_{n-1,j}, z^*_{nl})} \quad (7)$$

where

$$\xi(z^*_{n-1,j}, z^*_{n,k}) = \mathbf{E}[z^*_{n-1,j} z^*_{nk}] = \sum_{\mathbf{z}^*} \gamma(\mathbf{z}^*) z^*_{n-1,j} z^*_{n,k}$$

3) Maximization of $Q_3(\phi, \theta^{old})$: Emission probabilities:

$$Q_3(\phi, \theta^{old}) =$$
$$\sum_{\mathbf{Z}^*} \sum_{\mathbf{Z}} \sum_{n=1}^{N} \sum_{k=1}^{K} p(\mathbf{Z}^*, \mathbf{Z}/\mathbf{X}; \theta^{old}) z_{nk} \ln p(\mathbf{x}_n; \phi_k)$$
$$= \sum_{\mathbf{Z}} \sum_{n=1}^{N} \sum_{k=1}^{K} p(\mathbf{Z}/\mathbf{X}; \theta^{old}) z_{nk} \ln p(\mathbf{x}_n; \phi_k)$$
$$= \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) \ln p(\mathbf{x}_n; \phi_k)$$

In the case of spherical Gaussian emission densities we have $p(\mathbf{x}/\phi_k) = \mathcal{N}(\mathbf{x}; \mathbf{w}_k, \sigma_k)$, defined by its "mean" $\mathbf{w}_k$, which have the same dimension as input data, and its covariance matrix, defined by $\sigma_k^2 \mathbf{I}$ where σ_k is the standard deviation and $\mathbf{I}$ is the identity matrix. The maximization of the function $Q_3(\phi, \theta^{old})$ provides:

$$\mathbf{w}_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^{N} \gamma(z_{nk})} \quad (8)$$

$$\sigma_k^2 = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) ||\mathbf{x}_n - \mathbf{w}_k||^2}{d \sum_{n=1}^{N} \gamma(z_{nk})} \quad (9)$$

where d is the dimension of the element $\mathbf{x}$.

The EM algorithm requires initial values for the parameters of the emission distribution. One way to set these is first to treat the data initially as i.i.d. and fit the emission density by maximum likelihood, and then use the resulting values to initialize the parameters for EM.

C. The forward-backward algorithm: E-step

Next we seek an efficient procedure for evaluating the quantities $\gamma(\mathbf{z}^*_n)$, $\gamma(\mathbf{z}_n)$ and $\xi(\mathbf{z}^*_{n-1}, \mathbf{z}^*_n)$, corresponding to the E step of the EM algorithm. In the particular context of the hidden Markov model, this is known as the forward-backward algorithm [21], or the Baum-Welch algorithm [22], [23]. In our case it can be renamed topological forward-backward algorithm, because we use the graph structure to organize the sequential data. Some formula are similar if we don't use the graph structure. We will use the notations $\alpha(z^*_{nk})$ and $\alpha(z_{nk})$ to denote the value of $\alpha(\mathbf{z}^*)$ and $\alpha(\mathbf{z})$ when $z^*_{nk} = 1$, $z_{nk} = 1$ with an analogous notations of β.

$$\gamma(\mathbf{z}^*_n) = p(\mathbf{z}^*_n/\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{z}^*_n) p(\mathbf{z}^*_n)}{p(\mathbf{X})}$$
$$= \frac{p(\mathbf{x}_1, ..., \mathbf{x}_n, \mathbf{z}^*_n) p(\mathbf{x}_{n+1}, ..., \mathbf{x}_N/\mathbf{z}^*_n)}{p(\mathbf{X})}$$
$$\gamma(\mathbf{z}^*_n) = \frac{\alpha(\mathbf{z}^*_n) \beta(\mathbf{z}^*_n)}{p(\mathbf{X})}$$

Using the similar decomposition we obtain

$$\gamma(\mathbf{z}_n) = \frac{\alpha(\mathbf{z}_n) \beta(\mathbf{z}_n)}{p(\mathbf{X})}$$

The values of $\alpha(\mathbf{z}^*_n)$ and $\alpha(\mathbf{z}_n)$ are calculated by forward recursion as follows:

$$\alpha(\mathbf{z}^*_n) = \left[\sum_{\mathbf{z}} p(\mathbf{x}_n/\mathbf{z}_n) p(\mathbf{z}_n/\mathbf{z}^*_n) \right] \times \sum_{\mathbf{z}^*_{n-1}} \alpha(\mathbf{z}^*_{n-1}) p(\mathbf{z}^*_n|\mathbf{z}^*_{n-1}) \quad (10)$$

and

$$\alpha(\mathbf{z}_n) = p(\mathbf{x}_n|\mathbf{z}_n) \sum_{\mathbf{z}^*_n} p(\mathbf{z}_n/\mathbf{z}^*_n) \quad (11)$$
$$\left[\sum_{\mathbf{z}^*_{n-1}} \alpha(\mathbf{z}^*_{n-1}) p(\mathbf{z}^*_n|\mathbf{z}^*_{n-1}) \sum_{\mathbf{z}_{n-1}} p(\mathbf{z}_{n-1}|\mathbf{z}^*_{n-1}) \right]$$

where $p(\mathbf{z}_n/\mathbf{z}^*_n) = p(z_{nc} = 1/z^*_{nc^*} = 1) = \frac{K^T(\delta(c,c^*))}{\sum_{r \in \mathcal{C}} K^T(\delta(r,c^*))}$. To start this recursion, we need an initial condition that is given by

$$\alpha(\mathbf{z}^*_1) = p(\mathbf{x}_1, \mathbf{z}^*_1) = p(\mathbf{z}^*_1) \left[\sum_{\mathbf{z}_1} p(\mathbf{x}_1/\mathbf{z}_1) p(\mathbf{z}_1/\mathbf{z}^*_1) \right]$$

$$\alpha(\mathbf{z}_1) = p(\mathbf{x}_1, \mathbf{z}_1) = p(\mathbf{x}_1/\mathbf{z}_1) \left[\sum_{\mathbf{z}^*_1} p(\mathbf{z}^*_1) p(\mathbf{z}_1/\mathbf{z}^*_1) \right]$$

The value of $\beta(\mathbf{z}^*_n)$, are calculated by backward recursion as follows:

$$\beta(\mathbf{z}^*_n) = \sum_{\mathbf{z}^*_{n+1}} \beta(\mathbf{z}^*_{n+1}) p(\mathbf{x}_{n+1}/\mathbf{z}^*_{n+1}) p(\mathbf{z}^*_{n+1}/\mathbf{z}^*_n) \quad (12)$$

$$\beta(\mathbf{z}_n) = \frac{1}{p(\mathbf{z}_n)} \sum_{\mathbf{z}^*_n} p(\mathbf{z}^*_n) p(\mathbf{z}_n/\mathbf{z}^*_n) \sum_{\mathbf{z}_{n+1}} \sum_{\mathbf{z}^*_{n+1}} \quad (13)$$
$$p(\mathbf{z}_{n+1}/\mathbf{z}^*_{n+1}) \beta(\mathbf{z}^*_{n+1}) p(\mathbf{x}_{n+1}/\mathbf{z}^*_{n+1}) p(\mathbf{z}^*_{n+1}/\mathbf{z}^*_n)$$

where

$$p(\mathbf{x}_{n+1}/\mathbf{z}^*_{n+1}) = \left[\sum_{\mathbf{z}} p(\mathbf{x}_{n+1}/\mathbf{z}_{n+1}) p(\mathbf{z}_{n+1}/\mathbf{z}^*_{n+1}) \right]$$

$$p(\mathbf{z}_n) = \sum_{\mathbf{z}_n^*} p(\mathbf{z}_n^*) p(\mathbf{z}_n / \mathbf{z}_n^*)$$

and

$$\begin{aligned} p(\mathbf{z}_{n+1}/\mathbf{z}_{n+1}^*) &= p(z_{n+1,c} = 1 / z_{n+1,c^*}^* = 1) \\ &= \frac{K^T(\delta(c,c^*))}{\sum_{r \in \mathcal{C}} K^T(\delta(r,c^*))} \end{aligned}$$

Again we need a starting condition for the recursion, a value for $\beta(\mathbf{z}_N^*) = 1$ and $\beta(\mathbf{z}_N) = 1$. This can be obtained by setting $n = N$ in (expression 10).

Next we consider the evaluation of the quantities $\xi(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*)$ which correspond to the values of the conditional probabilities $p(\mathbf{z}_{n-1}^*, \mathbf{z}_n^* / \mathbf{X})$ for each of the $K \times K$ settings for $(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*)$. Using the applying Bayes theorem, we obtain

$$\begin{aligned} \xi(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*) &= p(\mathbf{z}_{n-1}^*, \mathbf{z}_n^* / \mathbf{X}) \\ &= \frac{p(\mathbf{X}/\mathbf{z}_{n-1}^*, \mathbf{z}_n^*) p(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*)}{p(\mathbf{X})} \\ \xi(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*) &= \frac{\alpha(\mathbf{z}_{n-1}^*) \left[\sum_{\mathbf{z}} p(\mathbf{x}_n / \mathbf{z}_n) p(\mathbf{z}_n / \mathbf{z}_n^*)\right]}{p(\mathbf{X})} \\ &\times \frac{p(\mathbf{z}_n^* / \mathbf{z}_{n-1}^*) \beta(\mathbf{z}_n^*)}{p(\mathbf{X})} \end{aligned}$$

If we sum both sides of $\alpha(\mathbf{z}^*)$ over $\mathbf{z}_N$, we obtain $p(\mathbf{X}) = \sum_{\mathbf{z}_N} \alpha(\mathbf{z}_N)$. Then we compute the forward α recursion and the backward β recursion and use the results to evaluate γ and $\xi(\mathbf{z}_{n-1}^*, \mathbf{z}_n^*)$. We use these results to compute a new parameter model θ using the M-step equations (6, 7, 8, 9). These both steps are repeated until some convergence criterion is satisfied.

III. Discussion about topological Markov model organization

The 3M-SOM model allows us to estimate the parameters maximizing the log-likelihood function for a fixed T. As in the topological clustering algorithm, we have to decrease the value of T between two values T_{max} and T_{min}, to control the size of the neighbourhood influencing a given state of HMM on the graph (grid) and at same time. For each T value, we get a likelihood function Q^T, and therefore the expression varies with T. When decreasing T, the model of 3M-SOM will be defined in the following way:

- The first step corresponds to high T values. In this case, the influencing neighbourhood of each state $\mathbf{z}^*$ on the HMM graph (grid) is important and corresponds to higher values of $K^T(\delta(c,r))$. Formulas use a high number of state and hence high number of observations to estimate model parameters. This step provides the topological order of Markov model.
- The second step corresponds to small T values. The number of observations in formulas is limited. Therefore, the adaptation is very local. The parameters are accurately computed from the local density of the data. In this case we can consider that we converge to traditional HMM (without using neighborhood). Recall that clustering based on mixture model for i.i.d. observations is a special case of the HMM [19, chap 9].

IV. Conclusion

In this paper, we presented an original model that could be applied to more advanced/complex data set (not i.i.d observations, time series). We provides here the mathematical formulation of our model. We present one way to estimate the parameter using EM algorithm with Baum-Welch algorithm. Visualization techniques and refined graphic displays can be developed to illustrate the power of 3M-SOM to to explore the not i.i.d data. As has been stressed, the 3M-SOM unsupervised topographic learning algorithm is purely batch learning. An extension to an on-line mode version is quite straightforward. Finally, providing an equivalent to the 3M-SOM for applications requiring Bernoulli emission probability density functions should be interesting task.

References

[1] Yoshua Bengio and Paolo Frasconi. An input output hmm architecture. In *NIPS*, pages 427–434, 1994.

[2] Samy Bengio and Yoshua Bengio. An EM algorithm for asynchronous input/output hidden Markov models. In L. Xu, editor, *International Conference On Neural Information Processing*, pages 328–334, Hong-Kong, 1996.

[3] Zoubin Ghahramani and Michael I. Jordan. Factorial hidden markov models. *Mach. Learn.*, 29(2-3):245–273, 1997.

[4] Herve A. Bourlard and Nelson Morgan. *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, Norwell, MA, USA, 1993.

[5] Marina Meila and Michael I. Jordan. Learning fine motion by markov mixtures of experts. Technical report, Cambridge, MA, USA, 1995.

[6] Catherine Recanati, Nicoleta Rogovschi, and Younès Bennani. The structure of verbal sequences analyzed with unsupervised learning techniques. In *LTC'07, Language Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics. October 5-7, 2007, Poznan, Poland*, 2007.

[7] Djamel Bouchaffra and Jun Tan. Structural hidden markov model and its application in automotive industry. In *ICEIS (2)*, pages 155–164, 2003.

[8] D. Bouchaffra and J. Tan. Structural hidden markov models: An application to handwritten numeral recognition. *Intell. Data Anal.*, 10(1):67–79, 2006.

[9] Djamel Bouchaffra. Embedding hmm's-based models in a euclidean space: The topological hidden markov models. In *ICPR08*, pages 1–4, 2008.

[10] Panu Somervuo and Teuvo Kohonen. Self-organizing maps and learning vector quantization forfeature sequences. *Neural Process. Lett.*, 10(2):151–159, 1999.

[11] Panu Somervuo. Competing hidden markov models on the self-organizing map. *Neural Networks, IEEE - INNS - ENNS International Joint Conference on*, 3:3169, 2000.

[12] Khalid Benabdeslem Arnaud Zeboulon, Youns Bennani. Hybrid connectionist approach for knowledge discovery from web navigation patterns. In *ACS/IEEE International Conference on Computer Systems and Applications*, pages 118–122, 2003.

[13] Christos Ferles and Andreas Stafylopatis. A hybrid self-organizing model for sequence analysis. In *ICTAI '08: Proceedings of the 2008 20th IEEE International Conference on Tools with Artificial Intelligence*, pages 105–112, Washington, DC, USA, 2008. IEEE Computer Society.

[14] C. Ferles and A. Stafylopatis. Sequence clustering with the self-organizing hidden markov model map. *BioInformatics and BioEngineering, 2008. BIBE 2008. 8th IEEE International Conference on*, pages 1–7, Oct. 2008.

[15] T. Kohonen. *Self-organizing Maps*. Springer Berlin, 2001.

[16] Fatiha Anouar, Fouad Badran, and Sylvie Thiria. Self-organizing map, a probabilistic approach. In *Proceedings of WSOM'97-Workshop on Self-Organizing Maps, Espoo, Finland June 4-6*, pages 339–344, 1997.

[17] C. M. Bishop, M. Svensén, and C. K. I.Williams. GTM: The generative topographic mapping. *Neural Comput*, 10(1):215–234, 1998.

[18] Mustapha Lebbah, Nicoleta Rogovschi, and Younés Bennani. Besom : Bernoulli on self organizing map. In *International Joint Conferences on Neural Networks. IJCNN 2007, Orlando, Florida, August 12-17*, 2007.

[19] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), 2006.

[20] S. P Luttrel. A bayesian analysis of self-organizing maps. *Neural Computing*, 6:767 – 794, 1994.

[21] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[22] L. E. Baum. An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes. *Inequalities*, 3:1–8, 1972.

[23] Edoardo M Airoldi. Getting started in probabilistic graphical models. *PLoS Comput Biol*, 3(12):e252, 12 2007.