International Journal of Computational Intelligence and Applications
Vol. 14, No. 2 (2015) 1550007 (29 pages)

DOI: 10.1142/S1469026815500078

Imperial College Press
www.icpress.co.uk

Probabilistic Self-Organizing Map for Clustering and Visualizing non-i.i.d Data

Mustapha Lebbah[*,‡], Rakia Jaziri[*,§], Younès Bennani[*,¶] and Jean-Hugues Chenot[†,‖]

[*]*Université Paris 13, Sorbonne Paris cité*
LIPN, CNRS UMR 7030, 99, avenue Jean-Baptiste Clément 93430 Villetaneuse, France

[†]*Institut National de l'Audiovisuel*
4, av. de l'Europe94 366 Bry-sur-Marne
[‡]*mustapha.lebbah@univ-paris13.fr*
[§]*rakia.jaziri@univ-paris13.fr*
[¶]*bennani@lipn.univ-paris13.fr*
[‖]*Chenot@ina.fr*

Received 20 December 2013
Revised 12 September 2014
Published 23 June 2015

We present a generative approach to train a new probabilistic self-organizing map (PrSOMS) for dependent and nonidentically distributed data sets. Our model defines a low-dimensional manifold allowing friendly visualizations. To yield the topology preserving maps, our model has the SOM like learning behavior with the advantages of probabilistic models. This new paradigm uses hidden Markov models (HMM) formalism and introduces relationships between the states. This allows us to take advantage of all the known classical views associated to topographic map. The objective function optimization has a clear interpretation, which allows us to propose expectation-maximization (EM) algorithm, based on the forward–backward algorithm, to train the model. We demonstrate our approach on two data sets: The real-world data issued from the "French National Audiovisual Institute" and handwriting data captured using a WACOM tablet.

Keywords: Mixture model; self-organizing map; hidden Markov model; visualization; EM algorithm; non-i.i.d data.

1. Introduction

Data visualization and clustering are important steps in the exploratory phase of data analysis, which becomes more difficult when it involves sequential data or non-independent identically distributed (i.i.d). Little research has been devoted to methods of clustering and visualizing for exploration of the dynamics of multivariate sequential data. The self-organizing map (SOM)[1] has probably been the most popular unsupervised clustering model since its introduction. The popularity of the SOM

‡Corresponding author.

is largely a result of its algorithmic elegance and superficially easily interpretable results. The clusters are sometimes also referred to as cells, nodes, neurons or prototypes. The majority of SOM applications include visualization and browsing of large datasets. The computational results essentially depend on two priorily chosen ingredients: The metric or the probability distribution, and the grid topology of the SOM. Unfortunately, the paradigms of self-organization cannot be easily transferred to non-i.i.d data. Different approaches have been developed to embed temporal information in the SOM. Popular way to treat sequential data is to simply ignore the sequential aspects and treat the observations as i.i.d in the first stage. For many applications, the i.i.d assumption will be a poor one. In many applications, the treatment is often decomposed in two steps: The first one is the clustering task, and in the 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 hidden Markov model (HMM). A drawback of such model would be the size as at least one HMM and cell would be required for each sequence. The state space map as defined does not provide immediate classification but it reduces the dimension of the original feature data significantly to simplify subsequent classification. 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 analyzing such structures.

The merits of this work is not to demonstrate that our approach is better than other approaches presented in the related works section, but to provide another original way to build a generative topographic model dedicated to non-i.i.d data. To yield the topology preserving maps, our model (probabilistic self-organizing model (PrSOMS) for sequential data) has the SOM like learning behavior with the advantages of probabilistic models, like handling of missing data values, or binary data using Bernoulli distribution or mixed data, as we proposed for i.i.d data in Refs. 2 and 3. This will allow us to take advantage of all the known classical view in which we add other specific displays for non-i.i.d data. Compared to other mixture models, the objective function optimization has a clear interpretation: It sums the data log-likelihood without penalty term as in Ref. 4.

The aim of this paper is to build a new probabilistic model for automating and self-organizing the construction of a statistical generative model of non-i.i.d data sets. The proposed model is based on the probabilistic formalism of the SOM used for i.i.d data sets.[2,5] We show that the HMMs are a special case of PrSOMS model. PrSOMS model can be defined and titled as a self-organzing HMM (SOHMM). Therefore, it consists of estimating the parameters of the model by maximizing the likelihood of the sequential data set. The learning algorithm that we propose is an application of the expectation-maximization (EM) standard algorithm. Our model must not be confused with the work presented in Ref. 6. In our model, we consider a SOM as a grid forming a discrete topology in which each cell represents a state. The word state or cell are interchangeable. The generation of the observed variable at a given time step is conditional on the neighborhood cells/states at that same time

step. Thus, a high proximity implies a high probability to contribute to the generation. This proximity is quantified using a neighborhood function. The formalism that we present is valid for all structure of the graph model.

The rest of this paper is organized as follows: Related work and existing solutions are deeply and carefully analyzed in Sec. 2. Section 3 covers the mathematical description of the PrSOM dedicated to non-i.i.d data. The experiments are presented in Sec. 4. This model is evaluated and validated on real TV broadcasts collected from 10 French channels and compared with experts, and on real handwriting data set. The paper ends with Sec. 5, which contains a general conclusion and possible future work.

2. Related Works

The SOM models for a sequence can roughly be split in three groups. The first group consist of the best match unit trace method and the SOM of competing HMMs. Both of these models generate a state sequence (a sequence of state probability distributions), which can be used as input for further processing. The most direct way is to include time-delayed versions as input or add preprocessing to capture spatial dynamics.[7,8] A variety of models exists for recurrent SOMs: The temporal Kohonen map (TKM), the recurrent SOM (RSOM), recursive SOM (RecSOM), and the SOM for structured data (SOMSD).[9–13] These methods differ in their way of internally representing the temporal context. In TKM and RSOM, each unit is equipped with a weight vector. In each cell or unit, one input item is processed at each time step in the context given by the past activations of that cell. When a new input is presented, the cells do not lose their past activity as in SOM, but the context information decays gradually. However, RSOM modifies TKM by summing the deviation of the weights vector as opposed to distances. The SOMSD presented in Ref. 14 is an extension of SOM designed to process patterns expressed as directed acyclic graphs (trees and sequences are special cases).

There are other approaches using a dynamic time warp. The DTW[15] is a dynamic programming algorithm, where a sequence of observations is compared to another reference sequence. The combination of such references to the cells of the map and the use of DTW in the assignment phase led to the DTW-SOM model.[16,17] An other group of methods includes hybrids models. The first map creates a state sequence, which is integrated and serves as the input to the second map. In these hybrid architectures, SOM models are used as front-end processors for vector quantization, and HMMs could be used in higher processing stages. In Refs. 18 and 19, 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 in a novel 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 unsupervised learning (SOM) and the HMM dynamic programming algorithms.

There are several alternatives to the SOM of which we briefly discuss two. Generative topographic mapping (GTM)[20] was designed to be a principled alternative to

the SOM. In GTM, the latent space interpretation of the SOM is made explicit, by projecting a latent manifold defined with a set a radial basis functions into the data space. These basis functions work like the neighborhood of the SOM. GTM has visualization capabilities. The GTM has been extended to model time series (GTM through time), GTM-TT[21] and structured data.[22] It explicitly accounts for the violation of the i.i.d condition, given that it is defined as a constrained HMM. The GTM-TT was defined only for the exploratory analysis of multivariate time series. The capabilities for clustering and visualization was assessed in detail in Ref. 23.

Recently, in Ref. 4 the author proposes the extension of the SOMM algorithm[24] for multivariate time series self-organizing hidden Markov models (SOHMMs). The SOHMM algorithm assumes that a time series is generated by a HMM. To yield the topology preserving maps, the SOHMM algorithm estimates the HMM parameters by the constrained EM algorithm. As with the SOMM, the author proposes to maximize the variational free-energy that sums data log-likelihood and Kullback–Leibler divergence between a normalized neighborhood function and the posterior distribution on the given data for the components.

The HMM model is the common model most widely used. HMM is very popular in the speech recognition field, and widely applied to other applications.[25–27] However, the organization process is not integrated in their approach. In order to overcome the limitations of HMMs, in Ref. 28 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 an Euclidian space. This approach models the local structure of HMM and extracts their shape by defining a unit of information as a shape formed by a group of symbols of a sequence. Another work, is the Markov random field, which defines discrete lattice among the states. Usually the lattice is a regular two-dimensional grid as in the case of the SOM.[29] Markov random fields appear naturally in problems such as image segmentation.

Therefore, it is very important to have probabilistic SOM algorithms that are able to infer from a data set of sequences not only the probability distributions but also the topological structure of the model and provide a friendly tool for visualizing and exploring data. Unfortunately, this task is very difficult and only partial solutions are available today. In this paper, we propose a PrSOM algorithm for multi-dimensional sequential data, which we call probabilistic self-organizing sequential data (PrSOMS), by assuming that the sequential data is generated in the same way as HMM. However, the manner in which PrSOMS achieves the topographic organization is different from those used in the SOHMM and GTM-TT model.

3. Self-Organizing and Mixture Models

Mixture models are used in a wide range of settings in machine learning including clustering, data visualization, dimension reduction, etc. A mixture model can be interpreted as a model, that assumes that there are K, group distributions, which

generate the data: Each distribution is selected to generate data with a probability equal to its mixing weight or prior probability and it generates data according to its component density. Marginalizing over the components, we recover the mixture model as the distribution over the data. Assigning each data item to the cluster that is most probable to have generated the data item, we associate a mixture model to the clustering paradigm. The EM algorithm[30] is a popular algorithm to fit the parameters of a mixture to given data. For a better understanding of our mixture model, we have used similar notations to those in the book (see Ref. 31, [Chap. 13]). Assume an observed vector sequence $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \ldots, \mathbf{x}_N\}$, where $\mathbf{x}_n$ is a element of sequence, and N is the length of a sequence. The learning problem is to estimate the parameters of the PrSOMS model. We assume that the PrSOMS architecture is the lattice $\mathcal{S}$ (called a map), which has a discrete topology (discrete output latent space) is defined by an undirected graph. We denote the number of cells (states) in $\mathcal{S}$ as K. For each pair of states (s,k) on the map, the distance $\delta(s, k)$ is defined as the length of the shortest chain linking cells k and s.

3.1. *Main idea*

We propose an algorithm that yields topology preserving maps of multi-dimensional sequences based on the HMMs. PrSOMS learning algorithm assumes that a sequence is generated by a constrained HMM. Each element of sequence $\mathbf{x}_n$ is generated by a couple of neighborhood states (cells) s and k. The s indicates the first cell s selected and k indicates a neighborhood cell. Learning is facilitated by introducing 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_{ns}^* and z_{nk} is equal to 1 and all other elements are equal to 0 (see Fig. 1). For each element $\mathbf{x}_n$, we start by picking a state z_{ns}^* from a map $\mathcal{S}$ according to the prior probability $p(z_{ns}^*)$. Next, we select an associated neighborhood state z_{nk} following the conditional probability $p(z_{nk}/z_{ns}^*)$ at the same time n. All states $z_{nk} \in \mathcal{S}$ contribute to the generation of an element $\mathbf{x}_n$ with $p(\mathbf{x}_n/z_{nk})$ according to the proximity to z_{ns}^* described by the probability $p(z_{nk}/z_{ns}^*)$. Thus, a high proximity to state z_{ns}^* implies a high probability, and therefore the contribution of state z_{nk} to the generation of $\mathbf{x}_n$ is high ($p(z_{ns}/z_{ns}^*)$=1).

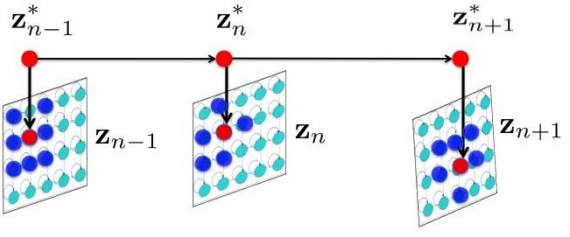


Fig. 1. A fragment of the factor graph representation for the PrSOMS model.

3.1.1. *PrSOMS parameters*

Using these notations, we can define the parameters of PrSOMS model as follows:

- The conditional distributions of the observed variables $p(\mathbf{x}_n/z_{nk}=1) \equiv p(\mathbf{x}_n/\mathbf{z}_n;\phi)$, where ϕ is a set of parameters governing the distribution which is known as emission probabilities, where $p(\mathbf{x}_n/\mathbf{z}_n;\phi) = \prod_{k=1}^{K} p(\mathbf{x}_n/\phi_k)^{z_{nk}}$. In the case of spherical Gaussian emission densities, $p(\mathbf{x}_n/\phi_k) = N(\mathbf{x}_n;\mathbf{w}_k,\sigma_k)$, centered on $(\mathbf{w}_k = (w_k^1,\ldots,w_k^j,w_k^d))$ having covariance matrix, defined by $\sigma_k^2 I$ where σ_k is the standard deviation and I is the identity matrix,

$$N(\mathbf{x}_n;\mathbf{w}_k,\sigma_k) = \frac{1}{(2\pi\sigma_k)^{\frac{d}{2}}} \exp\left[-\frac{\|\mathbf{x}_n-\mathbf{w}_k\|^2}{2\sigma_k^2}\right].$$

- The initial latent state $\mathbf{z}_1^*$, which has a marginal distribution $p(\mathbf{z}_1^*)$ (prior probability) represented by a vector of probabilities π with elements $\pi_k = p(z_{1k}^*=1)$, so that $p(\mathbf{z}_1^*|\pi) = \prod_{k=1}^{K} \pi^{z_{1k}^*}$ where $\sum_k \pi_k = 1$.
- Transition probability $\mathbf{A}$: The state transitions are governed by a first-order Markov chains. Probability distribution of $\mathbf{z}_n^*$ depends on the state of the previous latent variable $\mathbf{z}_{n-1}^*$ through a conditional distribution $p(\mathbf{z}_n^* \,|\, \mathbf{z}_{n-1}^*)$. The elements of $\mathbf{A}$ are known as transition probabilities denoted by

$$A_{jk} = p(\mathbf{z}_{nk}^* = 1/\mathbf{z}_{n-1,j}^*) = 1 \quad \text{with} \sum_k A_{jk} = 1.$$

  In our case, the number of transitions are limited by the grid (map). We can then write the conditional distribution explicitly in the form:

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

The model parameters are completed by defining neighborhood probability between a couple of states responsible for the generation of an element. To introduce the self-organizing process in the learning PrSOMS model, we assume that $p(z_{nk}/z_{ns}^*)$ is known as follows:

$$p(z_{nk}=1/z_{ns}^*=1) \equiv p(z_{nk}/z_{ns}^*) \equiv p(\mathbf{z}_n/\mathbf{z}_n^*) = \frac{\mathcal{K}^T(\delta(k,s))}{\sum_{r=1}^{K}\mathcal{K}^T(\delta(r,s))}.$$

Typically the neighborhood function $\mathcal{K}^T(\delta) = \mathcal{K}(\delta/T)$ is a positive function which decreases as the distance between two prototypes in the latent space increases and where T controls the width of the neighborhood function. Thus T is decreased between two values $T_{\max}$ and $T_{\min}$. So by using a normalized neighborhood function centered on one of the cells, we force the components with large responsibility for an observation to be close in the latent space presented by the grid $\mathcal{S}$. In this manner, we obtain a training algorithm for mixture models similar to the traditional SOM.

3.2. *Cost function and optimization*

For a better understanding, we have used notations similar to those in the book (see Ref. 31, [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 $\mathbf{x}_n$. We assume, for each sequence of observations in $\mathbf{X}$ corresponds the latent variables $\mathbf{Z}$ and $\mathbf{Z}^*$. We denote by $\{\mathbf{X}, \mathbf{Z}, \mathbf{Z}^*\}$ the complete data, and refer to the observed data $\mathbf{X}$ as incomplete. The likelihood function is obtained from the joint distribution by marginalizing over the latent variables $\mathbf{Z}^*$ and $\mathbf{Z}$

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

An important concept for probability distributions over multiple variables is the conditional independence.[32] Often used for the graphical model, so, the joint distribution of the sequence is equal to:

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

where $\theta = \{\pi, \mathbf{A}, \phi\}$ denotes the set of parameters governing the model.

The generation is nondependent only on the current state but also on the previous state (i.e., on the transitions) and on the neighborhood cells at that same time step $p(\mathbf{Z}/\mathbf{Z}^*)$. We use the EM 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 $p(\mathbf{Z}^*, \mathbf{Z}/\mathbf{X}, \theta^{\text{old}})$. We then use this posterior distribution to evaluate the expectation of the logarithm of the complete-sequence data likelihood function (2), as a function of the parameters θ, to give the function $Q(\theta, \theta^{\text{old}})$ defined by:

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

We can then write the cost function in the form:

$$\begin{aligned}
Q(\theta,\theta^{\text{old}}) &= \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{k=1}^{K} p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{\text{old}}) z^*_{1k}\ln\pi_k \\
&+ \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{n=2}^{N}\sum_{k=1}^{K}\sum_{j=1}^{K} p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{\text{old}}) z^*_{n-1,j} z^*_n \ln(A_{jk}) \\
&+ \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}}\sum_{n=1}^{N}\sum_{k=1}^{K} p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{\text{old}}) z_{nk}\ln(p(\mathbf{x}_n;\phi_k)) \\
&+ \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}} p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{\text{old}})\ln p(\mathbf{Z}/\mathbf{Z}^*).
\end{aligned}$$

We use $\gamma(\mathbf{z}_n)$ and $\gamma(\mathbf{z}^*_n)$ to denote respectively the marginal posterior distribution $p(\mathbf{z}_n/\mathbf{X})$ and $p(\mathbf{z}^*_n/\mathbf{X})$ of a latent variable z_n and z^*_n. 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 $(\mathbf{z}^*_{n-1},\mathbf{z}^*_n)$. We also use $\gamma(\mathbf{z}_{nk})$ and $\gamma(\mathbf{z}^*_{nk})$ to denote respectively conditional probability of $z_{nk}=1$ and $z^*_{nk}=1$, with a similar use of notation for $\xi(z^*_{n-1,j},z^*_{n,k})$

$$\begin{aligned}
Q(\theta,\theta^{\text{old}}) &= \sum_{\mathbf{Z}^*}\sum_{\mathbf{Z}} p(\mathbf{Z}^*,\mathbf{Z}/\mathbf{X};\theta^{\text{old}})\ln p(\mathbf{X},\mathbf{Z}^*,\mathbf{Z};\theta) \\
&= \sum_{k=1}^{K}\gamma(z^*_{1k})\ln\pi_k \\
&\quad + \sum_{n=2}^{N}\sum_{k=1}^{K}\sum_{j=1}^{K}\xi(z^*_{n-1,j}z^*_n)\ln(A_{jk}) \\
&\quad + \sum_{n=1}^{N}\sum_{k=1}^{K}\gamma(z_{nk})\ln p(\mathbf{x}_n;\phi_k) \\
&\quad + \xi(z^*_{n-1,j}z^*_n)\ln p(\mathbf{Z}/\mathbf{Z}^*).
\end{aligned}$$

In the M step, the maximization of the function $Q(\phi,\theta^{\text{old}})$ with respect to the parameters $\theta=\{\pi,\mathbf{A},\phi\}$ is easily achieved. The first term depends on initial probabilities; the second term depends on transition probabilities $\mathbf{A}$; the third term depends on ϕ, and the fourth term is constant. The expressions are defined as follows:

$$\mathbf{w}_k = \frac{\sum_{n=1}^{N}\gamma(z_{nk})\mathbf{x}_n}{\sum_{n=1}^{N}\gamma(z_{nk})}, \tag{3}$$

$$\sigma_k^2 = \frac{\sum_{n=1}^{N}\gamma(z_{nk})\|\mathbf{x}_n-\mathbf{w}_k\|^2}{d\sum_{n=1}^{N}\gamma(z_{nk})}, \tag{4}$$

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

$$\pi_k = \frac{\gamma(z^*_{1k})}{\sum_{j=1}^{K}\gamma(z^*_{1j})}, \tag{5}$$

$$A_{jk} = \frac{\sum_{n=2}^{N} \xi(z^*_{n-1,j}, z^*_{nk})}{\sum_{l=1}^{K} \sum_{n=2}^{N} \xi(z^*_{n-1,j}, z^*_{nl})}, \tag{6}$$

where

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

3.3. *The forward–backward algorithm*

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. As the HMM, this is known as the forward–backward algorithm,[33] or the Baum–Welch algorithm.[34,35] 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 do not use the graph structure (see Ref. 31, Chap. 6). We use the notations $\alpha(z^*_{nk})$ and $\alpha(z_{nk})$ to denote the values of $\alpha(\mathbf{z}^*)$ and $\alpha(\mathbf{z})$ when $z^*_{nk} = 1$, $z_{nk} = 1$ with an analogous notation of β. Using Bayes theorem, we have

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

Using the similar decomposition, we obtain

$$\begin{aligned}
\gamma(\mathbf{z}_n) &= p(\mathbf{z}_n / \mathbf{X}) = \frac{p(\mathbf{X} \,|\, \mathbf{z}_n) p(\mathbf{z}_n)}{p(\mathbf{X}s)} \\
&= \frac{p(\mathbf{x}_1, \ldots, \mathbf{x}_n, \mathbf{z}_n) p(\mathbf{x}_{n+1}, \ldots, \mathbf{x}_N / \mathbf{z}_n)}{p(\mathbf{X})} \\
\gamma(\mathbf{z}_n) &= \frac{\alpha(\mathbf{z}_n)\beta(\mathbf{z}_n)}{p(\mathbf{X})}.
\end{aligned}$$

The values $\alpha(\mathbf{z}^*_n)$ and $\alpha(\mathbf{z}_n)$ represent respectively the joint probability of observing all of the given data up to time n, the values of $\mathbf{z}^*_n$ and $\mathbf{z}_n$. $\beta(\mathbf{z}^*_n)$ and $\beta(\mathbf{z}_n)$ represent respectively the conditional probability of all future data from time $n+1$ up to N given the values of $\mathbf{z}^*_n$ and $\mathbf{z}_n$. The values of $\alpha(\mathbf{z}^*_n)$ and $\alpha(\mathbf{z}_n)$ are calculated

by forward recursion as follows:

$$\alpha(\mathbf{z}_n^*) = p(\mathbf{x}_n|\mathbf{z}_n^*) \sum_{\mathbf{z}_{n-1}^*} \alpha(\mathbf{z}_{n-1}^*) p(\mathbf{z}_n^*|\mathbf{z}_{n-1}^*),$$

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

and

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

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

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

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

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

$$\beta(\mathbf{z}_n) = \frac{1}{p(\mathbf{z}_n)} \sum_{\mathbf{z}_n^*} p(\mathbf{z}_n^*) p(\mathbf{z}_n/\mathbf{z}_n^*) \sum_{\mathbf{z}_{n+1}} \sum_{\mathbf{z}_{n+1}^*} \times p(\mathbf{z}_{n+1}/\mathbf{z}_{n+1}^*) \beta(\mathbf{z}_{n+1}^*) p(\mathbf{x}_{n+1}/\mathbf{z}_{n+1}^*) p(\mathbf{z}_{n+1}^*/\mathbf{z}_n^*), \quad (7)$$

where

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

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

It is clear that the neighborhood function is independent of time:

$$p(\mathbf{z}_{n+1}/\mathbf{z}^*_{n+1}) = p(\mathbf{z}_n/\mathbf{z}^*_n),$$

$$p(\mathbf{z}_{n+1}/\mathbf{z}^*_{n+1}) = p(z_{n+1,k} = 1/z^*_{n+1,s} = 1) = \frac{\mathcal{K}^T(\delta(k,s))}{\sum_{r\in\mathcal{S}}\mathcal{K}^T(\delta(r,s))}.$$

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 (Eq. (7)).

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 Bayes theorem, we obtain:

$$\xi(\mathbf{z}^*_{n-1}, \mathbf{z}^*_n) = p(\mathbf{z}^*_{n-1}, \mathbf{z}^*_n/\mathbf{X}) = \frac{p(\mathbf{X}/\mathbf{z}^*_{n-1}, \mathbf{z}^*_n)p(\mathbf{z}^*_{n-1}, \mathbf{z}^*_n)}{p(\mathbf{X})},$$

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

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 $\theta(\pi, A, \phi)$.

Let us summarize the steps required to train a PrSOMS model using the EM algorithm. We first make an initial selection of the parameters θ^{old} where $\theta(\pi, A, \phi)$. As the traditional HMM, the parameters $\mathbf{A}$ and π are initialized uniformly with a uniform distribution. The parameters ϕ can be initialized by applying the clustering algorithm on data with i.i.d hypothesis.

3.4. *Map visualization*

In order to visualize cell and state transitions, we use the probability $\gamma(\mathbf{z}^*_n) = p(\mathbf{z}^*_n/\mathbf{X})$. We use the Viterbi algorithm, which computes the most probable sequence of states. Selecting only the best sequence corresponds to the winner-take-all principle of the PrSOMS.[33,36] In our case, the Viterbi algorithm takes into account the neighborhood on the map in determining the most probable path. Proceeding in this way, we inherit all the visualization tools used in classical traditional maps. Note that the majority of the result should be analyzed in color mode.

4. Experimentations

The experiments concerns the assessment of the suitability of the PrSOMS model for the exploratory analysis of multi-dimensional sequences (or non-i.i.d data) through clustering, structuring and visualization. The experiments were performed in Matlab using Bayes Net Toolbox written by Kevin Murphy, 1997–2002 (http://code.google.

com/p/bnt/) and SOM Toolbox (http://www.cis.hut.fi/somtoolbox/). Several experiments were devised in order to assess the suitability of the PrSOMS model for analysis, assisted by visualization, of non-i.i.d data. These experiments were organized according to three different main objectives: First, evaluate the power of clustering, visualization and structuring of PrSOMS on simulated data, with known ground truth. Second, we aimed to compare the different results yielded by the PrSOMS and the GTM-TT when clustering multi-dimensional data. This way, the advantages of using the latter model would be highlighted. Finally, evaluate the ability of PrSOMS to deal sequences of variable lengths, and its power to model and visualize the dynamic of a phenomenon described by sequences of events through TV broadcast.

4.1. *Experimental data sets*

(1) *Handwritten-data* (Table 1): The dataset here were used for a Ph.D. study on primitive extraction using HMM-based models. The data consists of 2858 character samples, contained in the cell array "mixout",[37,38] https://archive.ics.uci.edu/ml/datasets/Character+Trajectories. The data was captured using a WACOM tablet, where three dimensions were kept — x, y, and pen tip force.

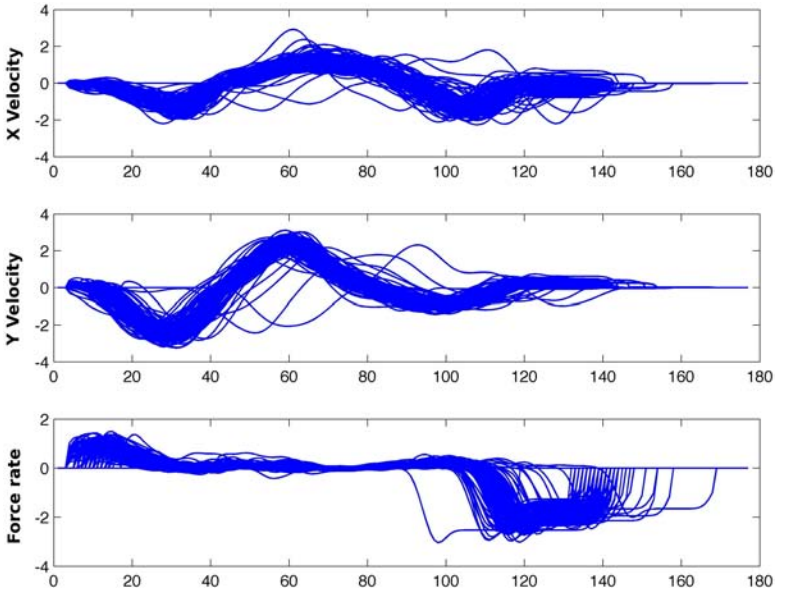
(2) *Audiovisual-data* (Table 1): A real dataset from INA (the "French National Audiovisual Institute"). The data consists of 10 sequences of varying lengths such that each sequence represents the various segments broadcast in a given day. Each segment of the sequence is multi-dimensional and characterized by 23 variables. These variables are constructed in consultation with an expert user. These contain descriptors about the number of repetitions, the interval and other variables that we cannot disclose in this paper.

4.2. *Handwritten-data*

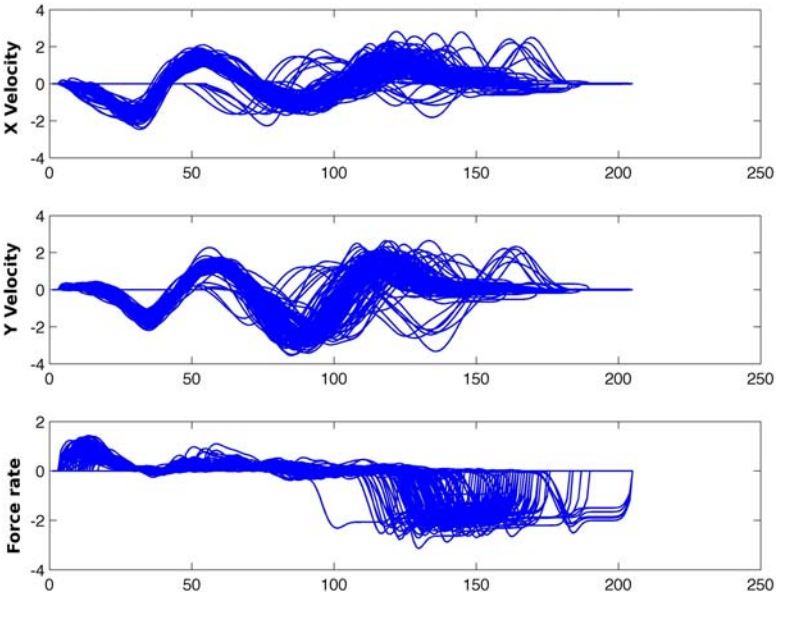
The data has been numerically differentiated and Gaussian smoothed, with a sigma value of 2. Data was captured at 200 Hz and normalized. Only characters with a single "PEN-DOWN" segment were considered. Character segmentation was performed using a pen tip force cut-off point. The characters have also been shifted so that their velocity profiles best match the mean of the set. Each character sample is a three-dimensional pen tip velocity trajectory. Figure 2 shows overlay plot of 131 character samples of the letter "p", and 124 of the letter "q" demonstrating the variation in the data. In order to show the advantages of self-organization of

Table 1. Data features.

Sequence data set	Size	# features
Handwritten-data	2858	3
Audiovisual-data	10 sequences of varying lengths	23



(a)



(b)

Fig. 2. Overlay plot of 131 character samples of the letter p and q, demonstrating the variation in the data. (a) lettre p, (b) lettre q.

PrSOMS, we run separately PrSOMS model on four data sets: p-data set, q-data set, abc-data set.

Figure 3 shows the cluster membership map. In this map, the PrSOMS latent states are represented by squares that are scaled according to the ratio of elements that the model captured using the Viterbi algorithm. Thus, our PrSOMS allows a clustering of the data taking into account the non-iid property.

Figures 4(a) shows the PCA projection of the p-data set visualized in the latent-variable (maps projection). The blue points present the original element of sequence. Figure 4(b) shows in the latent space the Viterbi path obtained for all character "p". Visualizing both figures, we observe a clear topological organization of the PrSOMS map. These projections provide a topographical visualization of sequential data set: Close samples are represented by close states. This figures show that PrSOMS is useful for visualizing in low-dimensional views of high-dimensional sequence. Another way to visualize the results is to plot the letter p-data set in tablet space and indicate (using a color) the number of the probable states provided in Viterbi path, Fig. 4(c). Figure 4(d) shows the same information by plotting the p-data set in velocity space. Both figures confirm that the topological order is respected. Overall, the Viterbi path starts with the states located in the top left of the map and moves towards the middle of the map. Finally it ends in the bottom left of the map and back in the middle of the map. Figure 5(a) shows the three dimensions, in the velocity

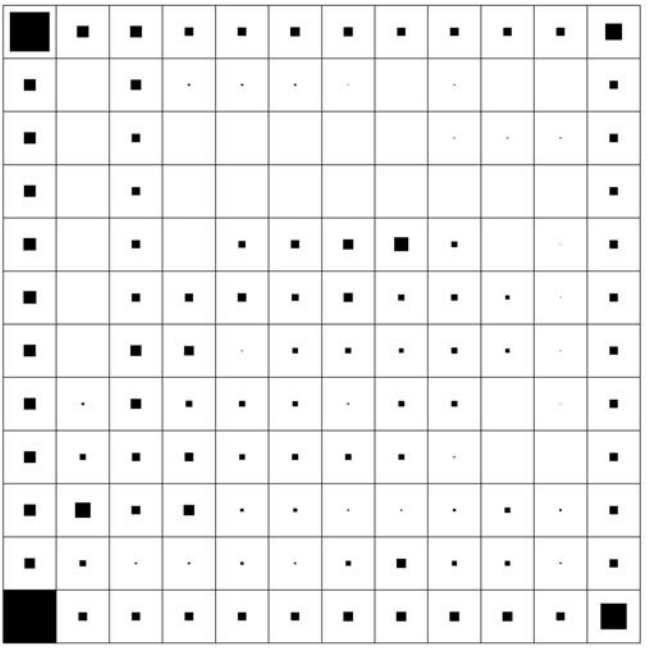


Fig. 3. p-PrSOMS 12×12 map. Cluster membership map. The squares are scaled according to the ratio of elements that the model captured using the Viterbi algorithm.

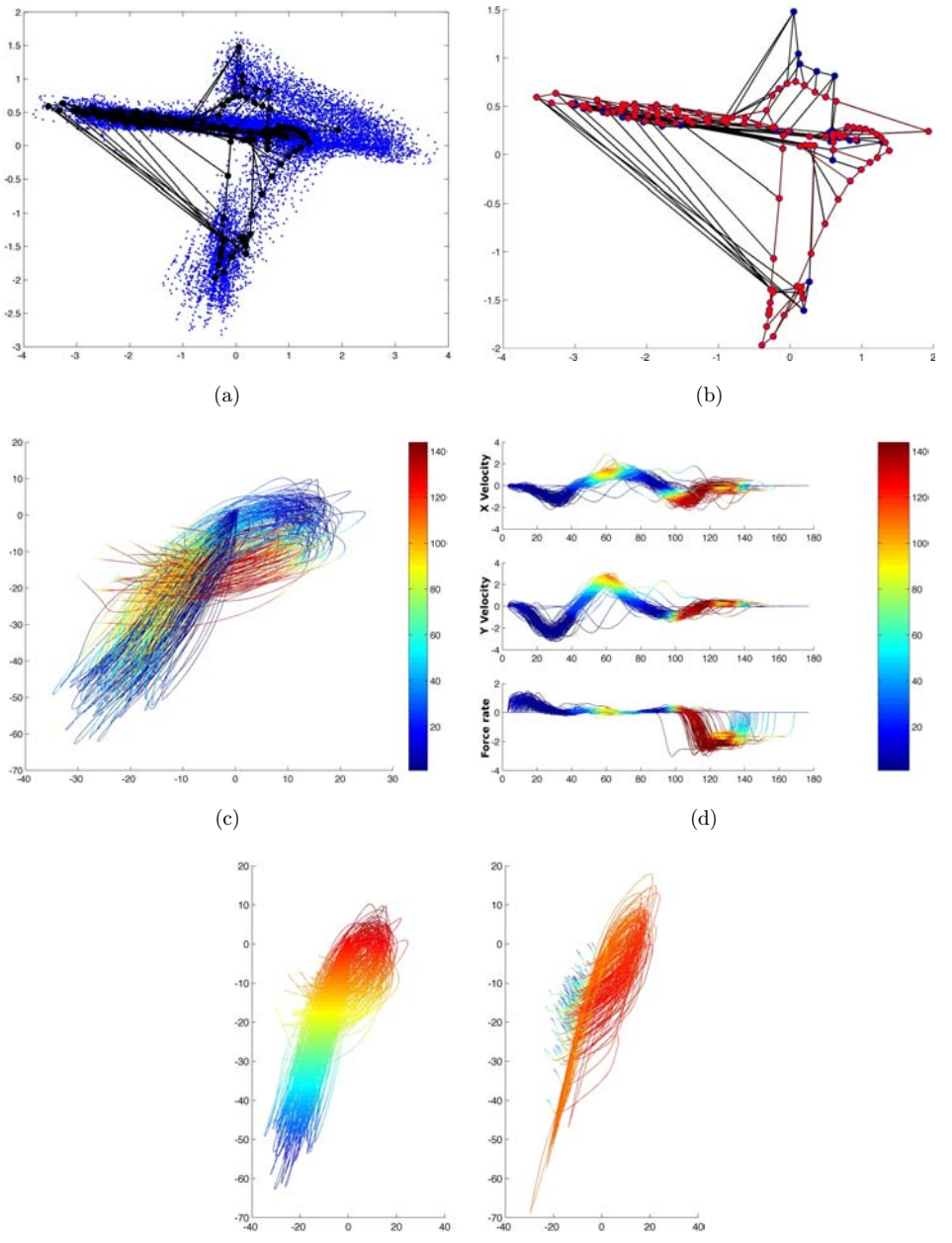
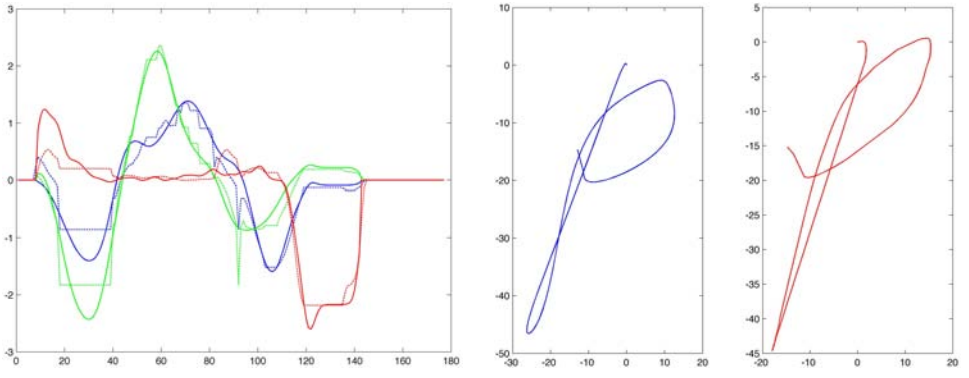


Fig. 4. (Color online) $12 \times 12p$-PrSOMS map. (a) PCA projection of samples and map. The points in blue present the element of the sequence. (b) The red line indicates the Viterbi path of all character "p" overlaid with PCA projection of states. (c) Samples of p-dataset, each color indicates the number of the probable states provided in Viterbi path. (d) Samples of p-dataset in velocity space. (e) Comparison of (left) original samples and (right) reconstruction samples in pen-space, where the color indicates the force rate. On the left the original samples.



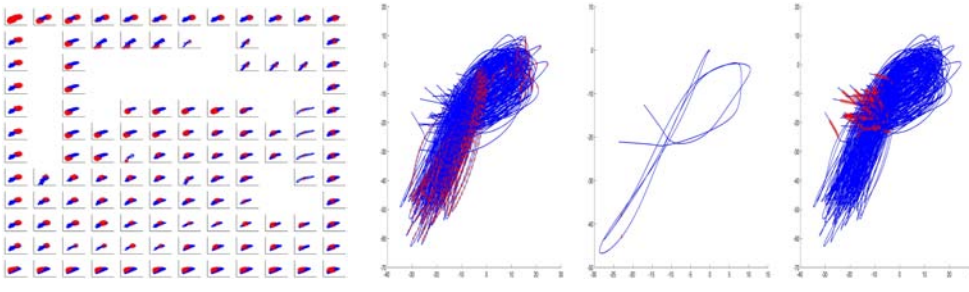
(a) (b)

Fig. 5. (Color online) Single character sample shown in velocity space (left) and position. (a) Shows three dimensions of original single character overlaid with the reconstructed character plotted as dashed lines. The dimensions are color-coded (blue = x velocity, green = y velocity, and red = pen force). (b) The left column shows the original data in blue; the right column shows the reconstruction in red.

space, of the single original character overlaid with the reconstruction motifs plotted as dashed lines. Other visualizations can be done, thus Fig. 6(a) shows the PrSOMS map, where each cell indicates the letter "p". The samples assigned to the cell are plotted in red. Figure 6(b) is an enlargement of states: 1, 55, 144. These visualizations could be used by an expert in order to explore the region of interest (in our case the element of sequence).

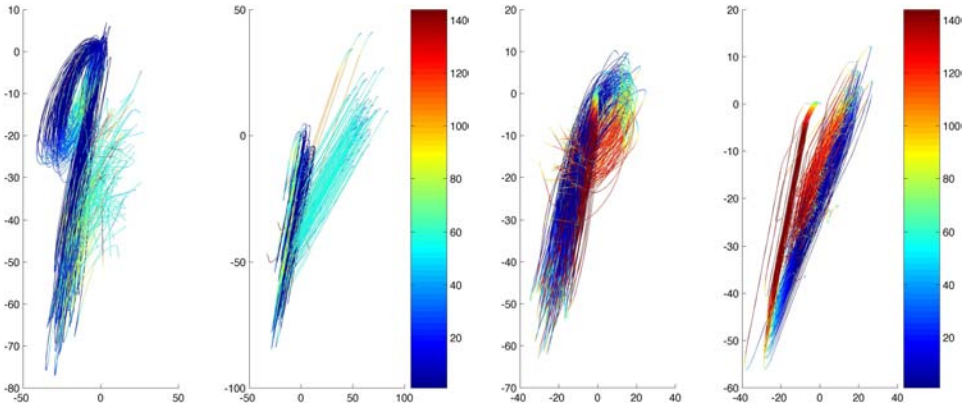
One of the characteristic features distinguishing PrSOMS from other HMM paradigms dedicated to non-i.i.d, is the topographic preservation using the neighborhood function.

To show the generative model power, we reconstructed the character p using the q-PrSOMS map learned with the character q and vice versa. Figures 7(a) and 7(b)



(a) (b)

Fig. 6. (Color online) $12 \times 12p$-PrSOMS map. (a) Visualizing letter in tablet space. The samples assigned to state is indicated with red color. (b) Zoom on states: 1, 55, 144.



(a) (b)

Fig. 7. (Color online) (a) Reconstruction of q-data set using the p-PrSOMS model. (b) Reconstruction of p-data set using the q-PrSOMS model. For each subfigure, the left figure displays the original characters and the right the reconstruction character. Each color indicates the number of probable states provided in Viterbi path.

show in both cases, that each model provides a good reconstruction of the common part of the character p and q.

As a more challenging problem, we used the *abcpq*-data set. We used the K-fold cross validation technique for visualization purposes, with $k = 3$. For each experiment, two subsets are used for training and then we tested the model on the remaining subset. Figure 8 shows samples of the reconstruction of novel characters using the same parameters after each training. For each row, we have three columns with two subfigures: On the left are the original characters and on the right are the reconstructed characters. Each figure is plotted in pen-space (x, y position and pen pressure). The color indicates the pen pressure value. The Viterbi-based reconstruction produces a very close reconstruction of the data set. At each experience, the novel character types are well reconstructed, and easily recognizable.

4.3. *Measurement criteria: Differences between PrSOMS and GTM-TT*

We used the K-fold cross validation technique, with $k = 3$, to estimate the performance of PrSOMS. In this case, we used data set with $\{a, b, c, p, q\}$ characters. For each run, the data set was split into three disjoint groups. We used two subsets for training 12×12 map and then tested the model on the remaining subset. We made two experiments: The first experiment is done by learning a PrSOMS map using all characters $\{a, b, c, p, q\}$ for each run. Thus, we assign the sequence data tests using the Viterbi algorithm and compute the quantization error. The second experiment concerns to test PrSOMS model as classifier. In this case, we learn that for each run five PrSOMS maps, each map is learned with a single character. In the first

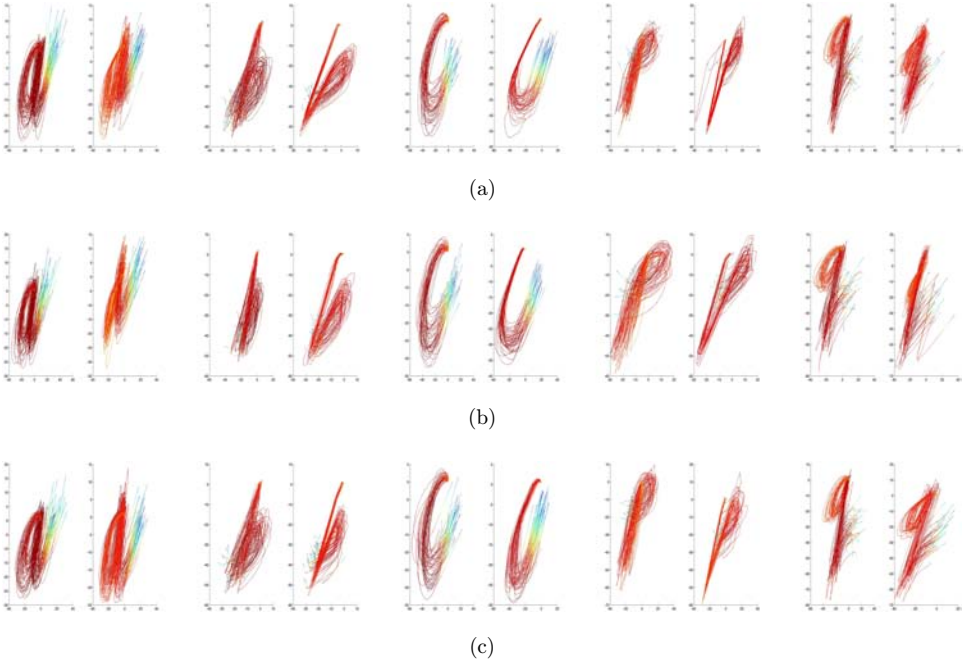


Fig. 8. PrSOMS training. Reconstructions of a novel dataset using 3-fold cross validation using $\{a, b, c, p, q\}$-data set. The characters are not present in the training set. For each experience (row), original characters are shown on the left, and character reconstructions are shown on the right. (a) Experience 1, (b) Experience 2, (c) Experience 3.

experiment, we observe that our PrSOMS approach can learn in one map multiple characters and multi-dimensional sequence using single map. Table 2 shows the rate of quantization error after Viterbi assignment defined in Sec. 3.4. We observe that PrSOMS model improves the performance and provides better result than GTM-TT. The GTM-TT is applied successfully using one-dimensional times series. Reference 23 provides a nice validation GTM-TT method.

In the second experiment, for classification purpose, we learned PrSOMS map and GTM-TT map using a single character. Table 3 indicates the rate of good classification (accuracy). We observe that topological maps improved the performance of PrSOMS algorithm, in the same time GTM-TT obtains similar result.

Table 2. Cross-validation with $\{a, b, c, p, q\}$-data set. We learn for each run a single $\{a, b, c, p, q\}$-PrSOMS map. The values indicate quantization error.

Model ($10^3 \times$)	a	b	c	p	q
PrSOMS	1.4756	1.0926	1.3791	1.5507	1.5636
GTM-TT	2.4902	2.4650	3.2825	2.5845	2.9909

Table 3. Cross-validation with a-PrSOMS map, b-PrSOMS map, c-PrSOMS map, p-PrSOMS map and q-PrSOMS map. The value indicates a good classification rate.

Model	a	b	c	p	q
PrSOMS	100	99.16	99.28	97.41	100
GTM-TT	100	99.38	99.18	97.35	99.94

4.4. *Discussion about self-organization process: From PrSOMS to traditional HMM*

The PrSOMS model allows us to estimate the parameters maximizing the log-likelihood function Q^T for a fixed T, which is decreased between $T_{\max}$ and $T_{\min}$. The PrSOMS model is trained on multi-character data sets. In order to study the self-organization process, we use in this case, random initialization. We used a 12×12 map in two-dimensional latent space. In Fig. 9, we show the PCA projection in latent space and it can be clearly seen how the PrSOMS spreads out over the data as the neighborhood function is shrunk. Figure 9 shows the configuration after five iterations and the latest iteration. We can observe that the PrSOMS spreads out over the data as the neighborhood function width decreases. When decreasing T, the model of PrSOMS is defined in the following way:

- The first step corresponds to high T values. In this case, the influencing neighborhood of each state z_{ns}^* on the PrSOMS map is important and corresponds to higher values of $\mathcal{K}^T(\delta(k, s))$. Each state $s \in \mathcal{S}$ uses a high number of neighborhood states to generate an element of the sequence. This step provides the topological order.
- The second step corresponds to small T values. The neighborhood is small and limited, therefore, the adaptation is very local and the parameters are accurately computed from the local density. In this case, we consider that PrSOMS converge to traditional HMM.

It is clear that the computational cost of our PrSOMS model is more expensive than traditional HMM. However, by restricting the neighborhood of $\mathbf{z}^*$ to limit a number of pairs of states $(\mathbf{z}, \mathbf{z}^*)$, we can obtain a low computational cost.

We could compare our method in classification mode, but it is not our purpose here. We have shown in this section, that our approach converges to a HMM at the end of learning. It can be easily used for classification mode. The most relevant thing, is that our approach provides both PrSOMS clustering taking into account the non-i.i.d. property and provides a dimension reduction. This allows visualization in a two-dimensional friendly space, which is useful for experts. It is also clear that our approach inherits all the visualization already known with i.i.d data using the traditional SOMs, that we cannot detail here.

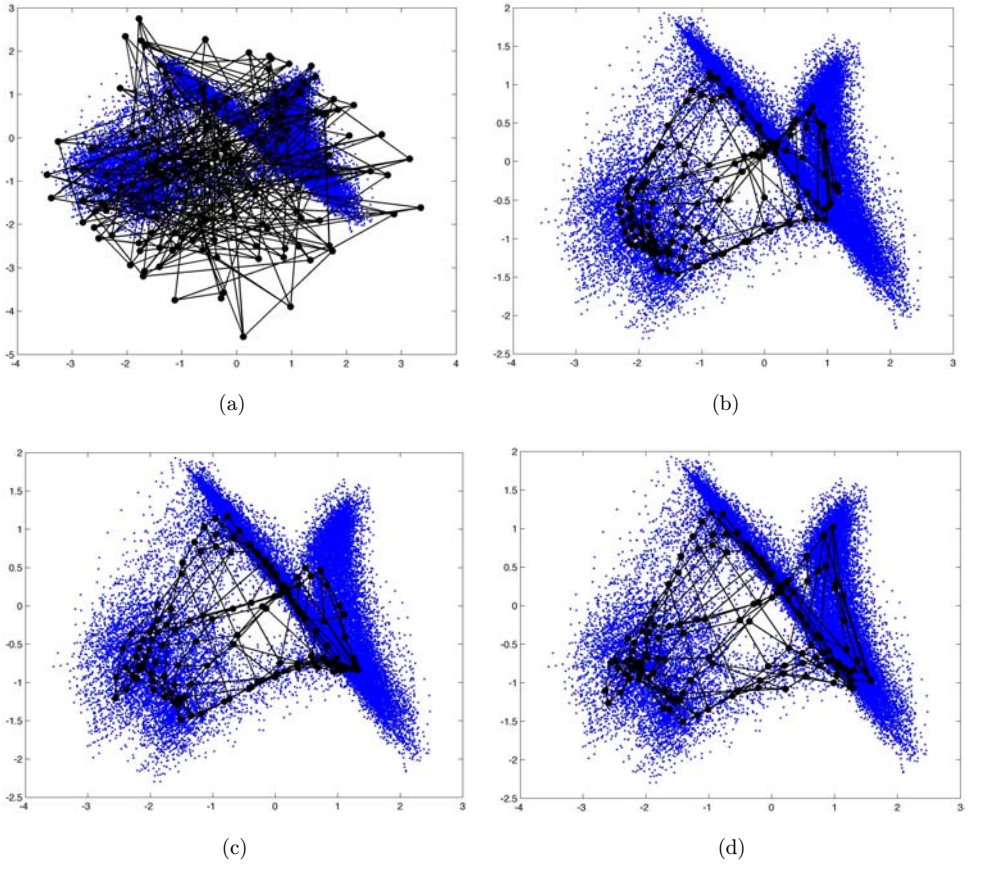


Fig. 9. Configuration of the *abc*-PrSOMS 12×12 training after 5; 15; 20 iterations. (a) Random Initialization, $T = 5$, (b) iteration:5, $T = 2.57$, (c) iteration:15, $T = 1.52$, (d) iteration:20, $T = 1$.

4.5. *Audiovisual-data: Broadcast digital TV analysis*

The resulting huge and continuously growing content has given rise to many novel services around TV and video platforms like TV-on-Demand (TVoD), interactive TV, Catch-up-TV, Network Personal Video Records (NPVRs), and so forth. This content is also part of our audio–visual heritage and must be properly archived. Digital TV content is generally achieved by national public institutions like INA in France, Geluid in Netherlands, ORF in Austria or BBC archives in UK. Therefore, the digital TV content has to be analyzed in order to be used within these services. Basically, analyzing a digital TV content consists of clustering and structuring the content. The objective of TV broadcast analysis is to recover the original structure of the TV stream and to cluster the homogeneous video segments. In TV streams, useful long TV programs (like movies, news, series, etc.), short programs (like weather forecast, very short games, etc.) and inter-programs (commercials, trailers,

sponsorships, etc.) are concatenated and broadcasted without any precise and reliable bags that identify their boundaries.

Here, the problem is two-fold: (1) Cluster the video segments for identification of the TV program. Metadata are used to label and extract programs using simple overlapping-based criteria. (2) TV broadcast structuring, that consists of automatically and accurately determining the boundaries (i.e., the start and end) of each broadcasted program and inter-program. In addition to precise and automatic boundary detection, TV broadcast structuring gathers different parts of the same program and labels them when metadata are available. Hence, it can detect complex structures that exist in the underlying structure of the content.

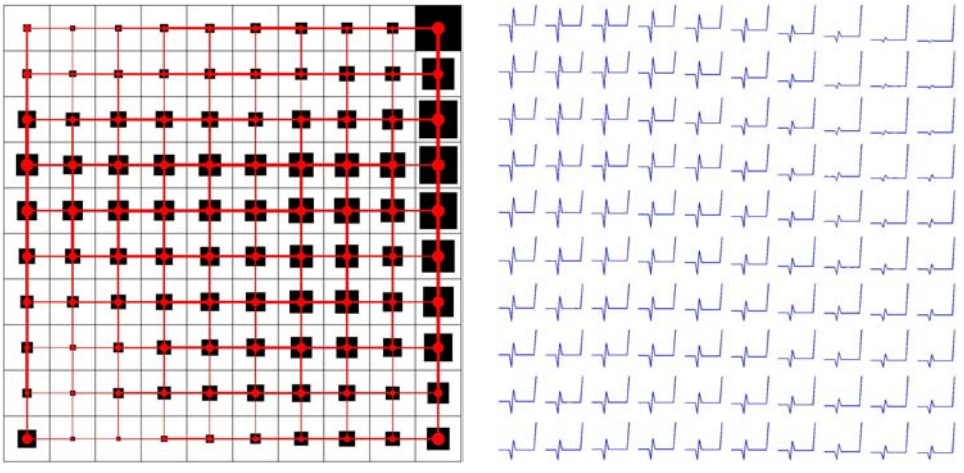
Analyzing a TV broadcast can be seen as an analysis problem of multivariate sequences and one of the simplest models to resolve this is to use our PrSOMS approach, because of its ability to deal with sequences of variable lengths, and its power to model and visualize the dynamic of a phenomenon described by sequences of events.

4.5.1. *Capturing the dynamics of channels through visualization*

In this experiment, the performance of clustering is studied. We recall that the problem here is to automatically detect sequence segments, that have the same subjects. The clustering could be used to detect redundant rebroadcasts of a same program (a long video segment that contains a program broadcast multiple times). Cluster is labeled as a title when all the segments appear at a certain timing of a day based on the assumption that programs represent a similar subject. TV program extraction also depends on the metadata information provided by TV channels. This metadata is required to be able to give names to automatically extracted clusters.

Figure 10(a) shows the cluster membership map after training sequences broadcasted in a given day on 10 French channels. In this map, the 10×10 PrSOMS latent states are represented by squares that are scaled according to the ratio of elements of all sequences that the model captured using the Viterbi algorithm. The red lines represent the Viterbi path provided for each sequence. The lines are scaled according to the number of transition between states. Figure 10(b) display the profiles or the prototype defined in each cell. Each cell displays the 19 components. The topological maps in our PrSOMS model allows us to visualize the partition. So, an experts could use these plots by zooming on a cell to analyze some of its features. Both figures allow us to analyze all the components of 10 channels or sequences at the same time.

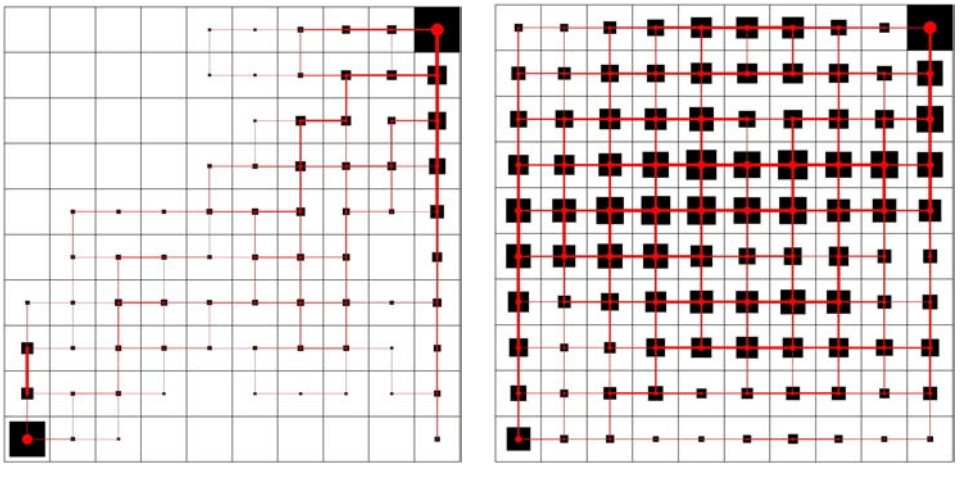
PrSOMS clustering of sequences can indeed be meaningful. This must be understood in the sense that the distribution of two different sequences over the PrSOMS map should be significantly different. Let us illustrate this with a comparison between Viterbi algorithm yielded by two channels, focusing on the illustration of the main differences in the visualization of sequences. Figures 11(a) and 11(b) display respectively the cluster membership map representing the Viterbi path



(a) (b)

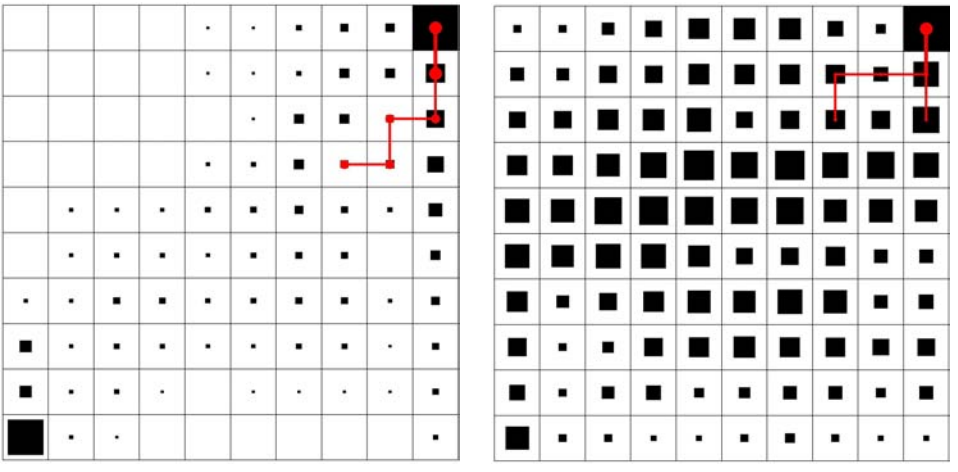
Fig. 10. (a) Cluster membership map. The squares and lines are scaled according to the ratio of elements and transitions that the model captured using the Viterbi algorithm. (The trajectory is represented by the lines between states.) (b) PrSOMS profile organizations. Each cell indicates state profiles composed of 19 variables.

calculated on channels 1 and 2. The representation corresponding to channel 1 (Fig. 11(b)) is more compact than channel 2 (Fig. 11(b)). This is due to the fact that element variabilities in channel 1 is less than in channel 2. Channel 1 is more concentrated in the bottom left and top right of the map.



(a) (b)

Fig. 11. (a) and (b) represent a PrSOMS cluster membership map representing all Viterbi path of all sequences of channels 1 and 2. (c) and (d) represent a PrSOMS cluster membership map representing an example of subsequence of two channels with the common states.



(c) (d)

Fig. 11. (*Continued*)

Figures 11(c) and 11(d) show two subsequences with some identical state trajectories, indicating that the model is actually identifying the similarities between sequences. Figures 12 and 13 display respectively images extracted from cells on the top right and bottom left. The majority of the videos in Fig. 12 correspond to the

Fig. 12. Images extracted from video captured on the top right of the map.

Fig. 13. Images extracted from video captured on the bottom left of the map.

short programs (meteo, short news) which is a characteristic of a news channel (channel 2). Figure 13 displays movies captured by the bottom left state in channel 1 (general channel).

We present in Figs. 14 and 15 the segments that are distributed respectively in channels 1 and 2. The length of the baton reflects a segment broadcast multiple times and the color of the baton refers to channels where it was broadcast: A gray stick indicates broadcast in a single chain (e.g., news event), whereas a colored stick means that it was broadcasted on several channels (e.g., advertising). We note the existence of the video segment groups spread the same way several times a day. These video segments were distributed over several channels so it is probably advertising. This behavior is similar to channel 1 except that the video segments were distributed only in channel 1 (Baton gray) so this is probably news.

From the figures above, we can draw several characteristics of channels such as video segments broadcast in several channels, channels broadcasting the same advertisement, channels which do not broadcast advertisements and channels that have specific programs.

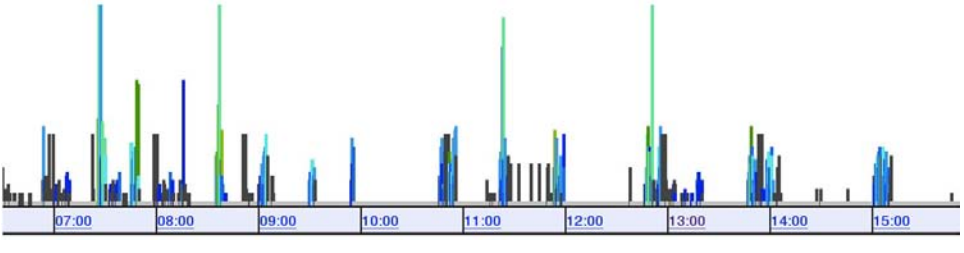


Fig. 14. Characteristic of the broadcast of segments in the channel 2 between 7 am and 3 pm.

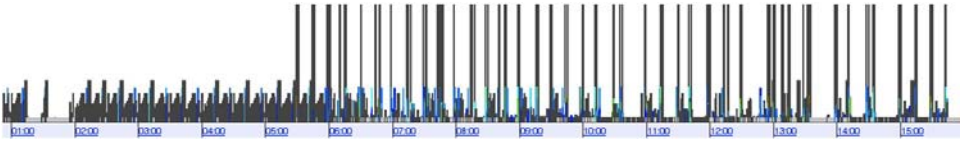


Fig. 15. Characteristic of the broadcast of segments in the channel 1 between 1 am and 3 pm.

4.5.2. *Structuring analysis*

We present the experiment that evaluates our system, that is program structuration. Let us illustrate this with clustering the latent space using Hierarchical clustering as used by i.i.d data. This step allows to reduce the number of clusters.[39] In order to get close to expert partitioning, we applied the ascendant hierarchical clustering over the PrSOMS maps to get a partition formed by eight clusters (Fig. 16). First, more than 90% of interprograms (i.e., commercials, sponsorships and trailers) are broadcast at least twice within one day over ten channel. Second, only about 30% of short programs are repeated. This also lets us suppose that detecting short programs is the key for an accurate TV program extraction. The accuracy of extracted programs has also been evaluated. The start (respectively the end) of each extracted program has been compared to the actual start (respectively the end) given by actual observations.

The obtained results show that the system has performed a successful TV stream segmentation. In particular, for the midday and the afternoon intervals, all the observed boundaries are correctly identified. In particular during the night, many long programs are sequenced without any separating inter-program. These results

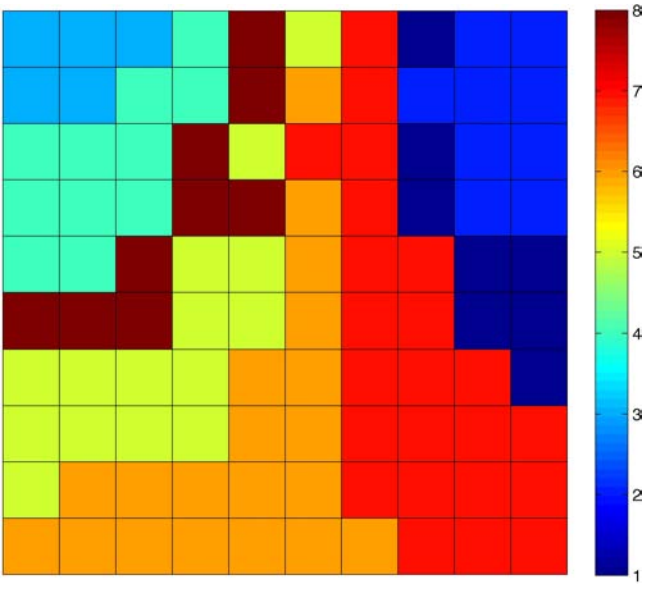


Fig. 16. Hierarchical clustering of PrSOMS map.

Fig. 17. The images corresponding to cell 1 of the blue cluster (top, left).

Fig. 18. The images corresponding to cell 2 of the blue cluster (top, left).

also suggest that TV program extraction is likely to be more accurate from midday to the evening.

The analysis of the dataset shows that most interprograms are broadcast several times. There are too many versions of this short TV game that share too many features, which confuses our system. The expert notice that the number of detected program segments is very high with respect to the number of actual program segments observed. This is due to an over segmentation that is easily dealt with using a merge procedure. This procedure properly fuses consecutive program segments that belong to the same program during the labeling step.

In the following, we visualize an example of images of the segments constituting the corresponding cells in each cluster. We note that Figs. 17–19 in the neighboring cells are homogeneous and represent the same type of program. We also note that the repeated segments are together in neighboring cells. We reviewed the audio-visual sequences associated to each cluster and found that each cluster represents a particular type of program different from other.

Fig. 19. The images corresponding to cell 3 of the blue cluster (top, left).

Fig. 20. Some prototype images describing each cluster.

Figure 20 shows for each cluster a set of prototype image component cells. We note that the clusters are composed of homogeneous content. In our case, the automatic clustering conforms perfectly to our actual observations, while indicators of the guide are quite staggering. The results are very encouraging, producing a correct clustering in most of cases. A major difficulty is that the quality of results depends on the detection of repetitions and the number of variables used for clustering.

5. Conclusion

Whilst visual exploration is an important stage in data mining processes, little research has been devoted to unsupervised methods for the visual exploration of multi-dimensional sequence (non-i.i.d data). In this paper, we have presented the capabilities of the PrSOMS model, that performs simultaneous non-i.i.d clustering and visualization whilst maintaining a low computational cost. The PrSOMS model generalizes HMM formalism and provides a self-organization map within a single probabilistic learning scheme for non-i.i.d data. The proposed PrSOMS model provides also a novel topographical visualization tool for non-i.i.d data. This model could be applied to more advanced/complex data sets (time series, sequences with mixed component (binary and continuous)). We provide here the mathematical formulation which could be used with different distributions. The parameters are estimated using an EM algorithm with a Baum–Welch algorithm. The proposed model was tested using a real world-data set. We present different views offered by

the PrSOMS. Since our probabilistic approach is close to the traditional maps, PrSOMS inherits all the known visual methods in the SOM.

We plan to extend this work in several directions. First, we intend to investigate the theoretical properties of PrSOMS, capturing long range correlations between the observed variables (i.e., between variables that are separated by many steps). Second, we can use the model to cluster data stream and perform characterization, novelty and visual detection by finding sequences of states which have a small probability under the trained model. An extension to an on-line mode version is s required. Finally, providing an equivalent of the PrSOMS for applications dealing with mixed component should be an interesting task.

References

1. T. Kohonen, *Self-Organizing Maps* (Springer Berlin, 2001).
2. M. Lebbah, Y. Bennani and N. Rogovschi, A probabilistic self-organizing map for binary data topographic clustering, *Int. J. Comput. Intell. Appl.* **7**(4) (2008) 363–383.
3. N. Rogovschi, M. Lebbah and Y. Bennani, A self-organizing map for mixed continuous and categorical data, *Int. J. Comput.* **10**(1) (2011) 24–32.
4. N. Yamaguchi, Self-organizing hidden markov models, in *Neural Information Processing. Models and Applications*, K. Wong, B. Mendis and A. Bouzerdoum, eds., Volume 6444 of Lecture Notes in Computer Science (Springer Berlin/Heidelberg, 2010), pp. 454–461.
5. F. Anouar, F. Badran and S. Thiria, Self-organizing map, a probabilistic approach, in *Proc. WSOM'97-Workshop on Self-Organizing Maps, Espoo*, Finland 4–6, June (1997), pp. 339–344.
6. S. Wu and T. W. S. Chow, Prsom: A new visualization method by hybridizing multidimensional scaling and self-organizing map, *IEEE Trans. Neural Networks* **16**(6) (2005) 1362–1380.
7. J. Kangas, Time-delayed self-organizing maps, in *Proc. Int. Joint Conf. Neural Networks, IJCNN'90*, San Diego, CA, USA (1990), pp. 331–336.
8. J. C. Wiemer, The time-organized map algorithm: Extending the self-organizing map to spatiotemporal signals, *Neural Comput.* **15** (2003) 1143–1171.
9. G. J. Chappell and J. G. Taylor, The temporal kohonen map, *Neural Network* **6**(3) (1993) 441–445.
10. N. Gianniotis and P. Tino, Visualization of tree-structured data through generative topographic mapping, *IEEE Trans. Neural Networks* **19**(8) (2008) 1468–1493.
11. M. Strickert and B. Hammer, Neural gas for sequences, in *Proc. Workshop on Self-Organizing Networks (WSOM)*, Kyushu Institute of Technology (2003), pp. 53–57.
12. M. Varsta, J. D. R. Millán and J. Heikkonen, A recurrent self-organizing map for temporal sequence processing, in *Proc. 7th Int. Conf. Artificial Neural Networks, ICANN '97* (Springer-Verlag, London, UK, 1997), pp. 421–426.
13. T. Voegtlin, Recursive self-organizing maps, *Neural Network.* **15** (2002) 979–991.
14. M. Hagenbuchner, R. Sperduti and A. C. Tsoi, A self-organizing map for adaptive processing of structured data, *IEEE Trans. Neural Networks* **14** (2003) 491–505.
15. H. Sakoe, Dynamic programming algorithm optimization for spoken word recognition, *IEEE Trans. Acoustics, Speech, Signal Process.* **26** (1978) 43–49.
16. P. Somervuo, Competing hidden markov models on the self-organizing map, *IEEE — INNS — ENNS Int. Joint Conf. Neural Networks* **3** (2000) 3169.

17. P. Somervuo and T. Kohonen, Self-organizing maps and learning vector quantization for feature sequences, *Neural Process. Lett.* **10**(2) (1999) 151–159.
18. C. Ferles and A. Stafylopatis, Sequence clustering with the self-organizing hidden markov model map, *8th IEEE Int. Conf. BioInformatics and BioEngineering, 2008. BIBE 2008.* October 2008, pp. 1–7.
19. C. Ferles and A. Stafylopatis, A hybrid self-organizing model for sequence analysis, in *ICTAI'08: Proc. 2008 20th IEEE Int. Conf. Tools with Artificial Intelligence* (Washington, DC, USA, 2008. IEEE Computer Society), pp. 105–112.
20. C. M. Bishop, M. Svensén and C. K. I. Williams, GTM: The generative topographic mapping, *Neural Comput.* **10**(1) (1998) 215–234.
21. C. M. Bishop, G. E. Hinton and I. G. D. Strachan, Gtm through time, in *IEE Fifth Int. Conf. Artificial Neural Networks* (1997), pp. 111–116.
22. D. Bacciu, A. Micheli and A. Sperduti, Compositional generative mapping of structured data, in *Proc. Int. Joint Conf. Neural Networks, IJCNN'10* (2010), pp. 1–8.
23. I. Olier and A. Vellido, Advances in clustering and visualization of time series using gtm through time, *Neural Network* **21** (2008) 904–913.
24. J. J. Verbeek, N. Vlassis and B. J. A. Krose, Self-organizing mixture models, *Neurocomputing* **63** (2005) 99–123.
25. S. Bengio and Y. Bengio, An EM algorithm for asynchronous input/output hidden Markov models, in *Int. Conf. Neural Information Processing*, L. Xu, ed. (Hong-Kong, 1996), pp. 328–334.
26. Y. Bengio and P. Frasconi, An input output hmm architecture, in *NIPS* (1994), pp. 427–434.
27. Z. Ghahramani and M. I. Jordan, Factorial hidden markov models, *Mach. Learn.* **29**(2–3) (1997) 245–273.
28. D. Bouchaffra, Embedding hmm's-based models in a euclidean space: The topological hidden markov models, in *ICPR08* (2008), pp. 1–4.
29. H. Kunsch, S. Geman and A. Kehagias, Hidden Markov random fields, *The Ann. Appl. Probab.* **5**(3) (1995) 577–602.
30. A. P. Dempster, N. M. Laird and D. B. Rubin, Maximum likelihood from incomplete data via the em algorithm, *Roy. Statist. Soc.* **39**(1) (1997) 1–38.
31. C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA, 2006).
32. S. P. Luttrel, A bayesian analysis of self-organizing maps, *Neural Comput.* **6** (1994) 767–794.
33. L. R. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, *Proc. IEEE* **77**(2) (1989) 257–286.
34. E. M. Airoldi, Getting started in probabilistic graphical models, *PLoS Comput. Biol.* **3**(12) (2007) e252.
35. L. E. Baum, An inequality and associated maximization technique in statistical estimation for probabilistic functions of markov processes, *Inequalities* **3** (1972) 1–8.
36. G. D. Forney, The viterbi algorithm, *Proc. IEEE* **61** (1973) 268–278.
37. A. Asuncion and D. J. Newman, UCI machine learning repository. http://www.ics.uci.edu/∼mlearn/MLRepository.html (2007).
38. B. H. Williams, *Extracting Motion Primitives from Natural Handwriting Data.* Ph.D. thesis, Institute for Adaptive and Neural Computation School of Informatics. University of Edinburgh (2008).
39. J. Vesanto and E. Alhoniemi, Clustering of the self-organizing map, *IEEE Trans. Neural Networks* **11**(3) (2000) 586–600.