A Probabilistic Self-Organizing Map for Binary Data Topographic Clustering *

Mustapha Lebbah
LIPN-UMR 7030 Université Paris 13 - CNRS.
93430 Villetaneuse, France
lebbah@lipn.univ-paris13.fr

Younès Bennani
LIPN-UMR 7030 Université Paris 13 - CNRS.
93430 Villetaneuse, France
bennani@lipn.univ-paris13.fr

Nicoleta Rogovschi
LIPN-UMR 7030 Université Paris 13 - CNRS.
93430 Villetaneuse, France
rogovschi@lipn.univ-paris13.fr

Received (21/12/2007)
Revised (15/01/2008)

This paper introduces a probabilistic self-organizing map for topographic clustering, analysis and visualization of multivariate binary data or categorical data using binary coding. We propose a probabilistic formalism dedicated to binary data in which cells are represented by a Bernoulli distribution. Each cell is characterized by a prototype with the same binary coding as used in the data space and the probability of being different from this prototype. The learning algorithm, BeSOM, that we propose is an application of the EM standard algorithm. We illustrate the power of this method with six data sets taken from a public data set repository. The results show a good quality of the topological ordering and homogenous clustering.

Keywords: Self-organizing map; Bernoulli distribution; Binary data

1. Introduction

Data visualization is an important step in the exploratory phase of data analysis. This step is more difficult when it involves binary data and categorical variables [35; 38; 39; 40]. Self-organizing maps are being increasingly used as tools for visualization, as they allow projection over small areas that are generally two dimensional. The basic model proposed by Kohonen [27], was only designed for numerical data, but it has been successfully applied to treating textual data, [25]. This algorithm has also been applied to binary data following transformation of the original data [20; 29; 31]. Developing generative models

*This is the extended version of the paper: M.Lebbah, N. Rogovschi and Y. Bennani. Be SOM : Bernoulli on Self Organizing Map. International Joint Conferences on Neural Networks. IJCNN 2007, Orlando, Florida -August 12-17, (2007).
of the Kohonen map has long been an important goal. These models vary in the form of the interactions, and they assume the hidden generators may follow in generating the observations. Some extensions and reformulations of the Kohonen model have been described in the literature. They include probabilistic self-organizing maps [3] which define a map as a gaussian mixture and use the maximum likelihood approach to define an iterative algorithm.

In [41], the authors propose a probabilistic generalization of Kohonen’s SOM which maximizes the variational free-energy that sums data log-likelihood and Kullback-Leibler divergence between a normalized neighbourhood function and the posterior distribution on the given data for the components. We have also Soft topographic vector quantization (STVQ), which uses some measure of divergence between data items and cells to minimize a new error function [14; 17]. Another model, often presented as the probabilistic version of the self-organizing map, is the Generative Topographic Map (GTM) [5; 23]. However, the manner in which GTM achieves the topographic organization is quite different from those used in the SOM models. In GTM mixture components are parameterized by a linear combination of nonlinear functions of the locations of the components in the latent space. The GTM was developed for continuous data. A specific GTM model was subsequently developed for binary data by adopting a variational approximation to the binomial likelihood [12]. In [2], the authors propose hierarchical density-based clustering of categorical data. It builds a hierarchy representing the underlying cluster structure of the categorical data set. Also, in [24], the authors concentrate on modelling binary coded data where only the presence or absence of a variable is of interest. In contrast to other approaches, the model is linear. The model is seen as a Bernoulli analogue of the multinomial decomposition model. In [22], the main of the proposed method is to speed-up convergence of EM, and second to yield same results (or not so far) than traditional EM using categorical data. Others similar techniques have been developed to cluster large data sets [4; 18; 28; 32; 36].

Here, we concentrate on modelling qualitative data using binary coding. This model involves use of the probabilistic formalism of the topological map used in [3]; Therefore, it consists of estimating the parameters of the model by maximizing the likelihood of the data set. The learning algorithm that we propose is an application of the EM standard algorithm, [34]. Some variants are proposed to speed-up EM in reducing the time spent in the E-step in the case of categorical data, [22]. We show how to combine the benefits of SOMs, Hamming distance and mixture models to design a new Bernoulli mixture for binary data. This allows a prototype to be defined using the same binary coding as data space.

The rest of the paper is organized as follows: In section 2 we provide more information about binary coding and the similarity measure we use. In section 3, we present the BeSOM (Bernoulli on Self-Organizing Map) model and the associated EM learning algorithm. In section 4 we discuss the relation between our model and the traditional SOM using the hamming distance. In the section 6, we present two applications. The experiments involve handwritten numerals (0 − 9), and a zoo data set available in [6]. These data sets allow us to illustrate the use of this algorithm for visualizing different high dimensional qualitative variables using binary coding. Our conclusions are reported in section 7.
2. General information about a binary data

Very often, a binary vector represents a coding of discrete features which have a finite, usually small, number of possible values. Let $\beta^n = \{0,1\}^n$ be a binary data space and $\mathcal{A} = \{z_i; i = 1, \ldots, N\}$ a set of observations, where each observation $z_i = (z^1_i, z^2_i, \ldots, z^n_i)$ is a binary vector in $\beta^n$. Some of these variables, called ordinal variables, have an implicit order, the others are just nominal variables. The general coding used in order to obtain binary data are: (a) The additive binary coding: this coding respect the order existing between modalities, (see a table 1). (b) The disjunctive complete coding: which transforms each nominal feature using the disjonctif coding (see a table 1).

<table>
<thead>
<tr>
<th>Modalities</th>
<th>Additive coding</th>
<th>Disjunctive coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 0 0</td>
<td>1 0 0</td>
</tr>
<tr>
<td>2</td>
<td>1 1 0</td>
<td>0 1 0</td>
</tr>
<tr>
<td>3</td>
<td>1 1 1</td>
<td>0 0 1</td>
</tr>
</tbody>
</table>

Table 1. Binary coding of categorical variable with 3 modalities.

Euclidian distance is not adapted to binary data, it is often much more interesting to use an apropriate similarity index. In this paper, we introduce the Hamming distance $\mathcal{H}$ which allows comparison of the binary vectors $z_m$ and $z_l$. The Hamming distance measures the number of mismatches between $z_m$ and $z_l$:

$$\mathcal{H}(z_m, z_l) = \sum_{j=1}^{n} |z^j_m - z^j_l|$$

The Hamming distance allows the binary median center to be calculated; the most important characteristic in this case of the median center is a binary vector which has the same interpretation (same coding) as the observations in the data space.

For the following, we assume that each $z_i$ is taken with corresponding weight $\gamma_i$. By definition the median center of $\mathcal{A}$ is any point $w = (w^1, w^2, \ldots, w^n)$ included in $\beta^n$ minimizing the inertia of $\mathcal{A}$: $I(w) = \sum_{i=1}^{I} \gamma_i \mathcal{H}(z_i, w)$. Each component $w^j$ minimize $I(w^j) = \sum_{i=1}^{N} \gamma_i |z^j_i - w^j|$, which can be rewritten as: $I(w^j) = w^j \Gamma_0 + (1 - w^j) \Gamma_1$, where $\Gamma_0 = \sum_{i=1}^{N} \gamma_i (1 - z^j_i)$, represents the sum of weighted observations which the value is equal to 0, and $\Gamma_1 = \sum_{i=1}^{N} \gamma_i z^j_i$ represents the sum of weighted observations which the value is equal to 1. Thus, to find the median value $w^j$ which minimizes $I(w^j)$, we select $w^j \in \{0,1\}$ so that $I(w^j)$ have minimum value $\Gamma_0$ or $\Gamma_1$. Hence, we take $w^j = 1$ if $\Gamma_1 > \Gamma_0$ and $w^j = 0$ if $\Gamma_1 < \Gamma_0$. This rule is simplified when the weights is identical for all variables: $w^j$ is the value 0 or 1 most often chosen by the observations on the variable $j$.
3. Binary data and Probabilistic self-organizing map

As with a traditional self-organizing map, we assume that the lattice $C$ has a discrete topology (discrete output space) defined by an undirected graph. Usually, this graph is a regular grid in one or two dimensions. We denote the number of cells in $C$ as $N_{cell}$. For each pair of cells $(c, r)$ on the map, the distance $\delta(c, r)$ is defined as the length of the shortest chain linking cells $r$ and $c$. The BeSOM model is based on the probabilistic formalism of topological maps. Following the bayesian formalism, presented in [3; 33], each cell $c$ models a mixture of probability distributions (figure 1).

3.1. General probabilistic formalism

We assume that each observation $z$ is generated by the following process: We start by associating to each cell $c \in C$ a probability $p(z/c)$ where $z$ is a vector in the data space. Next, we pick a cell $c^*$ from $C$ according to the prior probability $p(c^*)$. For each cell $c^*$, we select an associated cell $c \in C$ following the conditional probability $p(c/c^*)$. All cells $c \in C$ contribute to the generation of $z$ with $p(z/c)$ 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 $c$ to the generation of $z$ is high.

Due to the "Markov" property, $p(z/c, c^*) = p(z/c)$, the probability distribution of the observations generated by a cell $c^*$ of $C$ is a mixture $p_{c^*}(z/c^*)$ of probabilities completely defined from the map as:

$$p_{c^*}(z/c^*) = \sum_{c \in C} p(c/c^*) p(z/c)$$

The generative model considers the mixture of probabilities, given by:

$$p(z) = \sum_{c^* \in C} p(c^*) p_{c^*}(z/c^*)$$

(2)

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

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

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

3.2. BeSOM Model

In the following, we assume that our $n$ binary variables $z = (z^1, z^2, ..., z^k, ..., z^n)$ are independent, and so the conditional distribution in $p(z/c)$ is now given as the product
of univariate single distributions \( p(z/c) = \prod_{k=1}^{n} p(z^k/c) \). We will use the Hamming distance in the Bernoulli distribution: \( p(z/c) = p(z/c, w_c) = f_c(z, w_c, \epsilon_c) \) which admits parameters \( w_c = (w^1_c, w^2_c, ..., w^n_c) \) \( \epsilon_c = (\epsilon^1_c, \epsilon^2_c, ..., \epsilon^n_c) \). Each probability \( \epsilon^k_c \in [0, \frac{1}{2}] \) is associated to component \( w^k_c \in \{0, 1\} \) [13]. The parameter \( \epsilon^k_c \) defines the probability that a variable \( z^k \) is different from this component \( w^k_c \in \{0, 1\} \). Thus, the distribution associated with each cell \( c \in C \) is defined as follows: \( f_c(z, w_c, \epsilon_c) = \prod_{k=1}^{n} f_c(z^k, w^k_c, \epsilon^k_c) \), where

\[
f_c(z^k, w^k_c, \epsilon^k_c) = (\epsilon^k_c)^{|z^k-w^k|}(1-\epsilon^k_c)^{1-|z^k-w^k|}
\]

(3)

We can assume that the parameter \( \epsilon^k_c \) depends only on a cell \( c \in C \). Below, we develop the general model in which the probability depends on the cell \( c \) and binary variable \( z^k \). Using this expression, the model mixture generator (2) becomes:

\[
p(z) = \sum_{c' \in C} p(c') \sum_{c \in C} p(c/c') f_c(z, w_c, \epsilon_c)
\]

(4)

Therefore, the parameters \( \theta = \theta^c \cup \theta^{c*} \) which define the model mixture generator (4) are constituted of the parameters of the Bernoulli distribution \( \theta^c = \{\theta^c, c = 1..N_{cell}\} \), where \( \theta^c = (w_c, \epsilon_c) \), and all the prior probabilities, also called mixing coefficients \( \theta^{c*} = \{\theta^{c*}, c' = 1..N_{cell}\} \), where \( \theta^{c*} = p(c'|c) \). The difficulty now is to define the cost function and the learning algorithm for estimating all these parameters dedicated to binary data.

### 3.3. Cost function and optimization algorithm

The learning algorithm is based on maximizing the likelihood of the observations by applying the EM algorithm. Learning is facilitated by introducing \( N \) hidden variables \( \Xi = (\xi_1, ..., \xi_N) \). Hidden variable \( \xi = (c, c^*) \) indicates which of the cell pairs \( c \) and \( c^* \) generate the corresponding data observation \( z \). We introduce the hidden variable \( \xi = (c, c^*) \) in expression (4):

\[
p(z) = \sum_{\xi \in C \times C} p(z, \xi) = \sum_{c, c^* \in C} p(c^*)p(c/c^*)f_c(z, w_c, \epsilon_c)
\]

(5)
We define a binary indicator variable $\alpha_i^{(c,c^*)}$ which indicates the hidden generator that may follow in generating the observation $z_i$ as: $\alpha_i^{(c,c^*)} = \begin{cases} 1 & \text{for } \xi = (c,c^*) \\ 0 & \text{otherwise} \end{cases}$. Using expression (5), and the binary indicator $\alpha_i^{(c,c^*)}$, we can define the likelihood of the observations using the hidden variables as follows:

$$L^T(A, \Xi; \theta) = \prod_{i=1}^{N} \prod_{c^* \in C} \prod_{c \in C} \left[ \theta^{c^*} p(c|c^*) f_c(z, w_c, \epsilon_c) \right] \alpha_i^{(c,c^*)}$$

The log-likelihood becomes:

$$\ln L^T(A, \Xi; \theta) = \sum_{z_i \in A} \sum_{c^* \in C} \sum_{c \in C} \alpha_i^{(c,c^*)} \left[ \ln(\theta^{c^*}) + \ln \left( \frac{K^T(\delta(c^*, c))}{T_{c^*}} \right) + \ln(f_c(z, w_c, \epsilon_c)) \right]$$

where $T_{c^*} = \sum_{r \in C} K^T(\delta(r, c^*))$.

The application of the EM algorithm [9] for the maximization of log-likelihood requires $Q^T(\theta^t, \theta^{t-1})$ to be maximised for a fixed temperature $T$ defined as:

$$Q^T(\theta^t; \theta^{t-1}) = E \left[ \ln L^T(A, \Xi; \theta^t) / A, \theta^{t-1} \right],$$

where $\theta^t$ is the set of the parameters estimated at the $t^{th}$ step of the learning algorithm. However, the E-step calculates the expectation of log-likelihood with respect to the hidden variable while maintaining the established parameter $\theta^{t-1}$. During the M-step, after updating $Q^T(\theta^t, \theta^{t-1})$ from the previous step, we maximize the $Q^T(\theta^t, \theta^{t-1})$ with respect to $\theta^t$, ($\theta^t = \arg \max_{\theta^t} (Q^T(\theta^t, \theta^{t-1}))$). Two step optimization increases the function $Q^T(\theta^t, \theta^{t-1})$: $\ln L^T(A, \theta^t) \geq \ln L^T(A, \theta^{t-1})$, [7; 9]. Thus, the function $Q^T(\theta^t, \theta^{t-1})$ is defined as:

$$Q^T(\theta^t, \theta^{t-1}) = \sum_{z_i \in A} \sum_{c^* \in C} \sum_{c \in C} E(\alpha_i^{(c,c^*)} / z_i, \theta^{t-1}) \left[ \ln(\theta^{c^*}) + \ln \left( \frac{K^T(\delta(c^*, c))}{T_{c^*}} \right) + \ln(f_c(z, w_c, \epsilon_c)) \right]$$

The variable $\alpha_i^{(c,c^*)}$ is a Bernoulli variable. Thus, the expectation $E(\alpha_i^{(c,c^*)} / z_i, \theta^{t-1}) = p(\alpha_i^{(c,c^*)} = 1 / z_i, \theta^{t-1}) = p(c, c^* / z_i, \theta^{t-1})$, where

$$p(c, c^* / z_i, \theta^{t-1}) = \frac{p(c^*) p(c|c^*) p(z|c)}{p(z)}$$

Thus, the function $Q^T(\theta^t, \theta^{t-1})$ becomes:

$$Q^T(\theta^t, \theta^{t-1}) = Q_1^T(\theta^{c*}, \theta^{t-1}) + Q_2^T(\theta^{c*}, \theta^{t-1}) + Q_3^T(\theta^{t-1})$$

where

$$Q_1^T(\theta^{c*}, \theta^{t-1}) = \sum_{k=1}^{n} \sum_{z_i \in A} \sum_{c^* \in C} \sum_{c \in C} p(c, c^* / z_i, \theta^{t-1}) \ln(f_c(z^k, w_c^k, \epsilon_c^k))$$
A Probabilistic Self-Organizing Map for Binary Data Topographic Clustering

\[ Q_T^T(\theta^*, \theta^{t-1}) = \sum_{z_i \in A} \sum_{c^* \in C} \sum_{c \in C} p(c, c^*/z_i, \theta^{t-1}) \ln(\theta^*) \]

\[ Q_T^T(\theta^{t-1}) = \sum_{z_i \in A} \sum_{c^* \in C} \sum_{c \in C} p(c, c^*/z_i, \theta^{t-1}) \ln \left( \frac{K_T(\delta(c^*, c))}{T_c} \right) \]

The parameters \( \theta^C \) and \( \theta^{C^*} \) indicate the parameters estimated at the \( t \)th step. We observe that the objective function (6) \( Q_T^T(\theta^t, \theta^{t-1}) \) is defined as a sum of three terms. The first term \( Q_T^T(\theta^C, \theta^{t-1}) \) depends on \( \theta^c,k = (w^k_c, \epsilon^k_c) \); the second term \( Q_T^T(\theta^{C^*}, \theta^{t-1}) \) depends on \( \theta^{c^*} \), and the third term is constant. Maximizing \( Q_T^T(\theta^t, \theta^{t-1}) \) with respect to \( \theta^{c^*} \) and \( \theta^c \) can be performed separately including the parameter \( w_c \) and \( \epsilon_c \). The maximization of \( Q_T^T(\theta^t, \theta^{t-1}) \) leads to the updates that are calculated using the parameters estimated at the \( t-1 \)th step. A short development is presented in appendix 1. The expressions are defined as follows:

\[ \theta^* = p(c^*) = \frac{\sum_{z_i \in A} p(c^*/z_i, \theta^{t-1})}{N} \quad (7) \]

Each component of \( w_c = (w^1_c, ..., w^k_c, ..., w^n_c) \) and \( \epsilon_c = (\epsilon^1_c, \epsilon^2_c, ..., \epsilon^k_c, ..., \epsilon^n_c) \) is then computed as follows:

\[ w^k_c = \begin{cases} 0 & \text{if } \left[ \sum_{z_i \in A} p(c/z_i, \theta^{t-1}) (1 - z^k_i) \right] \geq \\ 1 & \text{otherwise} \end{cases} \quad (8) \]

\[ \epsilon^k_c = \frac{\sum_{z_i \in A} p(c/z_i, \theta^{t-1}) |z^k_i - w^k_c|}{\sum_{z_i \in A} p(c/z_i, \theta^{t-1})} \quad (9) \]

Where \( p(c^*/z_i, \theta^{t-1}) = \sum_{c \in C} p(c, c^*/z_i, \theta^{t-1}) \)

and \( p(c/z_i, \theta^{t-1}) = \sum_{c^* \in C} p(c, c^*/z_i, \theta^{t-1}) \).

In practice the EM algorithm starts with an initial value of \( \theta^0 \). In our case we use the clusters provided by the BinBatch model described in section 4.1. Thus, the application of E-M for the maximization gives rise to the iterative algorithm of BeSOM. The version of the BeSOM algorithm for a fixed \( T \) parameter is presented in the following way:
Principal stages of the learning algorithm BeSOM.

1. **Initialization** (iteration $t = 0$) Choose the initial parameters $(\theta^0)$ and the number of iterations $N_{\text{iter}}$.

2. **Basic Iteration at a constant $T$** (iteration $t \geq 1$) Calculate all the parameters $\theta^t = \{\theta^c, w_c, \epsilon_c\}$ from the previous parameters $\theta^{t-1}$ associated with each cell $c$ and $c^*$ by applying the formulas: (7), (8) and (9).

3. **Repeat** the basic iteration until $t > N_{\text{iter}}$.

The BeSOM learning algorithm allows us to estimate the parameters maximizing the log-likelihood function for a fixed $T$. As in the SOM algorithm, we decrease the value of $T$ between two values $T_{\text{max}}$ and $T_{\text{min}}$, to control the size of the neighbourhood influencing a given cell on the map. For each $T$ value, we get a likelihood function $L_T$, and therefore the expression varies with $T$. When decreasing $T$, the learning algorithm of BeSOM will be defined in the following way:

**Algorithm BeSOM varying $T$**

1. **Initialization Phase** (iteration $t = 0$): Choose $T_{\text{max}}$, $T_{\text{min}}$ and $N_{\text{iter}}$. Apply the principal stages of BeSOM algorithm described above for the value of $T$ fixed to $T_{\text{max}}$.

2. **Iterative step**: We assume that the previous parameter $\theta^t$ are known. Compute the new value of $T$ by applying the following formula: $T = T_{\text{max}} \left( \frac{T_{\text{max}}}{T_{\text{min}}} \right)^{t/N_{\text{iter}}}$. For fixed value of the parameter $T$, apply the basic iteration described in the principal stages, which estimates the new parameter $\theta^{t+1}$ using the formulas (7), (8) and (9).

3. **Repeat** the Iterative step while $t \leq N_{\text{iter}}$.

We can define two steps in the operating of the algorithm:

- The first step corresponds to high $T$ values. In this case, the influencing neighbourhood of each cell $c$ on the map is important and corresponds to higher values of $K_T(\delta(c,r))$. Formulas (7), (8) and (9) use a high number of observations to estimate model parameters. This step provides the topological order.
- The second step corresponds to small $T$ values. The number of observations in formulas (7), (8) and (9) is limited. Therefore, the adaptation is very local. The parameters are accurately computed from the local density of the data.

4. **BeSOM and the traditional SOM algorithm**

4.1. **Binary SOM**

Let us consider the batch SOM algorithm proposed by Kohonen. This algorithm was developed for continuous data, but a specific SOM model has been developed for binary data using the similar cost function [29]. Using the Hamming distance defined in expression (1),
the cost function in this case can be expressed as:

\[ G(w, \phi) = \sum_{z_i \in \mathcal{A}} \sum_{r \in C} K^T(\delta(\phi(z_i), r)) \mathcal{H}(z_i, w_r) \]  

Where \( \phi \) assigns each observation \( z \) to a single cell in \( C \). This model was called BinBatch [29]. The cost function (10), is minimized using an iterative process with two steps. Assignment step which leads to the use of the following assignment function:

\[ \forall z, \phi(z) = \arg \min_c \mathcal{H}(z, w_c). \]  

Optimization step which allows us to define the median center \( w_c \in \beta^n \) of the observations \( z_i \in \mathcal{A} \) weighted by \( K^T(\delta(c, \phi(z_i))) \). Each component of \( w_c = (w_c^1, ..., w_c^K, ..., w_c^n) \) is then computed as follows:

\[ w_c^k = \begin{cases} 0 & \text{if } \left[ \sum_{z_i \in \mathcal{A}} K^T(\delta(c, \phi(z_i))) \right] (1 - z_i^k) \geq \left[ \sum_{z_i \in \mathcal{A}} K^T(\delta(c, \phi(z_i))) z_i^k \right] \\ 1 & \text{otherwise} \end{cases} \]

Note that the update for the median center \( w_c \) in our BeSOM model coincides with the BinBatch model in which each datum \( z \) is weighted proportionally to the neighbourhood function centered at the winning prototype for that data vector and evaluated at the prototype \( w_c \). In both models we minimize the inertia of observation \( z \) in binary space using the weighted Hamming distance. This minimization is made by computing the binary median center \( w_c \) which uses the same binary coding as in the binary data space.

Note that the definition of the winner is different. In our new BeSOM model the assignment step is at the end of learning algorithm. Note also that there is a close link between the BinBatch model and mixture modelling. At each step we compute \( p(c|z) \) which is used to weight the observation \( z \). In BinBatch, we simplify the assignment by selecting so as to minimize \( \mathcal{H}(z, w_c) \). Hence, if we assume that \( \varepsilon \) is the same parameter for all cells and for all binary variables, and if the prior probabilities \( p(c^*) = \frac{1}{N_{cell}} \) are equal, the low Hamming distance is similar to state that the probability \( p(z|c) = \varepsilon^{\mathcal{H}(z, w_c)} (1 - \varepsilon)^{n - \mathcal{H}(z, w_c)} \) of finding \( z \) given a parameters \( (w_c, \varepsilon) \) of \( c \) is quite high. Thus, we infer in this case that the posterior probabilities \( p(c|z) \) and \( p(c^*|z) \) become high.

Under the conditions defined above, the maximization of the objective function \( Q^T(\theta^t, \theta^{t-1}) \) is the same as maximizing the cost function \( Q_1^T(\theta^c, \theta^{t-1}) \) depending on \( \theta^c = (w_c, \varepsilon) \) alone, which is rewritten as follows:

\[ Q_1^T(\theta^c, \theta^{t-1}) = \ln\left(\frac{1 - \varepsilon}{\varepsilon}\right) \sum_{c \in C} \sum_{z_i \in \mathcal{A}} \left[ -p(c|z_i, \theta^{t-1}) \mathcal{H}(z_i, w_c) \right] \\
+ \ln(1 - \varepsilon) \sum_{c \in C} \sum_{z_i \in \mathcal{A}} \left[ np(c|z_i, \theta^{t-1}) \right] \]

Maximizing \( Q_1^T(\theta^c, \theta^{t-1}) \) with respect to \( \varepsilon \) involves computing the derivative of \( Q_1^T(\theta^c, \theta^{t-1}) \) with respect to \( \varepsilon \) and setting it to zero, \( \frac{\partial Q_1^T(\theta^c, \theta^{t-1})}{\partial \varepsilon} = 0 \). Thus, we can compute the new parameters \( \varepsilon \) at the \( t^{th} \) step as follows:

\[ \varepsilon = \sum_{c \in C} \sum_{z_i \in \mathcal{A}} p(c|z_i, \theta^{t-1}) \mathcal{H}(z_i, w_c) \]
Maximizing $Q_T^1(\theta^c, \theta^{t-1})$ with respect to $w_c$ requires minimizing the cost function $G(w)$ defined as follows:

$$G(w) = \sum_{c \in C} \sum_{z_i \in A} p(z_i / c^*, \theta^{t-1}) H(z_i, w_c)$$

(11)

BinBatch tends to make all cells equally important and maximize the cost function $Q(\theta^t, \theta^{t-1})$. If we assume that $c^*$ is fixed with an assignment function, the cost function $G(w, \phi)$ (10) is a simplification of the cost function $G(w)$ (11) under conditions that applied previously. Thus, the BeSOM model provides more information (parameters) than the BinBatch model.

4.2. Probabilistic SOM

The Generative Topographic Map (GTM) is often presented as the probabilistic version of the self-organizing map. However, the way in which GTM achieves a topographic organization is quite different from that used in SOM models. The parameters of the BeSOM model, like those for GTM, are estimated by the E-M algorithm to maximize the data log-likelihood. The original GTM model is presented for Gaussian mixtures. The modified GTMs for specific data observation use mixtures which are members of the exponential family [12; 23]. We are not aware of any methods for extending the GTM to mixtures of component densities outside of the exponential family.

Our BeSOM algorithm was inspired by a probabilistic SOM model proposed by [3]; this model defines a conditional probability $p(z_i / c)$ as Gaussian distribution with mean vector $w_c$ and covariance matrix $\Sigma_c = \sigma_c^2 I$. The authors use the Maximum A Posterior (MAP) method and develop an iterative batch algorithm to minimize the log-likelihood of observation defined as follows:

$$L(\phi, w, \sigma) = \sum_{z_i \in A} - \ln \left( \sum_{c \in C} K^T(\delta(\phi(z_i), c)) f(z_i, w_c, \sigma_c) \right)$$

The MAP method assumes that each observation $z_i$ is generated by a well-defined local mixture $p_{\phi^*}(z_i / c^*)$, and uses $\phi(z_i) = \arg \max_{c^*} p_{\phi^*}(z_i / c^*)$ as the assignment function for assigning each $z_i$ to its local mixture $p(z_i / \phi(z_i))$. Using the probabilistic model presented in section 3.1 and the Gaussian mixture, we can define an E-M algorithm to estimate a parameter of Gaussian distribution.

Thus, our probabilistic model becomes a general model for continuous and binary data. The model we propose provides a parameter $w_c$ in the same space as the input data.

5. Computational cost

Our algorithm is inspired from a probabilistic SOM model proposed by [3] which defines a conditional probability $p(z_i / c)$ as Gaussian distribution and the learning algorithm has $O(N N_{cell}^2)$ run-time. The computational cost of our model BeSOM is the same with previous. It is $N_{cell}$ slower than Kohonen’s SOM or BinBatch and possibly more slow in
large data set. However, by restricting the neighborhood of \( c^* \) to limit a number of pairs of cell \((c, c^*)\), we can obtain a low computational cost. Another way to reduce the computational cost is to use classification EM which is called CEM \[8\]. Instead of assigning the observations at the end of training, we introduce an assignment step to restrict the number of pairs \((c, c^*)\). Note that at the end of training, our algorithm BeSOM with small temperature \( T \) is very similar to probabilistic K-means using Hamming distance proposed in \[35\].

Finally we can reduce also the computational cost by assuming that the parameter \( \epsilon_c \) of Bernoulli distribution (expression 3) depends only on a cell \( c \in C \), which is \( \epsilon_c = \epsilon_c \). Thus the parameter \( \epsilon_c \) for each cell of the model which is denoted BeSOM-\(\epsilon\) is defined as follows:

\[
\epsilon_c = \frac{\sum_{z_i \in A} p(c/z_i, \theta^{t-1}) \mathcal{H}(z_i, w_c)}{\sum_{z_i \in A} np(c/z_i, \theta^{t-1})}
\]

where \( n \) is the dimension of the observation. In the rest of the paper we use the general model which estimate the probability vector \( \epsilon_c = (\epsilon^1_c, \epsilon^2_c, ..., \epsilon^k_c, ..., \epsilon^n_c) \).

6. Experimental validations

To evaluate the quality of clustering, we adopt the approach of comparing the results to a "ground truth". We use the clustering accuracy for measuring the clustering results. This a common approach in the general area of data clustering. In general, the result of clustering is usually assessed on the basis of some external knowledge about how clusters should be structured. This may imply evaluating separation, density, connectedness, and so on. The only way to assess the usefulness of a clustering result is indirect validation, whereby clusters are applied to the solution of a problem and the correctness is evaluated against objective external knowledge. This procedure is defined by \[21\] as "validating clustering by extrinsic classification", and has been followed in many other studies \[1; 26\]. We feel that this approach is reasonable one if we don’t want to judge clustering results by some cluster validity index, which is nothing but a bias toward some preferred cluster property (e.g., compact, or well separated, or connected). Thus, to adopt this approach we need labeled data sets, where the external (extrinsic) knowledge is the class information provided by labels. Hence, if the BeSOM finds significant clusters in the data, these will be reflected by the distribution of classes. Therefore we operate a vote step for clusters and compare them to the behavior methods from the literature. The so-called vote step consists in the following. For each cluster \( c \in C \):

- Count the number of observation of each class \( l \) (call it \( N_{cl} \)).
- Count the total number of observation assigned to the cell \( c \) (call it \( N_c \)).
- Compute the proportion of observations of each class (call it \( S_{cl} = N_{cl}/N_c \)).
- Assign to the cluster the label of the most represented class (\( l = \arg \max_l (S_{cl}) \)).

A cluster \( c \) for which \( S_{cl} = 1 \) for some class labeled \( l \) is usually termed a "pure" cluster, and a purity measure can be expressed as the percentage of elements of the assigned class in a cluster. The experimental results are then expressed as the fraction of observations
falling in clusters which are labeled with a class different from that of the observation. This quantity is expressed as a percentage and termed "error percentage" (indicated as \( \text{Err\%} \) in the results). Regarding the evaluation method, we choose not to perform cross-validation or similar procedures, considering that the algorithm is trained in a completely unsupervised manner, and calibration already occurs (in a sense) on an external validation data set, that is the set of class labels. Cross-validation or resampling methods, however, could be very useful to assess the stability of the proposed method, by comparing clustering structures in repeated experiments.

6.1. Zoo data

This example is taken from [6]. We use this simple data set to show the good performance of the BeSOM algorithm. The data set contains 101 animals described with 16 qualitative variables: 15 of the variables are binary and one is numeric with 6 possible values. Each animal is labelled 1 to 7 according to its class. Using disjunctive coding for the qualitative variable with 6 possible values, defined in table 2, the data set consists of a \( 101 \times 21 \) binary data matrix. All 101 animals are used for learning with a map with the dimensions \( 5 \times 5 \) cells. The learning algorithm provides a parameter of Bernoulli distribution for each cell; this parameter is \( w_c = (w_c^1, w_c^2, ..., w_c^k, ..., w_c^{21}) \in \beta^n \) and the probability vector \( \epsilon_c = (\epsilon_c^1, \epsilon_c^2, ..., \epsilon_c^k, ..., \epsilon_c^{21}) \).

At the end of the learning phase, each observation, corresponding to an animal, is assigned to the cell with the highest posterior probability \( p(c/z) \). We use the names used in original data set, [6]. To visualize the coherence of the map with the labelling of animals, figure 2 shows the class number corresponding to each cell after the application of the majority vote rule in each cell. The map shows that the same class of animals is assigned to cells close to each other. We can observe that the insects which have label "6" are assigned to the bottom right of the map. The same type of analysis can be applied to the remaining clusters. Figure 3 displays the objective function \( Q^T(\theta^t, \theta^{t-1}) \) (expression 6) without computing the constant term \( Q^T_3(\theta^{t-1}) \) and varying \( T \) from \( T_{max} = 2 \) to \( T_{min} = 0.5 \). We

<table>
<thead>
<tr>
<th>Modalities</th>
<th>Binary code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1 0 0 0 0</td>
</tr>
<tr>
<td>2</td>
<td>0 1 0 0 0</td>
</tr>
<tr>
<td>3</td>
<td>0 0 1 0 0</td>
</tr>
<tr>
<td>4</td>
<td>0 0 0 1 0</td>
</tr>
<tr>
<td>5</td>
<td>0 0 0 0 1</td>
</tr>
<tr>
<td>6</td>
<td>0 0 0 0 0</td>
</tr>
</tbody>
</table>
Fig. 2. 5×5 BeSOM map. We display the animals associated with the highest probabilities. The number between parentheses indicates the class label of each cell after application of the majority vote rule also show the term $Q_T^T(\theta^c, \theta^{t-1})$ and $Q_T^2(\theta^{c^*}, \theta^{t-1})$. The increase (maximization) of the objective function is apparent.

6.2. Handwritten data

This experiment concerns a data set consisting of the handwritten numerals ('0'-'9') extracted from a collection of Dutch utility maps, [6]. There are 200 samples of each digit such that there is a total of 2000 samples. Each sample is a $15 \times 16$ binary pixel image. The data set consisted of a 2000 × 240 binary data matrix. Each qualitative variable is a pixel with two possible values "On=1" and "Off=0". Using the BeSOM algorithm, each cell is associated to a Bernoulli distribution with $w_c = (w^1_c, w^2_c, ..., w^k_c, ..., w^{240}_c) \in \beta^n$ and a probability vector $\epsilon_c = (\epsilon^1_c, \epsilon^2_c, ..., \epsilon^k_c, ..., \epsilon^{240}_c)$ where $n = 240$. Figure 4 shows the parameters estimated by our BeSOM model. Figure 4.(a) shows the prototype vector $w_c \in \beta^n$ which is displayed as a $15 \times 16$ binary pixel image. Figure 4.(b) shows the parameter vectors $\epsilon_c$ as a $15 \times 16$ pixel image. Thus, each pixel defines the probabilities $\epsilon^k_c$ of being different from the binary variable $w^k_c$ associated with the prototype vector $w_c$ displayed in figure 4.(a). We observe that these probabilities $\epsilon^k_c$ are high in pixels corresponding to the contour...
Fig. 3. The development of the objective function $Q_T(\theta^t, \theta^{t-1})$ with iterations in the Zoo data set. The dashed line shows the function $Q_T^1(\theta^C, \theta^{t-1})$. The dash-dot line shows the function $Q_T^2(\theta^C, \theta^{t-1})$ associated with each image (number). In both figures, the clear topological organization of the prototype images is clear.

Fig. 4. BeSOM $16 \times 16$ map. (a) Each image is the prototype vector $w_c = (w_{1c}, w_{2c}, ..., w_{k_c}, ..., w_{240c})$ which is a binary vector. The prototype images are topologically well organized. The white pixel indicates “0” and the black pixel indicates “1”. (b) Each image is the probability vector $\epsilon_c = (\epsilon_{1c}, \epsilon_{2c}, ..., \epsilon_{k_c}, ..., \epsilon_{240c})$. Note the clear topological organization of the reference images with smooth changes. The grey shading of each pixel is proportional to the probability of being different from the estimated prototype $w_c$ (figure 4).a

In the purpose to see the influence of the choose of Bernoulli distribution, we make training of BeSOM-$\epsilon$ which assume that the parameter $\epsilon_c$ depends only on a cell $c \in C$. Instead of estimating probability vector $\epsilon_c$ we estimate a scalar probability $\epsilon_c$ (expression 12). Figure 5 shows the parameters estimated by our BeSOM-$\epsilon$ model. Figure 5.(a) shows the prototype vector $w_c \in \beta''$ which is displayed as a $15 \times 16$ binary pixel image. The white
A Probabilistic Self-Organizing Map for Binary Data Topographic Clustering

pixel indicates "0" and the black pixel indicates "1". The clear topological organization of the prototype images is also clear. Figure 5.(b) shows the parameter $\varepsilon_c$ for each cell $c \in C$. Thus, the grey shading of each cell is proportional to the probabilities $\varepsilon_c$ of being different from the prototype binary vector $w_c$ displayed in figure 5.(a).

Clearly, we can see that the general model BeSOM presented in figure 4 is more informative than the reduced model BeSOM-$\varepsilon$ (figure 5). The model BeSOM-$\varepsilon$ is interesting when we deal with very large data set. We give global view of homogeneous clustering, we compute the error classification rate. Thus, we find for BeSOM model $Err = 6.06\%$ and $Err = 8.01\%$ for second model BeSOM-$\varepsilon$. In the following we use only the general model BeSOM.

In [12] the author presents GTM model dedicated to binary data. He uses the same digit data set and proves that the dedicated GTM provides a clear topographic ordering of the prototypes on $16 \times 16$ map. It is not obvious to compare our result to GTM dedicated to binary data. BeSOM and GTM provide a topological map with clear topological organization. The way in which GTM achieves a topographic organization is quite different from that used in BeSOM model. Although our BeSOM model handles data in the same binary input space.

Fig. 5. BeSOM-$\varepsilon$ $16 \times 16$ map. In figure (a) each image is the prototype vector $w_c = (w_1^c, w_2^c, ... , w_k^c, ... , w_{240}^c)$ which is a binary vector. The prototype images are topologically well organised. The white pixel indicates "0" and the black pixel indicates "1". In figure (b) each image is the probability $\varepsilon_c$. The grey shading of each cell is proportional to the probability of being different from the estimated prototype $w_c$ (figure(b)).
6.3. Synthetic and real data

6.4. Synthetic data set

In order to compare the performances of BeSOM with other traditional unsupervised clustering algorithms, we use binary data available on the web site www.wu-wien.ac.at/am, [10; 11; 31]. We extract from the benchmark two different data bases made of 6000 individuals for which clustering algorithms have been run using the full data set. These data are artificial simulated in order to compare several classical unsupervised clustering algorithms, they mimic typical situations from tourism marketing. The tourists are classified in 6 classes according to their answer("Yes" or "No") to twelve questions. we select the two difficult problems according to the performances given by the computed Bayes classifier (scenario 5ind (Theoretical Bayes Rate, TBR=89%), scenario 5dep (Theoretical Bayes Rate, TBR=79%)). Two 2-D topological maps with $5 \times 5$ cells were trained using BeSOM. The full data base was used for comparison purpose.

The comparison with different clustering algorithms (Neural Gas NGAS-ED or NGAS-AD, K-means, Self-Organizing Map SOM, are presented in table 3. This comparison shows that BeSOM allows to approach the theoretical limit provided by the Bayes classifier. We have demonstrated that BeSOM performs well for these data sets when the traditional clustering algorithms like SOM and k-means, completely fail.

Table 3. Comparison of the classification performances reached by BeSOM and traditional clustering algorithms on the two simulated data sets (scenario 5ind and scenario 5dep). TBR: Theoretical Bayes Rate.

<table>
<thead>
<tr>
<th>Set/Methods</th>
<th>$k$-means</th>
<th>NGAS-ED</th>
<th>NGAS-H</th>
<th>SOM</th>
<th>BinBatch</th>
<th>BeSOM</th>
<th>TBR</th>
</tr>
</thead>
<tbody>
<tr>
<td>5ind</td>
<td>51%</td>
<td>71%</td>
<td>74%</td>
<td>51%</td>
<td>86%</td>
<td>86%</td>
<td>89%</td>
</tr>
<tr>
<td>5dep</td>
<td>48%</td>
<td>52%</td>
<td>59%</td>
<td>49%</td>
<td>65%</td>
<td>78%</td>
<td>79%</td>
</tr>
</tbody>
</table>

6.5. Real data set

To conduct another experimental comparison and to verify the efficacy of our proposed model, we compare the result provided in [26] using a version of K-modes clustering method dedicated to categorical data. We use the following two others data sets of categorical data obtained from UCI repository [6] and used in [26].

Wisconsin Breast Cancer Data

This data has 699 instances with 9 variable. Each data object is labeled as benign (458 or 65.5%) or malignant (241 or 34.5%). In our case, all variables are considered categorical with values 1, 2, ..., 10 and coded using binary coding.

Congressional Vote Data

This data set includes votes for each of the U.S. House of Representatives Congressmen on
the 16 key votes identified by the CQA. The CQA lists nine different types of votes: voted for, paired for, and announced for (these three simplified to YES), voted against, paired against, and announced against (these three simplified to NO), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition). All variables are Boolean with "Yes" and "No" values. A classification label of Republican or Democrat is provided with each observation. The dataset contains 435 observations with 168 Republicans and 267 Democrats.

Table 4 lists the classification error obtained with different methods. We compute the fraction of observations falling in clusters which are labeled with a class different from that of the observation. We can see that the results provided by K-modes [26] are bad comparing with our models. We may observe that the results are generally similar using the BinBatch and BeSOM model, although it better with BeSOM. As we see in sections above that BeSOM model provide more informations than the classical SOM using Hamming distance called BinBatch.

Among these data sets are used with some other algorithms and with the same quality index. In [16] authors propose an hierarchical clustering based on artificial ants and provides 12% as error classification. This algorithm is less efficient than our BeSOM but has the advantage of having a time computing much faster than our model BeSOM. In [37] the authors propose Shared Farthest Neighbors clustering technique, which is a hierarchical clustering algorithm based on a novel agglomeration principle. Using Wisconsin-B-C data set this method provide 5.60%. This method is motivated by a typical unsupervised problem in bioinformatics. We remind here that BeSOM model provides more clusters than the hierarchical models, but models based on SOM architecture provides more informations, especially BeSOM model which deals with binary data.

### 7. Conclusion and future work

This study reports the development of a computationally efficient EM approach to maximize the likelihood of the data set to estimate the parameters of a Bernoulli probabilistic self-organizing map model dedicated to qualitative or categorical variables using binary coding. This algorithm has the advantage of providing a prototype with the same coding as the input data. The BeSOM offer an original manner to deal with binary data. In our
view, the only drawback model is the computational expensive, but we see above different manners to reduce the computational cost. The simplicity of BeSOM might therefore preferred depending on the computational resources. We also show that through different means of visualization, the BeSOM algorithm gives various information that could be used in practical applications. All the topographic visualizations show that the topological order obtained can be used for meaningful interpretation of BeSOM clustering.

We conclude that BeSOM probabilistic model becomes a general model for continuous and binary data. BeSOM extension could be considered for data set having mixed data (continuous and binary) and the problems of the choice of the best model for each data set [1; 19; 30; 42]. Another possible extension could be possible towards unsupervised feature selection to select variables [15].
Appendix

Maximization of the objective function

The function (6) shows that maximizing $Q^T (\theta^t, \theta^{t-1})$ with respect to $\theta^c$ and $\theta^e$ can be performed separately.

Maximization of $Q^T_c (\theta^c^*, \theta^{t-1})$:

To estimate the prior probabilities, we assume that we have an explicit probability distribution using soft-max probability defined as follows with parameter $\lambda$:

$$\theta^c^* = \frac{e^{\lambda^c^*}}{\sum_{r \in C} e^{\lambda^r}}$$

The derivative of $Q^T_c (\theta^c^*, \theta^{t-1})$ with respect to $\lambda^c^*$ can be set to zero ($\frac{\partial Q^T_c (\theta^c^*, \theta^{t-1})}{\partial \lambda^c^*} = 0$); in this case the derivative define the following expression:

$$\frac{\partial Q^T_c (\theta^c^*, \theta^{t-1})}{\partial \lambda^c^*} = \sum_{z_i \in A} \sum_{c^* \in C} p(c^*/z_i, \theta^{t-1}) - N \theta^c^* = 0$$

Thus, this leads to updating $\theta^c^*$ calculated with the parameters estimated at $t^{th}$ step as defined in equation (7).

Maximization of $Q^T_1 (\theta^c, \theta^{t-1})$:

Using the Bernoulli distribution (3), the objective function $Q^T_1 (\theta^c, \theta^{t-1})$ becomes:

$$Q^T_1 (\theta^c, \theta^{t-1}) =$$

$$\sum_{k=1}^{n} \sum_{c \in C} \sum_{z_i \in A} \left[ -p(c/z_i, \theta^{t-1}) \ln \left( \frac{1 - \varepsilon^k_c}{\varepsilon^k_c} \right) |z^k_i - w^k_c| \right]$$

$$+ \sum_{k=1}^{n} \sum_{c \in C} \sum_{z_i \in A} \left[ p(c/z_i, \theta^{t-1}) \ln(1 - \varepsilon^k_c) \right]$$

Maximizing $Q^T_1 (\theta^c, \theta^{t-1})$ with respect $w^k_c$ and $\varepsilon^k_c$ is performed separately. First, with the condition that $\varepsilon^k_c \in [0, \frac{1}{2}]$, the first term of $Q^T_1 (\theta^c, \theta^{t-1})$ is a negative term, and therefore maximizing $Q^T_1 (\theta^c, \theta^{t-1})$ requires minimizing the inertia $I^k_c$ associated with each binary variable $z^k_i$ defined as follows:

$$I^k_c (w^k_c) = \sum_{z_i \in A} p(c/z_i, \theta^{t-1}) |z^k_i - w^k_c|$$

Under the condition that $w^k_c \in \beta = \{0, 1\}$, it is clear that this minimization allows definition of the median center $w_c \in \beta^n$ of the observations $z_i \in A$ weighted by $\gamma_i = p(c/z_i, \theta^{t-1})$ (See section 2). Each component of $w_c = (w^1_c, ..., w^k_c, ..., w^n_c)$ is then computed according to expression (8).
REFERENCES

In the second step, we calculate the derivative of $Q^T_1(\theta^c, \theta^{t-1})$ with respect to $\varepsilon^k_c$ and setting it to zero ($\frac{\partial Q^T_1(\theta^c, \theta^{t-1})}{\partial \varepsilon^k_c} = 0$).

$$\frac{\partial Q^T_1(\theta^c, \theta^{t-1})}{\partial \varepsilon^k_c} = \sum_{z_i \in A} \left[ \frac{p(c/z_i, \theta^{t-1})|z_i^k - w^k_c|}{\varepsilon^k_c(1 - \varepsilon^k_c)} - \frac{\varepsilon^k_c p(c/z_i, \theta^{t-1})}{\varepsilon^k_c(1 - \varepsilon^k_c)} \right] = 0$$

This leads to the update that is calculated with the parameters estimated at $(t-1)^{th}$ step as defined in equation (9).

References

REFERENCES

REFERENCES


