ESANN'1997 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 16-17-18 April 1997, D-Facto public., ISBN 2-9600049-7-3, pp. 13-18

Probabilistic Self Organized Map Application to classification

F. Anouar[1], F. Badran[1] et S.Thiria[1,2]

1 CEDRIC, Conservatoire National des Arts et Métiers, 292 rue Saint Martin, 75 003 PARIS
2 Laboratoire d'Océanographie et de Climatologie (LODYC), Université de PARIS 6. 4 Place Jussieu, 75005 PARIS
E-mail : anouar@cnam.fr

Abstract
We propose in this paper a new learning algorithm using a probabilistic formalism for topological maps. This algorithm approximates the density distribution of the input set with a mixture of normal distributions. The unsupervised learning is based on the dynamic clustering principle and optimize the likelihood function. A supervised version of this algorithm is proposed. We perform numerical experiments on simulated and real data with the aim of achieving a classification task. Our results are compared with the classical SOM (Self organizing Map) algorithm.

1. Introduction

This paper deals with a probabilistic formalism for the Self Organizing Map (SOM) [Kohonen 94] and proposes a new learning algorithm PRSOM (PRobabilistic Self Organizing Map) which maximizes the likelihood function. Because of this probabilistic formalism, each neuron represents a gaussian function, and the learning algorithm estimates both mean and variance of each gaussian. Under some particular hypothesis, this algorithm is closely related to the classical SOM.
In the remainder of this paper, we first introduce the algorithm PRSOM that we prove the convergence to a local minimum. The next section is devoted to simulation results, it shows how to use PRSOM to solve classification tasks.

2. Probabilistic Self Organizing Map algorithm (PRSOM)

Let us first introduce the notations we used. Let D be the data space ($D \subset \Re^n$) and $A=\{z_i ; i=1,...,N\}$ the training set ($A \subset D$). We denoted by (C) a map of M neurons, this map is assumed to have a neighborhood system. The distance $\delta(c,r)$ between two neurons (c) and (r) of the map is the length of the shortest path between c and r on the map. The neighborhood size is controlled by a function $K_T(\delta(c,r)) = [1/T]K(\delta(c,r)/T)$ where $K(.)$ is a kernel function. $K_T(.)$ varies with respect to the size T of the neighborhood. In the following, the parameter T will be

ESANN'1997 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 16-17-18 April 1997, D-Facto public., ISBN 2-9600049-7-3, pp. 13-18

called temperature. We associate to each neuron c a gaussian density function f_c with mean the weight vector or reference vector $W_c = (W_c^1, W_c^2, \cdots, W_c^n)$ and covariance matrix $\sigma_c^2 I$.

2.1. Probabilistic formalism

In the probabilistic formalism proposed by Luttrel [Luttrel 94], the whole network will be designed as a three layers architecture: the input layer has n neurons receiving the input vector z. The classical map C is duplicated in two similar maps C_1 and C_2 provided with the same topology as C. C_1 and C_2 will be respectively the second and the third layer .

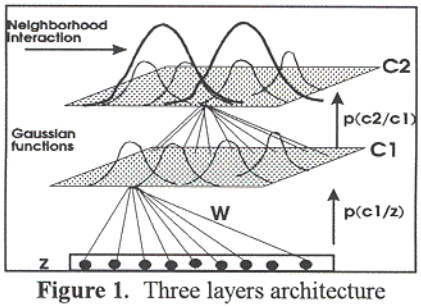


Figure 1. Three layers architecture

Each neuron of each layer computes its probable state. This structure is called a folded Markov chain [Luttrel 94] in which we assume the Markov property given by

$$p(c_2 / z, c_1) = p(c_2 / c_1) \text{ and } p(z / c_1, c_2) = p(z / c_1)$$

We obtain the joint probability :

$$p(z) = \sum_{c_2} p(c_2) p_{c_2}(z) \text{ where } p_{c_2}(z) = p(z / c_2) = \sum_{c_1} p(c_1 / c_2) p(z / c_1)$$

The probability density $p(z)$ is completely defined from the map given the conditional probability $p(c_1 / c_2)$ on the map and the conditional probability $p(z / c_1)$ on data. $p(z / c_1)$ may be any given density, in the following we will deal with gaussian densities. We assume that

$$p(c_1 / c_2) = [1/T_{c_2}] K_T(\delta(c_1, c_2)) \text{ where } T_{c_2} = \sum_r K_T(\delta(c_2, r)) ,$$

$p(z / c_1) = f_{c_1}(z, W_{c_1}, \Sigma_{c_1})$ where f_{c_1} is the gaussian density with mean vector W_{c_1} and covariance matrix $\Sigma_{c_1} = \sigma_{c_1}^2 I$. Under these assumptions

$$p_{c_2}(z) = [1/T_{c_2}] \sum_{r \in C_1} K(\delta(c_2, r)) f_r(z, W_r, \sigma_r) .$$

and the global density $p(z)$ is a mixture of the density $p_{c_2}(z)$ whose parameters have to be estimated. We propose in the following a learning procedure which maximizes the likelihood function, and estimates the mixture parameters.

ESANN'1997 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 16-17-18 April 1997, D-Facto public., ISBN 2-9600049-7-3, pp. 13-18

2.2. Parameters estimation

To deal with the maximization of the likelihood function, we use a dynamic clusters method [Diday 76] [Schroeder 76] and we develop an iterative batch algorithm operating in tow steps : an assignment step which assigns each observation z to one cell c_2 using an assignment function χ and an optimization step which minimizes the conditional likelihood function. The purpose of these steps is to optimize the likelihood. Each step maximizes partially the likelihood. We assume that the observations are independent, the likelihood of the training set depends on the gaussian parameters W and σ and on the assignment function χ:

$$p(z_1,z_2,\cdots,z_N,W,\sigma,\chi) = \prod_{i=1}^{N} p_{\chi(z_i)}(z_i)$$

We minimize the log-likelihood function $E(\chi,W,\sigma)$ according to the assignment functions χ and to the parameters W and σ :

$$E(\chi,W,\sigma) = \sum_{i=1}^{N} -Ln[\sum_{r\in C} K(\delta(\chi(z_i),r)) f_r(z_i,W_r,\sigma_r)]$$

χ ,W and σ will be noted χ^k , W^k and σ^k at the k^{th} iteration.

Assignment step :

This step needs to choose a assignment function χ to improve the log-likelihood E. The dynamic clustering method assumes that each observation z_i is generated from one of the M density functions $p_{c_2}(z)$ by the most likely cell c_2 which is defined by:

$$\chi(z) = \arg\max_{c_2} p_{c_2}(z) \qquad (1)$$

χ depends on the current parameters. The assignment step uses the current function χ to partition the input space,

Minimization step :

The optimization step updates the parameters by setting the derivatives of the conditional log-likelihood $E(W^k,\sigma^k / \chi^k)$ to zero. If we suppose that the initial estimates of the parameters are sufficiently close to the true values, it yields the following new parameters :

$$W_r^k = \frac{\sum_{i=1}^{N} z_i K(\delta(r,\chi^{k-1}(z_i))) \frac{f_r(z_i,W_r^{k-1},\sigma_r^{t-1})}{p_{\chi^{k-1}(z_i)}(z_i)}}{\sum_{i=1}^{N} K(\delta(r,\chi^{k-1}(z_i))) \frac{f_r(z_i,W_r^{k-1},\sigma_r^{t-1})}{p_{\chi^{k-1}(z_i)}(z_i)}} \qquad (\sigma_r^k)^2 = \frac{\sum_{i=1}^{N} \left\| W_r^{k-1} - z_i \right\|^2 K(\delta(r,\chi^{k-1}(z_i))) \frac{f_r(z_i,W_r^{k-1},\sigma_r^{t-1})}{p_{\chi^{k-1}(z_i)}(z_i)}}{n\sum_{i=1}^{N} K(\delta(r,\chi^{k-1}(z_i))) \frac{f_r(z_i,W_r^{k-1},\sigma_r^{t-1})}{p_{\chi^{k-1}(z_i)}(z_i)}} \qquad (2)$$

PRSOM algorithm

Initialization : Choose randomly the initial parameters W^0 et σ^0 or χ^0.
Iterative step : at the iteration step k ,W^{k-1} et σ^{k-1} are known,
Minimization step : Compute the new parameters W^k and σ^k according to (2).
Assignment step : Compute the new assignment function χ^k associated to W^k et σ^k according to (1).
Repeat the iteration step until stabilization.

We prove that the PRSOM algorithm converges in a finite number of iterations to a local minimum when the parameter temperature T is fixed.

2.3. Convergence proof

The minimization step at the k^{th} iteration minimize $E(\chi^{k-1}, W, \sigma)$ according to the parameters W and σ for a fixed assignment function χ^{k-1}, we obtain the following inequality :

$$E(\chi^{k-1}, W^k, \sigma^k) \leq E(\chi^{k-1}, W^{k-1}, \sigma^{k-1}) .$$

The assignment step at the k^{th} iteration minimize $E(\chi, W^k, \sigma^k)$ with respect to χ for a given parameters W^k and σ^k and gives :

$$E(\chi^k, W^k, \sigma^k) \leq E(\chi^{k-1}, W^k, \sigma^k) .$$

Combining the two inequalities we obtain :

$$E(\chi^k, W^k, \sigma^k) \leq E(\chi^{k-1}, W^{k-1}, \sigma^{k-1}) .$$

The set of partitions is finite and each partition obtained by χ^k has a unique associated parameters W^k and σ^k, thus the number of expected values of $E(\chi^k, W^k, \sigma^k)$ is finite. Since the log-likelihood function $E(\chi, W, \sigma)$ decreases at each iteration, the algorithm converges in a limited number of iterations. The limit point is a local minimum of $E(\chi, W, \sigma)$.

3. Supervised PRS0M

The architecture of a supervised PRSOM is composed of three layers. The first and the second layers represent the PRSOM architecture. The third layer is composed of linear neurons which computes the weighted sum $y(z) = \sum_{c=1}^{k} \lambda_c f_c(z) + \lambda_0$ where the basis function f_c are gaussian functions, λ_0 is a threshold and λ_c are parameters to be estimated. f_c can be presented as radial basis functions.

Learning in the supervised PRSOM uses the PRSOM algorithm to organize the map and to learn the first layer of parameters (W and σ). The second layer of the parameters λ is adapted by a gradient method algorithm which minimize the quadratic error : $E = \sum_{i=1}^{n} \|y(z_i) - t_i\|^2$ where $y(z_i)$ is the output, z_i the input and t_i the desired output.

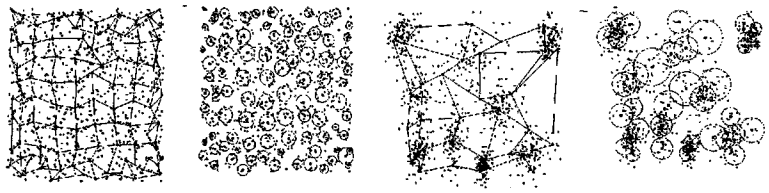
4. Applications

4.1 Simulated data

In the following, we first show that, like the classical SOM algorithm, the PRSOM algorithm preserves the topological order of the map. This topological order is associated with the gaussian densities formalism assigning to each neuron its estimated mean and standard deviation.

We simulated two different data sets: the first one has 800 examples generated according to the uniform density in D=[-12,12][-12,12]; the second data set has 900

examples and is generated according to a mixture of 9 gaussians. For all the simulations we use a map with square neighborhood. The kernel function is defined by: $K_T(u) = [1/T]e^{[-u^2/T^2]}$ where the parameter T represents the width of the Kernel. For the first data set the map has 10x10 neurons, and for the second it has 6x6 neurons. For each experience, we display the referents and the resulting maps after learning:



(a) (b) (c) (d)

Figure 1. (a) Represents the final map after learning with PRSOM algorithm for the first data base (uniform distribution). (b) Displays standard deviation as a circle centered on the referent. (c) Represents the final map after learning with PRSOM algorithm for the second data base (9 Gaussian distributions). (d) Displays standard deviation as a circle centered on the referent.`

Figures 1(a) and 1(c) show the maps for the two data sets after training. We can observe the topological order of the map. Figures 1(b) and 1(d) show the estimated standard deviation for each referent, each one is represented by the radius of a circle centered on it. One circle represents the "influence region" of the associated referent. One can see the influence of the σ parameters, they fit into the local distribution: they are small in regions with high density and large in regions with weak density.

4.2 Real data : classification

We studied the coronary data [SAP90] which concerns 101 coronary victims. 51 patients will died and 50 will survived The input have seven variables. Unlike the supervised classification, the unsupervised classification do not use the desired information during learning. We will present results in both cases using a binary states 0 and 1 for the tows classes.

4.2.a Unsupervised Classification

The unsupervised PRSOM algorithm can be used in unsupervised classification tasks. In that case each referent can be labeled, using the learning set, by the majority vote procedure. In the SOM algorithm the labeling and the classification procedure use the Euclidean distance. In PRSOM, the classification is achieved using the square of a local Mahalanobis distance $(z - W_i)^2 / \sigma_i^2$ between an observation z and the referent i. In order to compare these two classifiers we studied the coronary data. The learning set contains the whole data. Results are presented in the table 1 and show that PRSOM do better than the SOM algorithm on these data.

	PRSOM	SOM
9 referents	86.14%	84.16%
16 referents	88.11%	86.14%

Table 1: performances of PRSOM and SOM algorithms on coronary data and for tow maps with different size

4.2.b Supervised classification

We compare performances of the supervised classification using a Multi-Layer Perceptron (MLP) with the supervised PRSOM. These results are given in table 2. The best result of the MLP has been obtained with an architecture with tow neurons on one hidden layer

	MLP Architecture:7x2x2	Supervised PRSOM 16 referents
Performance	90.009%	93.069%

Table 2 : performances of the MLP and the supervised PRSOM.

5. Conclusion

We have presented a new formalism of topological maps in which each neuron represents a gaussian function. Under this formalism, we have developed a new probabilistic unsupervised learning algorithm PRSOM based on the likelihood estimation. The aim is to approximate the density function of data by a mixture of densities. Each elementary density of the mixture is related to one neuron of the map and depends on the gaussian functions defined on his neighborhood. Thus unlike the SOM algorithm PRSOM optimizes an objective function defined by the likelihood function.

We have also develop a supervised version of the PRSOM algorithm. We have presented an application of the PRSOM algorithm to classification task. We show that PRSOM does better than the classical SOM. This can be explained by the fact that PRSOM encloses more general statistical considerations and allows to estimate the variances which define an influence zone around each neuron. This variances are used to define a local measure which is more efficient in a classification task than the classical quadratic error.

References

[1] Diday E. , Simon J. C., "Clustering analysis" Digital pattern recognition Edited by K. S. Fu, Springer-Verlag, New York 1976..

[2] Luttrel S.P , 1990. "Derivation of a class of training algorithms". *IEEE Trans. Neural Networks* 1, 229-232.

[3] Luttrel S.P , 1994. "A Bayesian Analysis of Self-Organizing Maps". *Neural Computing* vol 6,

[4] Kohonen T. 1994. Self-Organizing Maps. Springer.

[5] Saporta G.,"Probabilités, analyse des données et statistiques" Technip, Paris 1990

[6] Schroeder A. , 1976. "Analyse d'un mélange de distribution de probabilité de même type". *RSA*. vol 24, n° 1.