Engineering Applications of Artificial Intelligence 59 (2017) 218–235

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Beta Scale Invariant Map

Héctor Quintián*, Emilio Corchado

Department of Computer Science and Automation, University of Salamanca, Plaza de la Merced s/n, Salamanca 37007, Spain

ARTICLE INFO

Keywords:
Topology preserving maps
Quality measures
SIM
MLHL-SIM
ViSOM
GNG
Beta distribution
Imbalanced datasets

ABSTRACT

In this study we present a novel version of the Scale Invariant Map (SIM) called Beta-SIM, developed to facilitate the clustering and visualization of the internal structure of complex datasets effectively and efficiently. It is based on the application of a family of learning rules derived from the Probability Density Function (PDF) of the residual based on the beta distribution, when applied to the Scale Invariant Map. The Beta-SIM behavior is thoroughly analyzed and successfully demonstrated over 2 artificial and 16 real datasets, comparing its results, in terms of three performance quality measures with other well-known topology preserving models such as Self Organizing Maps (SOM), Scale Invariant Map (SIM), Maximum Likelihood Hebbian Learning-SIM (MLHL-SIM), Visualization Induced SOM (ViSOM), and Growing Neural Gas (GNG). Promising results were found for Beta-SIM, particularly when dealing with highly complex datasets.

1. Introduction

Among the great variety of tools for multidimensional data visualization, several of the most widely used are those belonging to the family of the topology preserving maps (Chen et al., 2013; Fuertes et al., 2010; Kohonen, 1998; Mohebi and Bagirov, 2016; Wu et al., 2011). Probably the best known among these algorithms is the Self-Organizing Map (SOM) (Chen et al., 2013; Kohonen, 1998, 2013; Haimoudi et al., 2016). It is based on a type of unsupervised learning called competitive learning; an adaptive process in which the units in a neural network gradually become sensitive to different input categories or sets of samples in a specific domain of the input space. The main feature of the SOM algorithm is its topology preservation. When not only the winning unit, but also its neighbors on the lattice are allowed to learn, neighboring units gradually specialize to represent similar inputs, and the representations become ordered on the map lattice.

Several extensions of SOM can be found in the literature such as the Generative Topographic Mapping (GTM) (Bishop et al., 1998; Ghassany and Bennani, 2015), which was developed by Bishop et al. as a probabilistic version of the SOM, in order to overcome some of its limitations, particularly the lack of an objective function. An important application of the GTM is to allow a simpler visualization of high-dimensional data.

Another extension of SOM is the Topographic Product of Experts (ToPoE), and the Harmonic Topographic Map (HaToM) (Fyfe, 2005; Jeong et al., 2015), where the topology preserving map is created from a product of experts.

The use of ensembles with SOM (Akhand and Murase, 2012; Cho, 2000; Dietterich, 2000; Wang and Gupta, 2015) has also been studied to increase the stability and performance of a specific algorithm. One of the most recent developments of ensembles, in the field of topology preserving maps, is the Weighted Voting Superposition (WeVoS) (Baruque and Corchado, 2014). The principal idea is to obtain the final units of the map by a weighted voting among the units in the same position in different maps, according to a quality measure.

The Visualization Induced SOM (ViSOM) (Corchado and Baruque, 2012; Huang and Yin, 2009), is a SOM extension proposed for the direct preservation of the local distance information on the map, along with the topology. The ViSOM constrains the lateral contraction forces between units and hence regularizes the inter-unit distances, so that distances between units in the data space are in proportion to those in the input space. The ViSOM not only takes into account the distance between a unit's weights from one iteration to the next, but also the distance between that unit and the Best Matching Unit within the whole map (BMU). This allows the ViSOM to preserve topology by maintaining distance between neighbors of the winner unit.

Two other interesting topology preserving models are the Scale Invariant Map (SIM) (Baruque and Corchado, 2014, 2009) and the Maximum Likelihood Scale Invariant Map (MLHL-SIM) (Baruque and Corchado, 2011; Corchado and Fyfe, 2002). Both are designed to perform their best with radial datasets, due to the fact that both create a mapping where each neuron captures a "pie slice" of the data according to the angular distribution of the input data (see Fig. 1). However, when SOM is trained, it approximates a Voronoi tessellation

* Corresponding author.
E-mail addresses: hector.quintian@usal.es (H. Quintián), escorchado@usal.es (E. Corchado).

http://dx.doi.org/10.1016/j.engappai.2017.01.002
Received 7 April 2016; Received in revised form 3 November 2016; Accepted 2 January 2017
Available online 12 January 2017

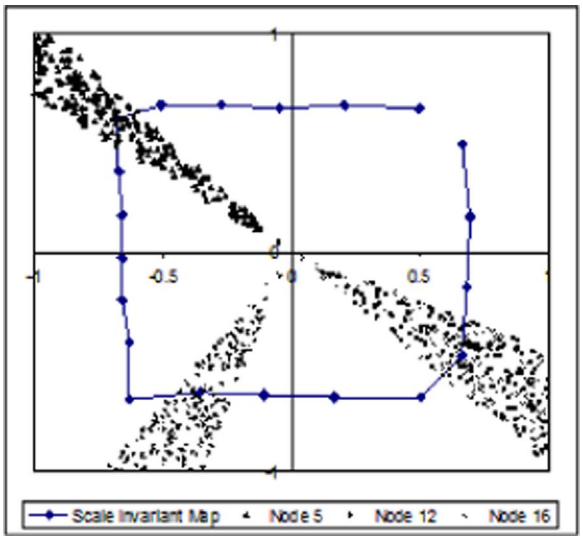

Fig. 1. Scale Invariant Map mapping, where each neuron captures a "pie slice" of the data according to the angular distribution of the input data.

of the input space (Kohonen, 1998). The Scale Invariant Map is an implementation of the negative feedback network (Fyfe, 2005) to form a topology preserving mapping. The main difference between this mapping and the SOM (Kohonen, 1998, 2013) is that this mapping is scale invariant.

Finally, another widely used clustering and classification algorithm is the Growing Neural Gas (GNG) algorithm, proposed by Fritzke (1995), Zapater et al. 2015). It is based on the Neural Gas (NG) algorithm previously proposed by Martinetz et al. (1993) for finding optimal data representations based on feature vectors, which is in turn a modification of the widely known SOM. The main characteristic of the NG algorithm is that instead of expanding through the data input space as a fixed grid of units (as done by the SOM algorithm), the NG algorithm allows the neighboring relationships of its units to change, expanding more like a gas over the data space.

The GNG method is different from the previous algorithms in that it is an incremental algorithm, so there is no need to determine a priori the number of nodes. Network shape and size are determined during the training, while the SOM and NG are often trained on a fixed network size throughout.

The GNG (Zapater et al., 2015) is a combination of Fritzke's Growing Cell Structures (GCS) (Fritzke, 1994) and Martinetz's Competitive Hebbian learning (CHL) (Martinetz, 1993). The network topology of the GNG is generated incrementally by the CHL algorithm, which successively inserts topological connections or edges. The main principle of the CHL is that for each input x, it connects the two closest centers (measured by Euclidean distance) with an edge.

This research study presents a novel and efficient technique for data clustering called Beta-Scale Invariant Map (Beta-SIM). It is based on a modification of a topology preserving map that can be used for scale invariant classification (Baruque and Corchado, 2014; Corchado and Baruque, 2012; Baruque et al., 2011; Corchado and Colin, 2002). The main objectives of this study are:

- To study and derive a family of learning rules from Beta distribution and apply them to the Scale Invariant Map (SIM) (Baruque and Corchado, 2014, 2009) to improve the clustering and visualization of internal structure of high dimensional datasets, specifically with radial structure.
- To thoroughly study the advantages and disadvantages of the novel Beta-SIM algorithm over 2 artificial and 16 real datasets, testing its capabilities.
- To test the capacity of the novel proposed algorithm (Beta-SIM) to adapt to sparse clusters or to neglect outliers through the right combination of α and β values, depending on task to be carried out.

This paper is organized as follows: Section 2 presents in detail the SIM algorithm which leads on to the MLHL and MLHL-SIM algorithms that are explained in Sections 3 and 4. Section 5 introduces the Beta Hebbian Learning used to derive the learning rules for the new algorithm, Beta-SIM, which is described in detail in Section 6. Section 7 presents 3 quality measures, previously proposed in the literature, used to evaluate different properties of topology-preserving mapping algorithms in general. Section 8 analyzes the capabilities of the Beta-SIM algorithm by applying it to perform a detailed study over 2 artificial datasets and 16 real benchmark datasets with diverse characteristics. Finally, Section 9 contains the final conclusions and outlines future lines of research.

2. Scale Invariant Map

The main target of the family of topology preserving maps (Kohonen, 1998) is to produce low dimensional representations of high dimensional datasets, maintaining the topological features of the input space.

SIM (Baruque and Corchado, 2014, 2009) is an algorithm similar to SOM (Kohonen, 1998), but the training methodology is based on negative feedback networks (Fyfe, 2005, 1997). SIM uses a neighborhood function and competitive learning in the same way as the SOM. The SIM model is defined by Eqs. (1)–(3):

$$Feedforward: y_i = \sum_{j=1}^{N} W_{ij}x_j, \tag{1}$$

$$Feedback: e = x - W_c y_c \quad (y_c=1), \tag{2}$$

$$Weights\ update: \Delta W_i = h_{ci}\eta(x - W_c), \quad \forall i \in N_c, \tag{3}$$

where x is an N-dimensional input vector, and y an M-dimensional output vector, with W_{ij} being the weight linking input j to output i; e is the residual or error, η the learning rate, W_c refers to the weights of the winning neuron and h_{ci} represents the neighborhood function, which is a Gaussian function in this case.

The input data x_j is feedforward through weights W_{ij} to create output data y_i, where a linear summation is performed to obtain the activation of the output neurons (1). Based on the activation from the feedforward algorithm, a winner neuron is selected using the minimum Euclidean distance (the neuron whose output vector is closest to the input vector wins) or using the maximum activation (the output neuron with the highest activation wins). After selection of an output winner, denoted as c, it is deemed to be firing (y_c=1) and all other outputs are suppressed (y_i=0, $\forall i \neq c$).

The winner's activation is then used as feedback (2) using the winner's weights subtracted from the input data, and simple Hebbian learning to update the weights of all nodes in the neighborhood of the winner (3).

3. An exponential family of learning rules

Maximum Likelihood Hebbian Learning (MLHL) (Corchado et al., 2004) is a family of rules created from exponential distributions, which can be derived to express the Probability Density Function (PDF) of the residual after feedback as (4):

$$p(e) = \frac{1}{Z}exp(-|e|^p), \tag{4}$$

It can then be denoted as a general cost function associated with this network as (5):

$$J = E(-log(p(e))) = E(|e|^p + K), \tag{5}$$

where K is a constant independent of W and the expectation is taken over the input dataset. Therefore, performing gradient descent on J we have (6):

$$\Delta W \propto -\left.\frac{\partial J}{\partial W}\right|_{W(t-1)} = -\left.\frac{\partial J}{\partial e}\frac{\partial e}{\partial W}\right|_{W(t-1)} \approx E\{y(p\,|e|^{p-1}sign(e))^{T}|_{W(t-)}\}, \tag{6}$$

where T denotes the transpose of a vector and the power of the norm of e is taken on an element-wise basis as it is derived from a scalar form of a vector.

Computing the mean of a function of a dataset (or even the sample averages) can be tedious, and it is important to cater for the situation in which new samples are continuously added to the dataset. If the conditions of stochastic approximation (Mendel, 1994) are satisfied, the mean can be approximated with a difference equation. The function to be approximated is clearly sufficiently smooth and the learning rate can be designed to approximately satisfy $\eta_k \geq 0$, $\sum_k \eta_k = \infty$, $\sum_k \eta_k^2 < \infty$ and so we have the rule (7):

$$\Delta W_{ij} = \eta \bullet y_i \bullet sign(e_j) \bullet |e_j|^{p-1}, \tag{7}$$

It is expected that for leptokurtic residuals (more kurtotic than a Gaussian distribution), values of $p < 2$ would be appropriate, while for platykurtic residuals (less kurtotic than a Gaussian), values of $p > 2$ would be appropriate.

4. Maximum Likelihood Scale Invariant Map (MLHL-SIM)

The Maximum Likelihood Scale Invariant Map (MLHL-SIM) Corchado and Fyfe, 2002 is an extension of the SIM based on the application of the Maximum Likelihood Hebbian Learning (MLHL) (Corchado et al., 2004).

The main difference with regards to the SIM is how the MLHL is used to update the weights of all nodes in the neighborhood of the winner, once the winner has been updated. This can be expressed as (8):

$$\Delta w_i = h_{ci} \bullet \eta \bullet sign(e - W_c) \bullet |e - W_c|^{p-1}, \quad \forall i \in N_c, \tag{8}$$

By giving different values to p, the learning rule is optimal for different probability density functions of the residuals. h_{ci} is the neighborhood function as in the case of the SOM and N_c is the number of output neurons. Finally, η represents the learning rate.

During the training of the SIM or the MLHL-SIM, the weights of the winning node are fed back as inhibition to the input vector, and then in the case of the MLHL-SIM, MLH learning is used to update the weights of all nodes in the neighborhood of the winner as explained above.

5. Beta Hebbian learning

5.1. Beta distribution

Beta distribution is a family of continuous probability distributions defined in the interval [0,1] with two positive shape parameters, denoted by α and β. Beta distribution is defined by (9):

$$f(x;\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)} \tag{9}$$

where $f(x;\alpha,\beta)$ is the PDF, x is the input value to the distribution, α and β are the parameters that determine the shape of the PDF curve, and $B(\alpha,\beta)$ is the beta function, which is a normalization constant to ensure that the total probability integrates to 1. The Beta function is calculated using the gamma function (10) and defined by (11).

$$\Gamma(n) = (n-1)!, \tag{10}$$

$$B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)} \tag{11}$$

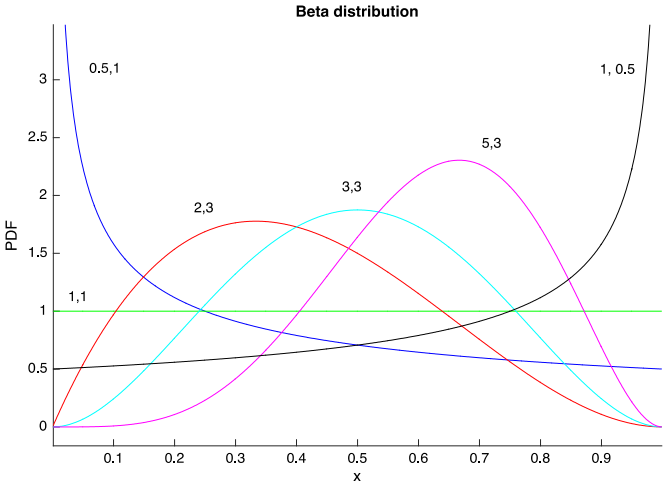

Fig. 2. Probability Density Function (PDF) of the Beta distribution for different values of α and β.

Beta distribution is very malleable based on the parameters α and β (see Fig. 2). The relation between parameters α and β determines the shape of the PDF, with the capability of generating distributions with positive ($\alpha > \beta$) and negative ($\alpha < \beta$) skewness, platykurtic ($\alpha=\beta$ smalls values < 3), *meso*kurtic ($\alpha=\beta\approx3$) and leptokurtic ($\alpha=\beta$ large values > 3) distributions, and combinations of these.

5.2. A new family of learning rules: Beta Hebbian Learning

In this research we thoroughly investigate a family of learning rules derived from the PDF of the residual based on the Beta distribution, called Beta Hebbian Learning (BHL), and how they can be applied to the SIM.

If the residual, e, is taken from the Beta distribution, $B(\alpha,\beta)$, with the following probability density function (12):

$$e^{\alpha-1}(1-e)^{\beta-1} = (x - Wy)^{\alpha-1}(1-x - Wy)^{\beta-1}, \tag{12}$$

then, to maximize the likelihood of the data with respect to the weights (W), the gradient descent is performed by means of Eq. (13):

$$\frac{\partial p}{\partial W}(e_j^{\alpha-2}(1-e_j)^{\beta-2}(-(\alpha-1)(1-e_j) + e_j(\beta-1))) = (e_j^{\alpha-2}(1-e_j)^{\beta-2}(1-\alpha + e_j(\alpha+\beta-2))), \tag{13}$$

For instance, in the case in which $\alpha=\beta=2$, we get (14):

$$\frac{\partial p}{\partial W} = y(-(1-e) + e) = y(2e-1), \tag{14}$$

Therefore, the BHL architecture is defined as follows:

$$Feedforward: y_i = \sum_{j=1}^{N} W_{ij}x_j \quad , \forall_i, \tag{15}$$

$$Feedback: e_j = x_j - \sum_{i=1}^{M} W_{ij}y_i, \tag{16}$$

$$Weight\ change: \Delta W_{ij} = \eta(e_j^{\alpha-2}(1-e_j)^{\beta-2}(1-\alpha + e_j(\alpha+\beta-2)))y_i \tag{17}$$

where x is an N-dimensional input vector, and y an M-dimensional output vector, with W_{ij} being the weight linking input j to output i; e is the residual or error, and η the learning rate.

6. Beta-SIM

6.1. Beta SIM learning rule

In this paper we present and analyze, for the first time, a novel

version of the SIM called Beta-Scale Invariant Map (Beta-SIM), based on the application of a family of learning rules derived from the PDF of the residuals of a Beta distribution when they are applied to the SIM.

The main difference with the SIM is that Beta Hebbian Learning is used to update the weights of all nodes in the neighborhood of the winner, once the winner has been updated. The Beta-SIM model is defined by (18)–(20):

$$Feedforward: y_i = \sum_{j=1}^{N} W_{ij}x_j \quad , \forall_i \,, \tag{18}$$

$$Feedback: e_j = x_j - W_c y_c (y_c = 1), \tag{19}$$

Then, if we apply the BHL method to the SIM to update the weights, we get the following rule for the Beta-SIM (20):

$$\begin{aligned} Weight\ change: \Delta W_i = \eta \bullet h_{ci} \bullet sign(x - W_c) \\ \bullet \|x - W_c|^{\alpha-2}(1-|x - W_c|)^{\beta-2} \\ (1-\alpha + |x - W_c|(\alpha + \beta - 2))| \end{aligned} \tag{20}$$

Then, by maximizing the likelihood of the residual with respect to the actual distribution, we are matching the learning rule to the pdf of the residual(e).

The stability of the learning rule was also analyzed in this study, and based on such analysis it can be concluded that the Beta-SIM algorithm is only stable when the absolute value of the residuals is lower than 1 (see Fig. 3). When values of the residuals are beyond this limit, the value of the weights update tends towards infinity. To avoid the possibility that the residuals have values higher than 1, the datasets should be normalized in order to satisfy this limitation and preserve the internal topology between dataset dimensions.

6.2. Influence of the choice of α and β parameters on the Beta-SIM learning rule

In the following, we thoroughly study how the choice of α and β parameters influences the weights update (20) for the different cases: Case 1**: $\alpha=\beta$,** Case 2**: $\alpha > \beta$** and Case 3**: $\alpha < \beta$.**

6.2.1. Case 1: α=β

When *α=β,* the PDF of the beta distribution behavior tends to correspond to a family of exponential distributions (see Fig. 4). Therefore it is expected (Corchado et al., 2004) that for leptokurtic residuals, the choice of high values of α and β (*i.e. α=β=10*) would be more appropriate, while for platykurtic residuals, low values of *α and β (i.e. α=β=2)* would be more appropriate.

Fig. 5 presents the weights updates versus the residual providing

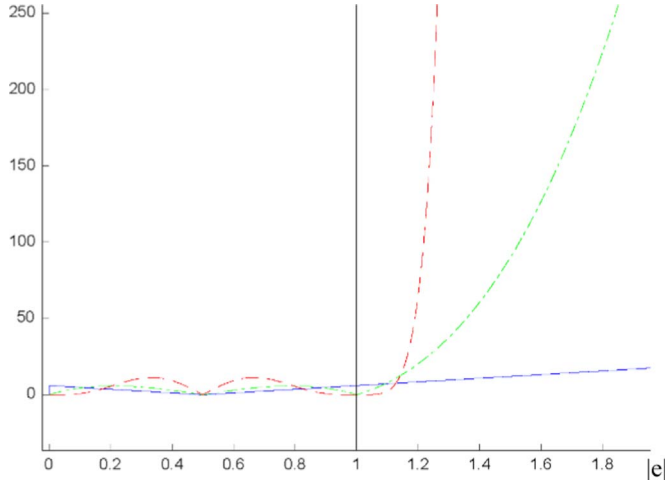

Fig. 3. Stability of the Beta-SIM Learning rule (it becomes unstable for values of |e| > 1).

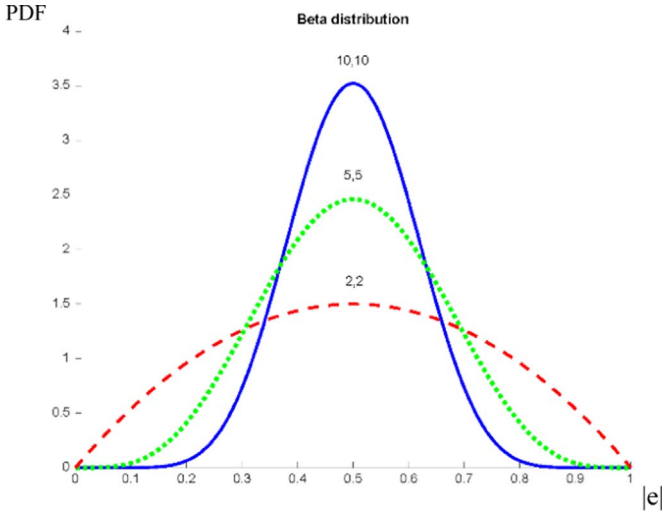

Fig. 4. Beta-Distribution of residuals for values of *α=β:2,2; 5,5; 10,10.*

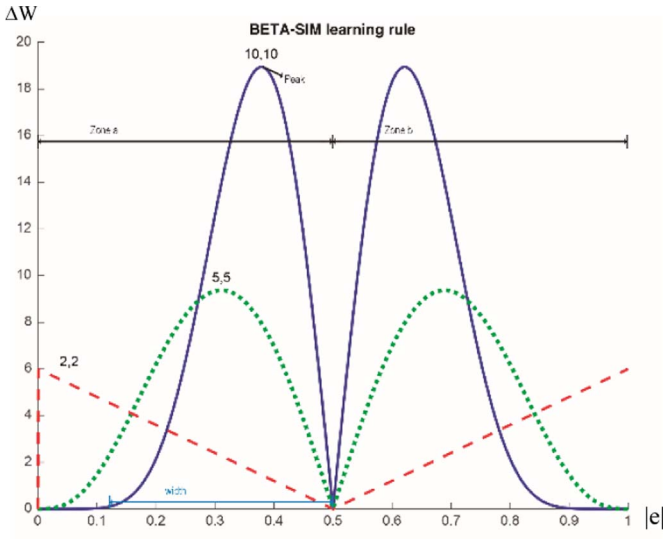

Fig. 5. Beta-SIM learning rule of the residuals for values of *α=β.* (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

relevant information about the behavior of the learning rule based on the choice of α and β. We have analyzed the 3 different possible scenarios related to Case 1: *α=β.*

***Case 1.a): α=β*=high values (i.e. α=β=10)**
***Case 1.b): α=β*=medium values (i.e. α=β=5)**
***Case 1.c): α=β*=low values (i.e. α=β=2)**

In the three cases, it can be seen how the behavior of the weights update (ΔW) versus the value of the residual ($e=x-W_c$) is associated to two zones: **zone *a*** where $0 < e < 0.5$ and **zone *b*** where $0.5 < e < 1$. Such behavior in both areas is symmetric (see Fig. 5).

In Case 1.a when *α=β*= high values (*i.e. α=β=10*), in ***zone a***, as the value of the residual increases, the ΔW value (solid blue line in Fig. 5) increases until the peak of the function is reached and then it reduces to zero very fast.

This is in line with what it is expected in theory; if a winning neuron is near to the input data (i.e. the error is low) the ΔW is also low. As the error increases, the ΔW increases up to the peak of ***zone a***. Finally, when the error approaches 0.5, the ΔW tends to zero and the neuron is not attracted at all to the input.

As highlighted before, the behavior of ΔW versus e in ***zone b*** is

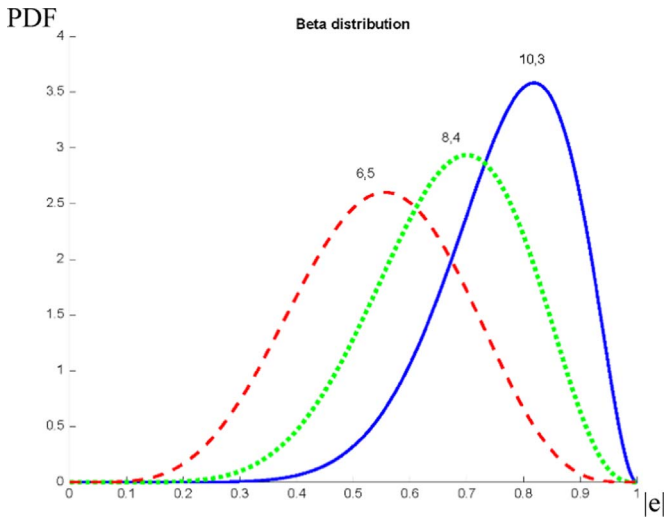

Fig. 6. Beta-Distribution for values of $\alpha > \beta$ *(6,5; 8,4; 10,3).*

symmetric to ***zone a*** (see Fig. 5).

In both zones the highest values for ΔW are related to the peaks of the function, which are based on the choice of α and β.

In Case 1.b) when α=β=medium values (*i.e.* $\alpha=\beta=5$), (Fig. 5, green dotted line) the behavior resembles the "bell curve" shape. The width of the function is greater and the height is lower than the case of high values of $\alpha=\beta$ *(case 1.a-* solid blue line*).*

In Case 1.c) α=β=low values (*i.e.* $\alpha=\beta=2$), (Fig. 5, red dashed line) it can be deduced that in ***zone a***, low values of e create larger changes in the weights update. As the residual increases, the value ΔW decreases (see Fig. 5, red dashed line) along ***zone a. Zone b*** has, as in the previous two cases, a symmetric behavior than ***zone a***.

6.2.2. Case 2: $\alpha > \beta$

When $\alpha \neq \beta$, an asymmetric distribution is obtained, with positive skewness if $\alpha > \beta$, and negative skewness if $\alpha < \beta$. The PDF of the Beta distribution when $\alpha > \beta$ is shown in Fig. 6 for several values of both parameters. For values of $\alpha > > > \beta$, leptokurtic and positive skewness residuals are larger (Corchado et al., 2004). Fig. 7 shows the representation of the weights update versus the residual (learning rule (20)) for different values of $\alpha > \beta$.

Again two zones associated to the magnitude of the residual can be identified. ***Zone a***, where the W increases as the residual increases until reaching the first peak of the function and then decreases quickly to zero (see Fig. 7). From this point, the behavior of the learning in ***zone b*** (high values of e) is very interesting as it may be utilized for the identification of sparse clusters or outliers, and based on Fig. 7, their influence can be taken into account, or be discarded, depending on the values of α and β. Such effect may be useful depending on the nature of the dataset to be analyzed and on the purpose of the study.

During the training process, the samples from the dataset are selected randomly, so the probability to select a sample of a sparse cluster is lower than the probability to select a sample from a non-sparse cluster (with a high number of samples). This means that normally, the network is trained more often over samples of non-sparse clusters, so the network tends to adapt to these non-sparse clusters. At this point, if we use Beta-SIM with parameters ***α > β (Case 2)***, when a sample of a sparse cluster is selected for training, the distance of this sample to the winning neuron (residual e=) normally will be large. This case is related to *zone b* of the learning rule (see Fig. 7). Therefore, the winner node, and its neighbors, are updated attracting the network grid strongly to the sparse cluster. Then, at the end of the training process, at least some nodes of the network grid will be close to the sparse clusters.

As a conclusion, the Beta-SIM network (specifically when ***α > β (Case 2)***) can be seen as a new tool in the data mining community, in the sense that sparse clusters, that are part of high dimensional datasets, can be taken into account and emphasized during clustering tasks instead of being neglected as in many other topology preserving maps.

6.2.3. Case 3: $\alpha < \beta$

If $\alpha < \beta$ the effect on the beta distribution (Fig. 8) and Beta-SIM learning rule (Fig. 9) is the opposite to Case 2: $\alpha > \beta$. This means that the higher values of the error **(*zone b*)** have now low impact on the weights update, and if a winning output vector is far from the input vector (high $e=x-W_c$), this neuron will be less "attracted" to the input than if it was nearer to the input. This is also a useful research finding as it can be seen as a tool to force the learning process to take into account data associated to low residuals, meaning outliers and sparse datasets would have less influence on the weights.

Therefore, the Beta-SIM model can be seen as a novel tool for the data mining community, as it can help to model sparse data in highly complex datasets, or limit the influence of outliers and noise by selecting appropriate α and β parameters to create the optimal learning rule.

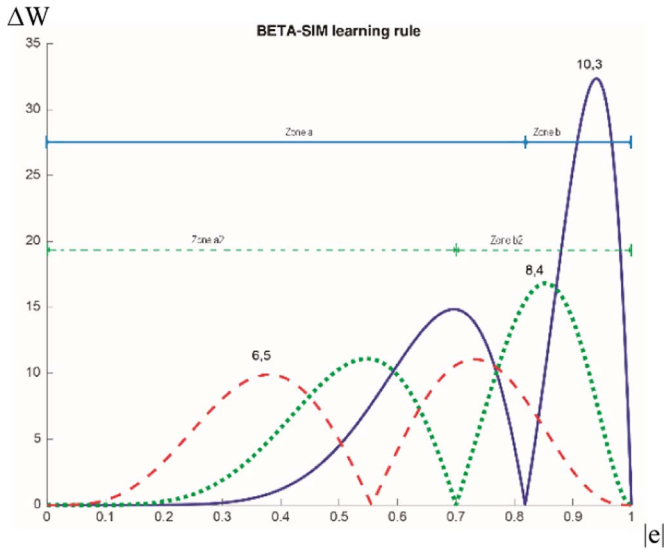

Fig. 7. Beta-SIM learning rule for values of $\alpha > \beta$.

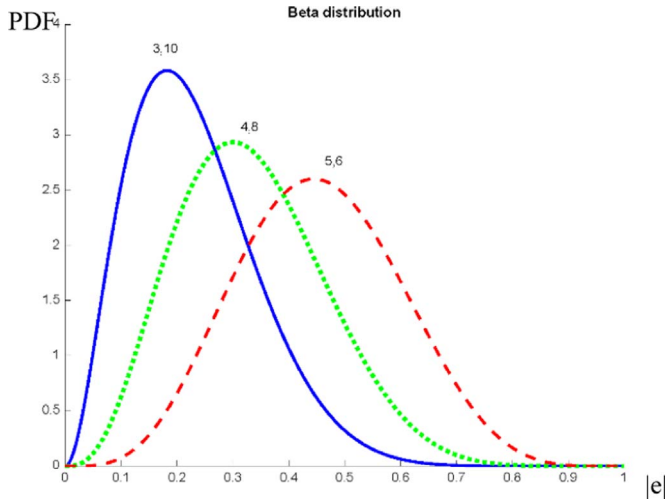

Fig. 8. Beta-Distribution for values of $\alpha < \beta$ *(5,6; 4,8; 3,10).*

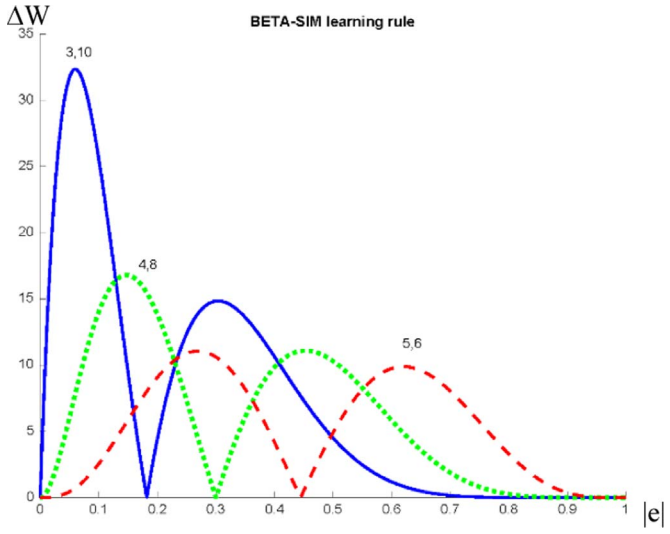

Fig. 9. Beta-SIM learning rule for values of $\alpha < \beta$.

7. Quality measures for topology preserving maps

Several quality measures have been proposed in literature to study the reliability of the results displayed by topology preserving models to represent the given dataset (Baruque and Corchado, 2014; Pölzlbauer, 2004). There are no global or unified measures, but rather a set of complementary measures that each assess a specific characteristic of the performance of the model in different visual representation areas. The three measures used in this study are briefly presented in the following paragraphs.

7.1. Classification error (CE)

Using its inherent pattern matching characteristics, the topology preserving maps can generally be used for classification tasks. Intuitively, the instances that activate the same neuron of the network are very likely to belong to the same category. When a new sample is presented to the network, the sample can be classified in the same class as the majority of samples activating the same neuron. A consistent behavior when classifying samples indicates a correctly trained map. So, although this is not the main function of this kind of network, the measure of how many samples are wrongly classified has been used, to an extent, to assess the quality of the final map in numerous previous studies (Baruque and Corchado, 2014; Pölzlbauer, 2004).

7.2. Topographic error (TE)

TE is the simplest of the topology preservation measures. For all data samples, the respective best and second-best matching units (BMUs) -1st BMU and 2nd BMU- are determined. If these are not adjacent on the map lattice, this is considered an error. The total error is then normalized to a range from 0 to 1, where 0 means perfect topology preservation.

7.3. Mean Quantization Error (MQE)

MQE is related to all forms of vector quantization and clustering algorithms. Thus, this measure completely disregards map topology and alignment. MQE is computed by determining the average distance of the dataset entries to the cluster centroids by which they are represented. In the case of the SOM, the cluster centroids are the characteristics vectors.

8. Experimental datasets

In order to test the novel method presented in this research, 2 artificial and 16 real datasets were used.

The 2 artificial datasets were used to compare the behavior of the Beta-SIM algorithm with the theoretical analysis described in the previous Section *"6.2 Influence of the choice of α and β parameters on the Beta-SIM learning Rule"* and the 16 real datasets, composed of clusters of different sparsity, were used to test the Beta-SIM algorithm on clustering tasks.

Following, 2 real datasets were used to analyze the behavior of the Beta-SIM algorithm to contrast the conclusions obtained in the experiments over the artificial dataset. Once this analysis was performed, the Beta-SIM algorithm was tested over 14 real benchmark datasets by means of a statistical test using three quality measures to compare the novel algorithm results against 5 other well-known topology preserving algorithms (SOM, SIM, MLHL-SIM, ViSOM and GNG).

The objective of the experiments was to show that the novel Beta-SIM network, based on the appropriate choice of α and β parameters, outperforms other topology preserving models, when they are applied to different datasets composed of clusters with different levels of sparsity (imbalanced datasets).

In all experiments, parameters were chosen in an experimental process of trial and error. As parameter selection is a task that is very dependent on the dataset to be used, several initial experiments were conducted with a range of combinations of these parameters.

8.1. Artificial datasets

Two artificial datasets were created to measure the adaptation of the network grid to datasets with different sparsity zones.

The objective of these experiments is to analyze the behavior of the Beta-SIM algorithm to contrast the theoretical analysis developed in the previous Section *"6.2 Influence of the choice of α and β parameters on the Beta-SIM learning Rule"*.

8.1.1. Artificial dataset 1

In this experiment, a 2-D dataset with radial layout was generated. The dataset consists of a uniform distribution in the shape of an ellipse, where some of the samples were removed to get areas with different levels of sparsity over the X axis (see Fig. 10).

To generate this sparsity over the X axis, a small ellipse was created inside the dataset with an offset in respect to the center of the dataset, with all samples inside this ellipse removed from the main dataset (see Fig. 10-red dots).

The objective of this experiment is to measure the adaptation of the different topology preserving networks (SOM, SIM, MLHL-SIM, Beta-SIM) to the dataset, as it presents areas with different levels of sparsity.

As previously mentioned (Section 7), the following quality measures are applied to measure the adaptation of the network grids to the dataset: MQE and TE. In these experiments the network grid consisted of 20 neurons.

Table 1 shows the parameters used for the four networks (SOM, SIM, MLHL-SIM and Beta-SIM) trained over this 2-D dataset, including the values calculated for their QME and TE.

Fig. 10 shows the converged weights on the artificial dataset, for each network. The Beta-SIM (Fig. 10d), after choosing adequate values for the α and β parameters, obtained weights that enclosed the data properly, performing better than the SOM (Fig. 10a) in terms of preserving the topology on the areas with more data (left side of the dataset), and was able to adapt the weights in the sparser area to be more representative of its properties than the SIM (Fig. 10b right side of the data). This research finding is also confirmed by results presented in Table 1, where Beta-SIM achieves the lowest MQE in comparison with the other three topology preserving models, and also

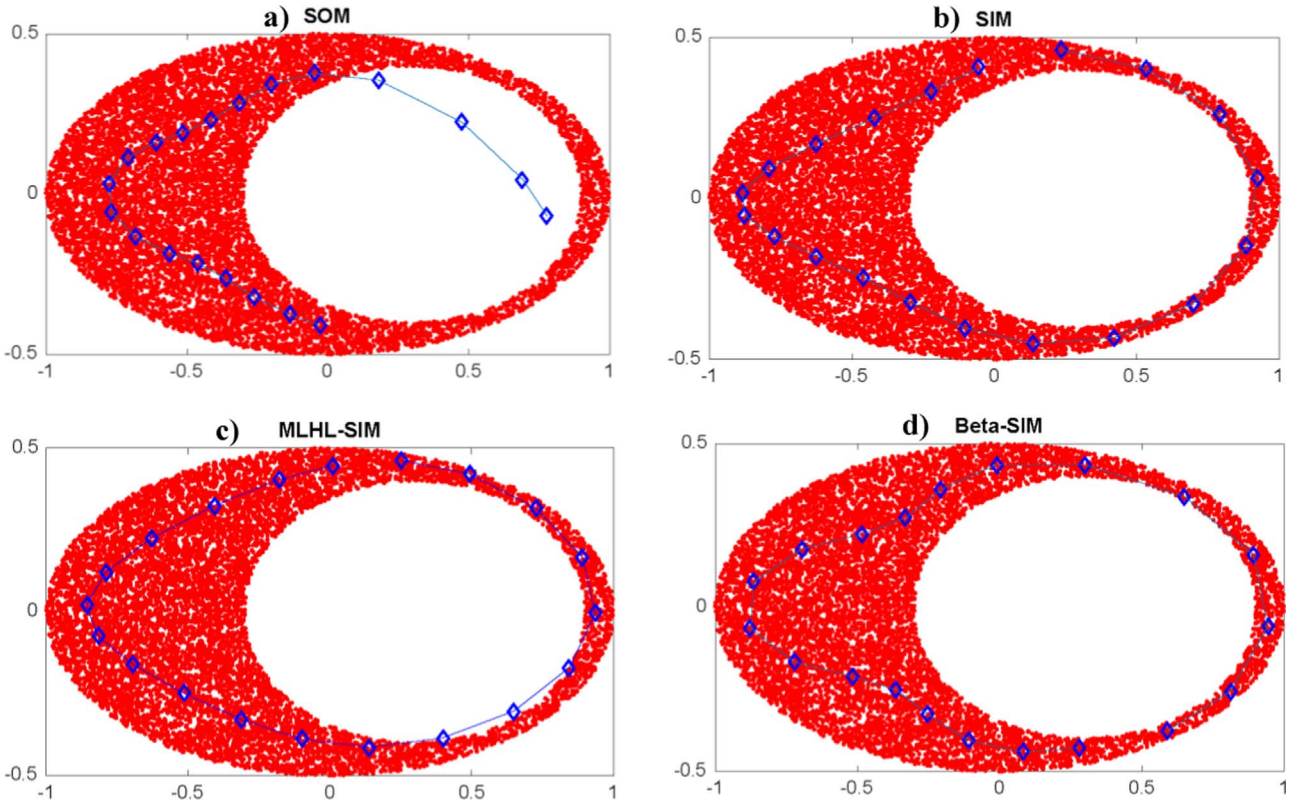

Fig. 10. SOM (a), SIM (b), MLHL-SIM (c) and Beta-SIM (d) final network grid (with 20 neurons) for artificial dataset 1.

Table 1
SOM, SIM, MLHL-SIM and Beta SIM parameters and its MQE and TE.

	SOM	SIM	MLHL-SIM	Beta-SIM
Size	[1,20]	[1,20]	[1,20]	[1,20]
Iterations	10,000	10,000	10,000	10,000
Learning rate	0.1	0.1	0.02	0.1
Neighborhood	10	20	20	15
p	–	–	0.8	–
α;β	–	–	–	*α=3;β=8*
MQE	0.1250	0.0995	0.1102	**0.0972**
TE	**0.0123**	0.0649	0.0526	0.0355

the second lowest value for the TE. The SOM performed better on the TE measure as the number of winning neurons on the left side of the ellipse (with high-density) is greater than in the Beta-SIM. However, in the case of the SOM, the convergence of the weights fails for areas with sparse data (right side of the Fig. 10a).

8.1.2. Artificial dataset 2

This artificial 2-dimensional dataset was created to test the Beta-SIM algorithm, in order to confirm the assertions outlined in Section 6. The dataset was generated using uniform distributions, which were centered at four different points in the 2-D space. Clusters were created with different densities in the range [−1,1]:

1. Cluster 1: 10,000 samples; center [−0.8, 0], radius 0.2.
2. Cluster 2: 50 samples; center [0,0], radius 0.2.
3. Cluster 3: 10 samples; center [0.4, 0.7], radius 0.1.
4. Cluster 4: 5 samples; center [0.8, −0.5], radius 0.1.

Different combinations of values of α and β were tested in order to analyze the effect on the training, and the results were compared with the SOM, SIM and MLHL-SIM algorithms in order to validate them. Experiments performed with this dataset were organized into 3 cases: where $\alpha=\beta$, where $\alpha > \beta$ and where $\alpha < \beta$.

8.1.3. Case 1: $\alpha=\beta$

With low values of $\alpha;\beta$ (i.e. $\alpha=\beta=4$, see Fig. 11), which theoretically are more appropriate for platykurtic residuals than for leptokurtic residuals (Corchado et al., 2004), the behavior of the network outperformed the models where higher values of $\alpha=\beta$ were used. Fig. 11h shows the best result of the network for values of $\alpha=\beta$, which is also compared with the best results of SOM (Fig. 11a), SIM (Fig. 11b) and MLHL-SIM (Fig. 11c–g).

In all cases, the final adaption of the network grid to the dataset is very similar, focusing on the clusters with higher density of samples (non-sparse clusters: cluster 1 (C1) and 2 (C2)), failing to cover and adapt to the other sparse clusters (C3 and C4).

8.1.4. Case 2: $\alpha > \beta$

In this case, residuals with high values should theoretically have more influence on the weights updating (see Fig. 7), therefore the adaptation of the network grid should cover all the clusters, even the sparse ones (C3 and C4). In other words, the final positions of some units (neurons of the network grid) will be in closer proximity with the samples of the sparse clusters.

Generally, the learning process is highly conditioned upon samples from clusters of high density (C1 and C2). In general, the final network grid adapts to these high density clusters. By using values of $\alpha > \beta$, the Beta-SIM algorithm reinforces the learning for the residuals of a sparse cluster sample, as in this dataset where clusters with few samples (low density) are far from clusters with high density of samples. In this case, the final network grid should adapt better over these sparse clusters (C3 and C4).

Fig. 12 shows that when $\alpha > \beta$, the Beta-SIM algorithm is able to assign a small number of units to cover these sparse clusters (C3 and C4), ensuring an effective clustering task even when "imbalanced" datasets, like this dataset, are involved.

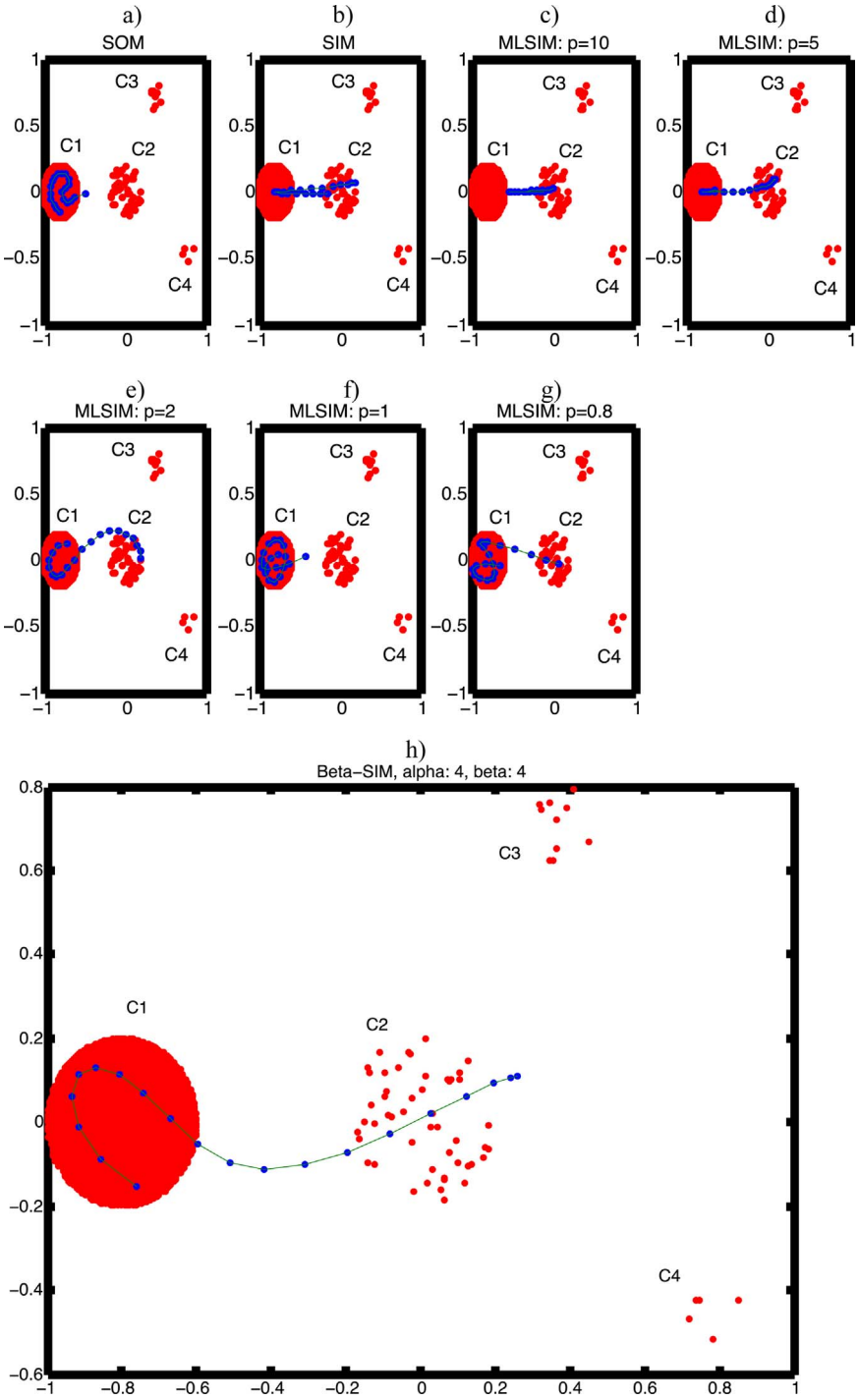

Fig. 11. Beta-SIM (h) results for values $\alpha=\beta$, and its comparison with SOM (a), SIM (b) and MLHL-SIM (c–g) algorithms: 20 neurons, 50,000 iterations; η=0.02; neighborhood function= Gaussian; MLHL-SIM: p=10, 5, 2, 1, 0.8; Beta-SIM $\alpha=\beta=4$.

8.1.5. Case 3: $\alpha < \beta$

In this case, by selecting appropriate values of the parameters $\alpha < \beta$ it is possible to make that neurons of the network only react to clusters with higher density of samples (C1, see Fig. 13).

When $\alpha < \beta$, (see Fig. 9), low residuals create larger weight updates, making the network grid adapt only to high density clusters (C1, see Fig. 13). Therefore, by selecting values where $\alpha < \beta$, it is possible to minimize or eliminate the effect of sparse datasets and/or data outliers on the learning process.

Based on the achieved results, it can be concluded that the combination of parameters α and β allows the selection of how the network grid fits over the dataset. These results seem to confirm that the network takes into account the sparser clusters when $\alpha > \beta$, and on the contrary, is capable of neglecting, for instance, the existence of outliers (high values of e) associated to noise in the data, by using values of $\alpha < \beta$.

8.2. Real datasets

In this Section *("8.2 Real dataset")*, 2 real datasets were used to analyze the behavior of the Beta-SIM algorithm to contrast the conclusions obtained in the experiments over the artificial datasets. Later, in Section *"8.3 Validation over 14 real benchmark datasets"*, the Beta-SIM algorithm is compared to other algorithms using statistical

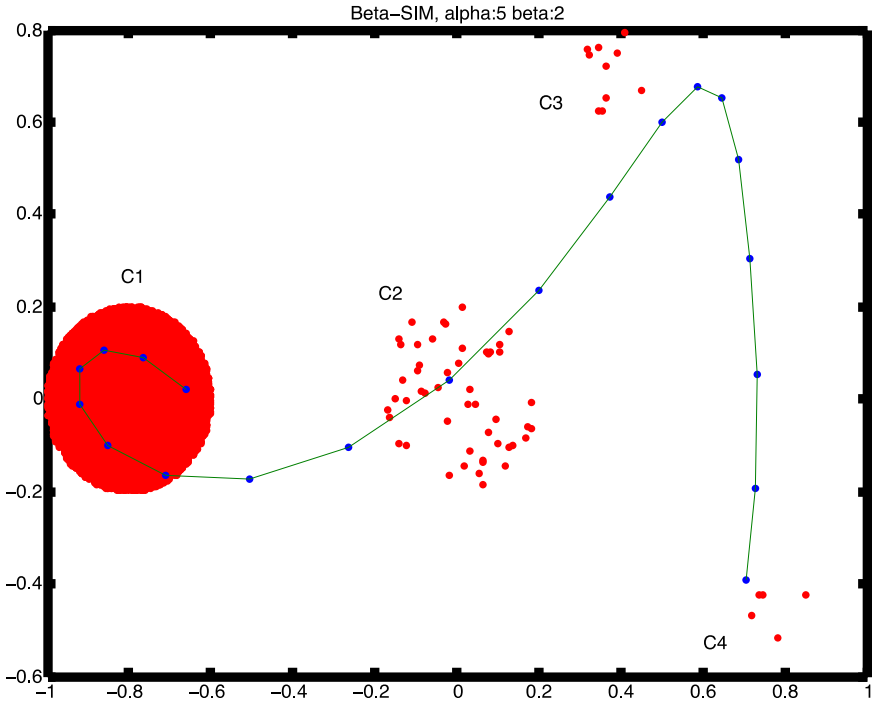

Fig. 12. Beta-SIM results for values $\alpha > \beta$; α=5, β=2, η=0.5, iterations=50,000, neighborhood function= Gaussian; neurons grid=[1,20].

tests over 14 real benchmark datasets. This makes a total of 16 real datasets used to test the behavior of Beta-SIM algorithm.

8.2.1. E. Coli dataset

The first real dataset used in these experiments is the well known E. Coli dataset from UCI repository (Lichman, 2013). The objective of this dataset is to predict the localization site of proteins by utilizing measurements of the cells' characteristics (cytoplasm, inner membrane, perisplasm, outer membrane, outer membrane lipoprotein, inner membrane lipoprotein inner membrane, cleavable signal sequence).

SOM, SIM, MLHL-SIM and Beta-SIM algorithms were applied over this dataset. The application of Topology-preserving models to these kind of tasks has previously proved interesting (Baruque and Corchado, 2014).

The dataset consists of 336 instances with 7 attributes, divided in 8 classes with sizes: 143(C1), 77(C2), 2(C3), 2(C4), 35(C5), 20(C6), 5(C7), 52(C8), which create an imbalanced dataset.

In all experiments a normalization of the dataset [−1,1] was performed and a 10-fold-cross validation was used.

Table 2 shows the parameters used for the algorithms throughout all the experiments. Parameters have been chosen in an experimental

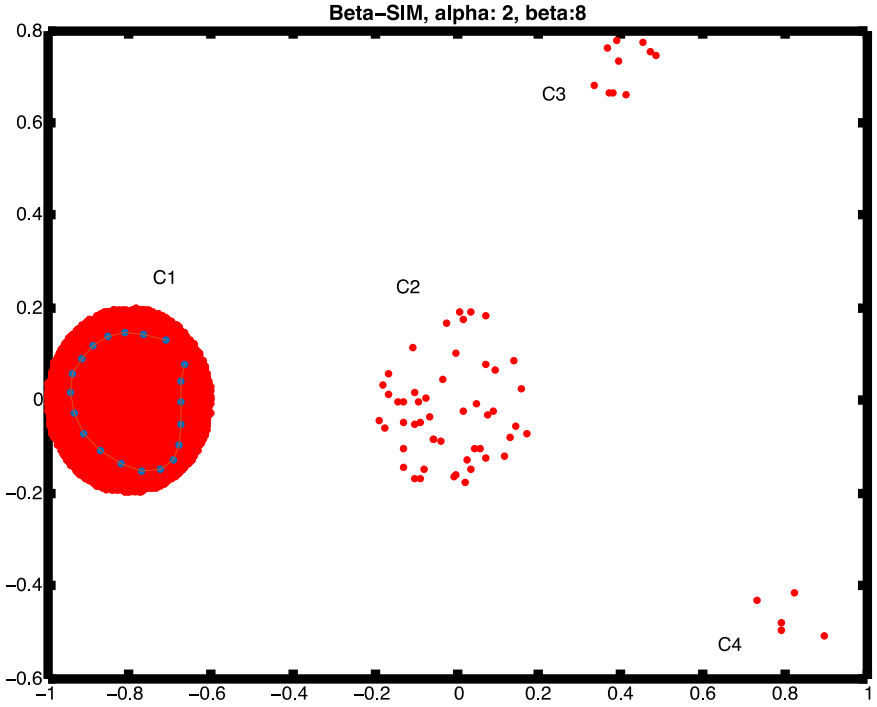

Fig. 13. Beta-SIM results for values $\alpha < \beta$; α=2, β=8, η=0.01, iterations=50,000, neighborhood function= Gaussian; neurons grid=[1,20].

Table 2
SOM, SIM, MLHL-SIM and Beta-SIM parameters and their associated measures: CE, MQE and TE for E. Coli dataset.

	SOM	SIM	MLHL-SIM	Beta-SIM
Size	[10,10]	[10,10]	[10,10]	[10,10]
Iterations	5,000	10,000	5,000	5,000
Learning rate	0.1	0.01	0.1	0.1
Neighborhood	10	5	10	15
p	–	–	0.9	–
α;β	–	–	–	*α=3;β=2*
CE	11,9%	11,3%	11%	**9,82%**
MQE	0,32	0,26	0,21	**0,20**
TE	**0,11**	0,33	0,63	0,54

process of trial and error. The best performing set of parameters was selected to later conduct all the experiments detailed in the comparison.

Table 2 also shows the results for the CE, TE, and MQE obtained for this dataset. It can be seen that Beta-SIM obtains the best CE and MQE. However, the best TE is obtained by SOM as expected.

In Fig. 14, the final adaptation of the network grids to the dataset is shown for the 4 algorithms (SOM, SIM, MLHL-SIM, Beta-SIM). The Beta-SIM network generates a clear network grid over the dataset, as it is more spread out over the dataset, especially the group at the top of the image (Fig. 14d), which seems to be a sparse cluster.

In the case of Beta-SIM, SIM, and MLHL-SIM the topology of the network grid is affected by groups placed at the top of the figure (Fig. 14b–c) and several neurons are dragged by this group. This effect is clearly reflected in the larger TE compared to the SOM network. For the SOM (Fig. 14a), the model does not spread the network grid over the group placed at the top, and hence the effect on the topology of the network is lower than in the case of the other algorithms, leading to a lower TE.

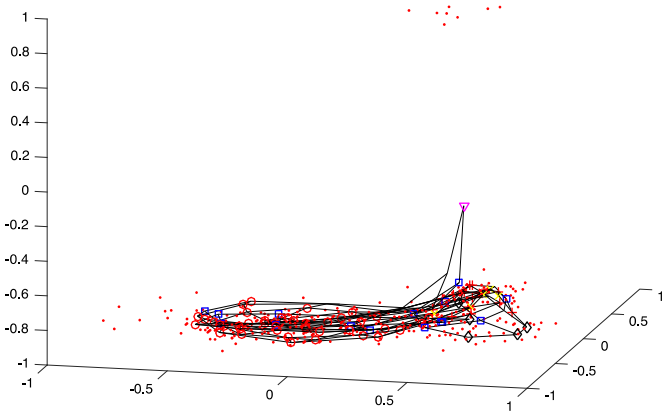

Fig. 15. Beta-SIM final network grid adaptation over the E. Coli dataset, with *α=2 and β=4*.

If we now modify the values of α and β, Beta-SIM is able to neglect the sparse clusters (top groups of Fig. 15), as they have high residual values (e) and the network considers them as outliers (see final network grid adaptation over the dataset in Fig. 15).

Fig. 16 shows the final map for SOM, SIM, MLHL-SIM, and Beta-SIM algorithms, where only the BMUs are displayed. Each BMU is labelled based on the training inputs to which it is reacting. This means that if neuron 10 is activated by 20 training inputs, and 19 of them belongs to class 1, this neuron will be labelled as class 1 (blue circle in Fig. 16).

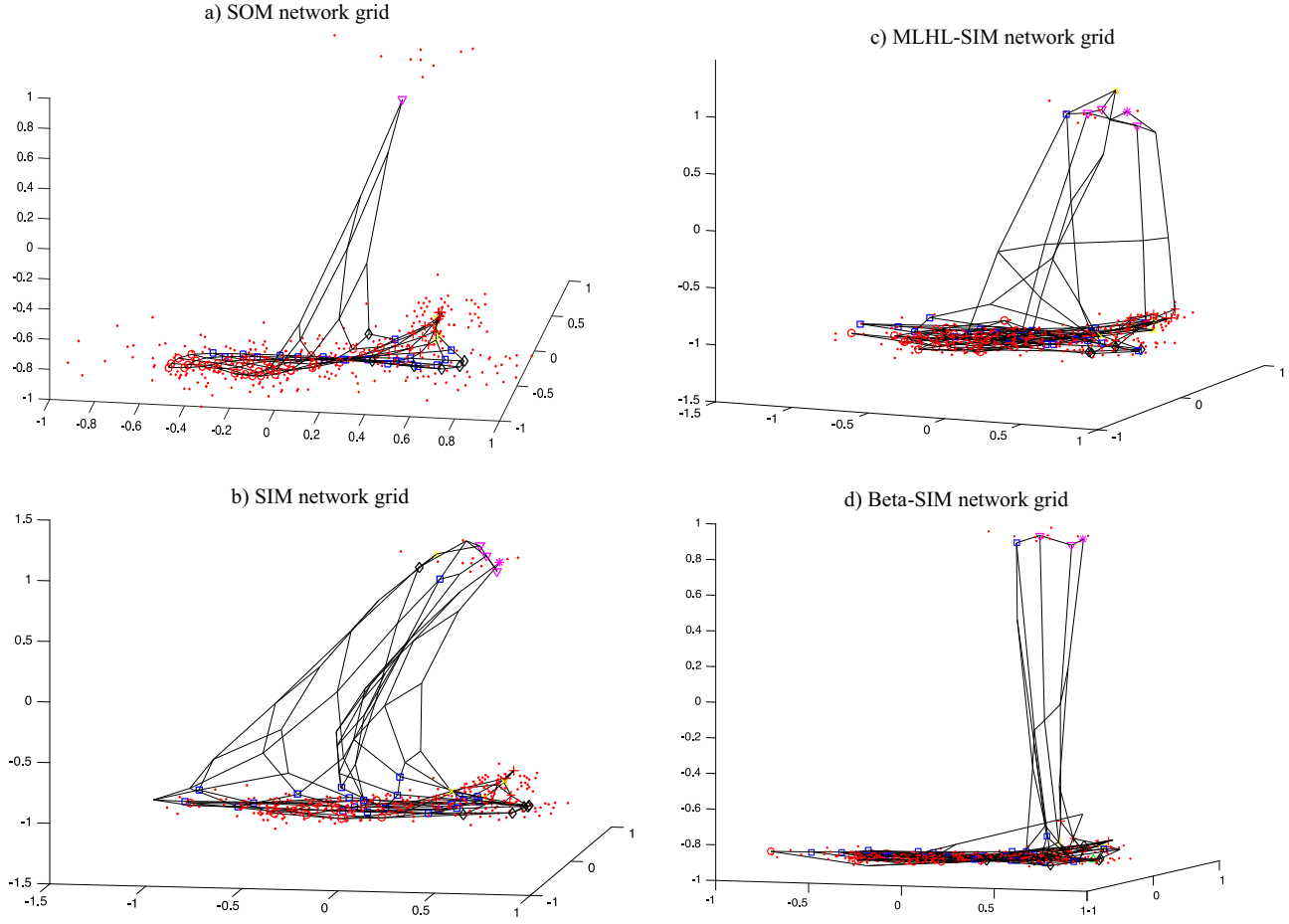

Fig. 14. SOM, SIM, MLHL-SIM and Beta-SIM final network grids adaptation over the E. Coli dataset.

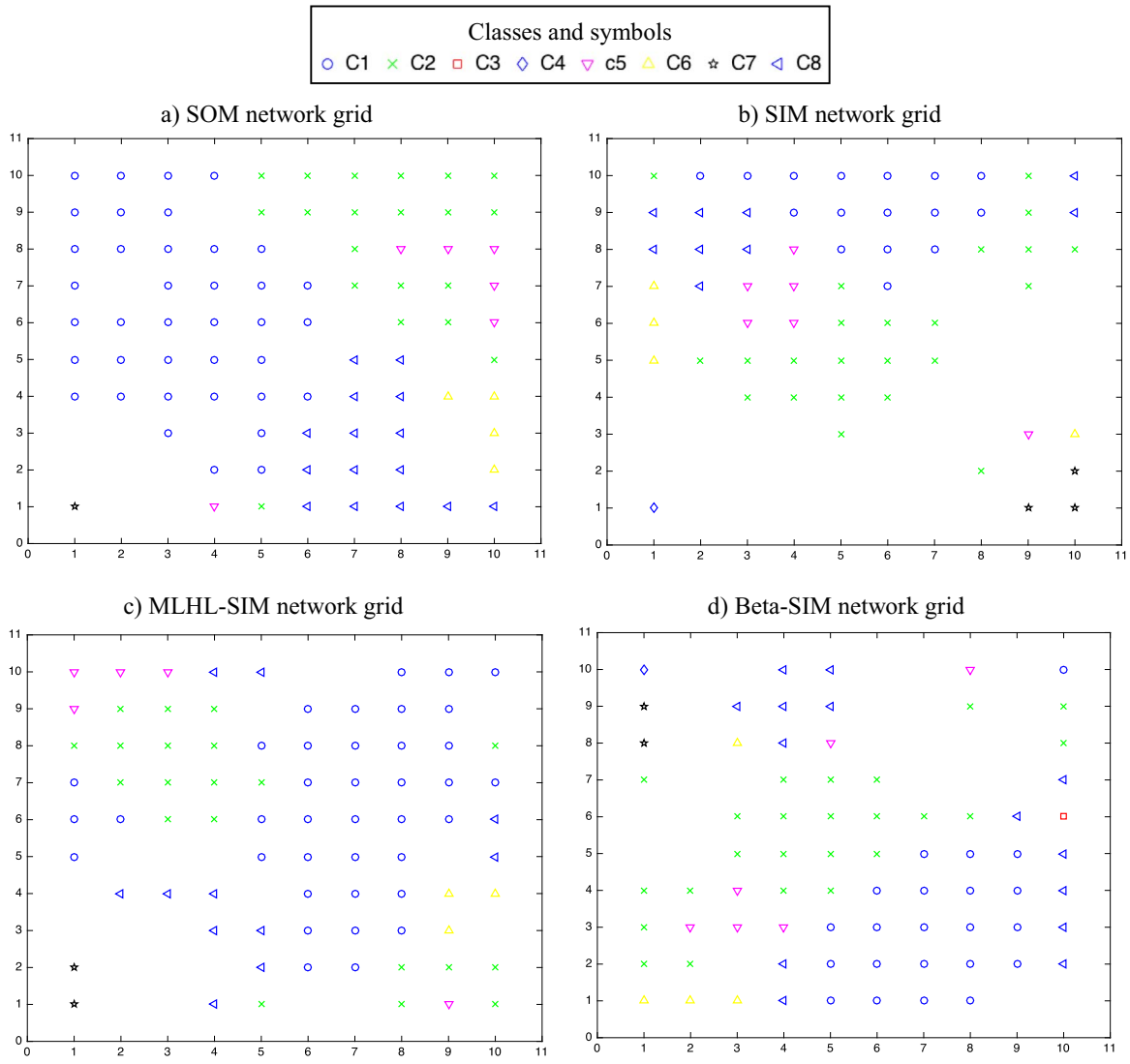

Fig. 16. SOM, SIM, MLHL-SIM and Beta-SIM final maps. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

Based on the maps from Fig. 16, only the Beta-SIM network (Fig. 16d) is able to assign at least one neuron to each class. Also, in the Beta-SIM map, the number of neurons assigned to each class is proportional to the size of the classes, so classes C1 and C2 have the highest number of BMUs, followed by classes C8, C5, and C6 respectively, and the lower number of BMUs are associated to sparse classes (C3, C4 and C7).

However, the SOM network (Fig. 16a) has the best organization, where neurons are not disordered and the classes are not mixed. In the case of Beta-SIM, several neurons appear disordered, for instance C8 (blue left-pointing triangle in Fig. 16) appears at the top-center and also at the bottom-right side.

Finally, if we present the final Beta-SIM map when $\alpha=2$ and $\beta=4$ (see Fig. 17), we get as expected the opposite results to Beta-SIM when $\alpha=3$ and $\beta=2$. So now the organization of the map is much clearer, but it is not able to assign BMUs to all classes.

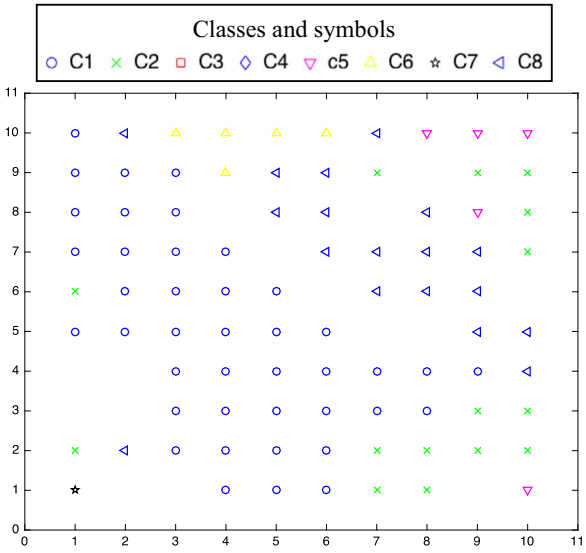

Fig. 17. Beta-SIM final map when $\alpha=2$ and $\beta=4$.

8.2.2. High precision machine dataset

Our second real dataset was obtained from a dynamic high-precision machinery used for the manufacturing of metal dental pieces (Redondo et al., 2015; Vera et al., 2013). This real industrial use case is described by an initial dataset of 190 samples obtained by a dental scanner in the manufacturing of dental pieces with different tool types

Table 3
SOM, SIM, MLHL-SIM and Beta-SIM parameters and their MQE and TE for the high precision machine dataset.

	SOM	SIM	MLHL-SIM	Beta-SIM
Size	[15,15]	[15,15]	[15,15]	[15,15]
Iterations	50,000	100,000	50,000	50,000
Learning rate	0.01	0.01	0.1	0.1
Neighborhood	10	5	10	5
p	–	–	0.5	–
α;β	–	–	–	*α=2;β=3*
MQE	0,55	0,57	0,49	**0,38**
TE	**0,10**	0,36	0,26	0,24

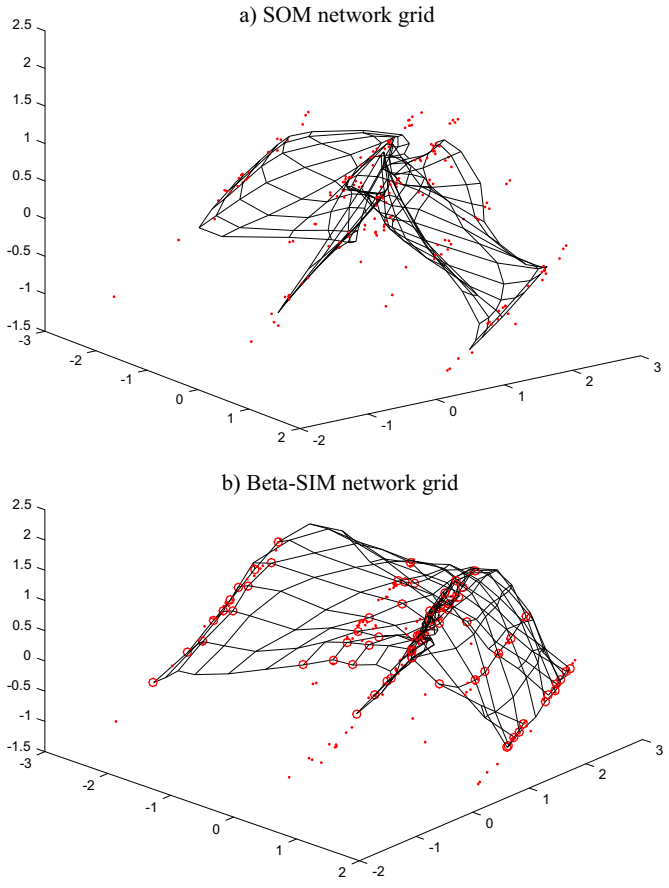

Fig. 18. SOM and Beta-SIM final network grids adaptation over the dataset.

Table 5
p-values for CE (Beta-SIM against all algorithms).

Dataset	SOM	ViSOM	SIM	MLHL-SIM	GNG
1	0.6580	1.0000	1.0000	0.9999	0.989
2	1.0000	0.9837	0.9837	1.0000	1.0000
3	1.0000	1.0000	0.9989	1.0000	0.9982
4	0.9999	0.9999	0.9999	0.9973	0.9999
5	1.0000	1.0000	1.0000	0.6053	0.9996
6	1.0000	0.2786	0.9676	0.8666	1.0000
7	0.4614	0.0481*	0.5622	0.4231	0.9602
8	0.9612	0.7831	0.9971	0.2273	0.0139*
9	0.0007*	0.0000*	0.0000*	0.0002*	0.0000*
10	0.0033*	0.0059*	0.0118*	0.1622	0.0048*
11	0.1867	0.4699	0.9950	0.9683	0.7909
12	0.0013*	0.0002*	0.0489*	0.1306	0.3019
13	0.0000*	0.0000*	0.0003*	0.0004*	0.0044*
14	0.0000*	0.0000*	0.0000*	0.0000*	1.0000

◇Beta-SIM is significantly worse (significance level of 0.05) than the other model.
* Beta-SIM is significantly better (significance level of 0.05) than the other model.

Table 6
p-values for MQE (Beta-SIM against all algorithms).

Dataset	SOM	ViSOM	SIM	MLHL-SIM	GNG
1	0.0031*	0.0003*	0.9847	1.0000	0.9925
2	0.3168	0.5267	0.5422	0.7727	0.0068◇
3	0.0231*	0.0000◇	0.0000◇	0.0000◇	0.0000◇
4	0.0087*	0.6621	1.0000	0.9987	0.2649
5	0.9627	0.7475	0.5616	0.3808	0.0673
6	0.0000*	0.0565*	0.2433	0.8285	0.1819
7	0.0000*	0.0059◇	0.9994	0.0004◇	0.0000◇
8	0.0001*	0.0000*	0.0051*	0.0007*	0.0000*
9	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*
10	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*
11	0.0000*	0.1097	0.8189	0.9515	0.3771
12	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*
13	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*
14	0.0000*	0.0000*	0.0000*	0.0000*	0.0000*

* Beta-SIM is significantly better (significance level of 0.05) than the other model.
◇ Beta-SIM is significantly worse (significance level of 0.05) than the other model.

(flat, toric, spherical and drill) and is characterized by eleven input variables (number of pieces, type of dental piece, tool, radius, revolutions, feed rate X, Y and Z, thickness, initial temperature, and initial diameter of the tool).

The objective was to test the adaptation of the networks to this real dataset and compare the quality of the Beta-SIM against other algorithms. The MQE and TE were used to measure the quality of the final network grid in 4 algorithms: SOM, SIM, MLHL-SIM and Beta-SIM.

Table 3 shows the parameters used for the algorithms throughout all the experiments and the final MQE and TE for each algorithm.

Table 4
Benchmark datasets.

	Name of the dataset	Samples	Features	Classes	Source
Dataset 1	Liver Disorders	345	6	2	UCI repository
Dataset 2	Fertility	100	10	2	UCI repository
Dataset 3	Bank Marketing	45211	16	2	UCI repository
Dataset 4	Iris	150	4	3	UCI repository
Dataset 5	Wine	178	13	3	UCI repository
Dataset 6	Contraceptive	1473	9	3	UCI repository
Dataset 7	Car	1728	6	4	UCI repository
Dataset 8	Dermatology	358	33	6	UCI repository
Dataset 9	Image Segmentation	2310	19	7	UCI repository
Dataset 10	Landsat Satellite	6435	36	7	UCI repository
Dataset 11	Yeast	1484	8	10	UCI repository
Dataset 12	Pen-Based Recog. of Handwritten Digits	10992	16	10	UCI repository
Dataset 13	Optical Recognition of Handwritten Digits	5620	64	10	UCI repository
Dataset 14	Letter Recognition	20000	16	29	UCI repository

Table 7
p-values for TE (Beta-SIM against all algorithms).

Dataset	SOM	ViSOM	SIM	MLHL-SIM	GNG
1	0.0000◇	0.0000◇	0.9999	0.6811	0.0000*
2	0.7557	1.0000	1.0000	0.7557	0.0000*
3	0.0000◇	0.0000◇	0.0000◇	0.0000◇	0.0000*
4	0.6460	0.0001◇	0.9989	0.9721	0.0000*
5	0.0000◇	0.0000◇	0.0000◇	0.0008◇	0.0000*
6	0.0000◇	0.0000◇	0.7769	0.1592	0.0000*
7	0.0000◇	0.0000◇	0.0000◇	0.0000◇	0.0000*
8	0.0010◇	0.0001◇	0.7565	1.0000	0.0050◇
9	0.0000◇	0.0000◇	0.9999	1.0000	0.0000*
10	0.0000◇	0.0000◇	0.0000◇	0.0000◇	0.9796
11	0.0000◇	0.0000◇	0.4494	1.0000	0.0000*
12	0.0000◇	0.0000◇	0.1581	0.0517	0.0000*
13	0.0000◇	0.0000◇	0.0009◇	0.0001◇	0.0000*
14	0.0000◇	0.0000◇	0.0000◇	0.0000◇	0.0000*

* Beta-SIM is significantly better (significance level of 0.05) than the other model.
◇ Beta-SIM is significantly worse (significance level of 0.05) than the other model.

Parameters were chosen in an experimental process of trial and error. The best performing set of parameters was selected to later conduct all the experiments detailed in the comparison.

As expected, based on the results of previous experiments over different datasets, the SOM is the one which obtains the best TE, and the Beta-SIM obtains the best MQE. By modifying the parameters α and β it is possible to reduce the TE to values similar to SOM, but the MQE becomes worse.

In Fig. 18, the final adaptation of the network grids to this real dataset is shown for SOM and Beta-SIM. The Beta-SIM network generates a clearer network grid over the dataset, as it is more spread out over the dataspace.

8.3. Validation over 14 real benchmark datasets

In this subsection, the Beta-SIM algorithm is validated over 14 diverse real benchmark datasets with diverse number of samples and features by means of a statistical test (ANOVA + post-hoc analysis). The Beta-SIM algorithm is also compared with other well known topology preserving algorithms such as: SOM, ViSOM, SIM, MLHL-SIM, and GNG in order to validate its capabilities, comparing the methods in terms of CE, MQE and TE.

8.3.1. Benchmark datasets description

A total of 14 diverse benchmark datasets from the UCI Machine Repository were used to validate the Beta-SIM algorithm. In Table 4, a summary of these datasets is presented in terms of samples, features, and number of classes.

8.3.2. Results and statistical analysis

In all experiments over all datasets, a normalization of the dataset was performed, and a 10-fold-cross validation was used. Parameters were chosen in an experimental process of trial and error.

Table A1 (Appendix A) shows the average CE, MQE and TE ± their standard deviation (STD) for each algorithm with each dataset.

Table A2 (Appendix A) presents the parameters selected for each algorithm over the different experiments (for each dataset). In all cases, the number of neurons were the same for all algorithms in each experiment.

Tables 5–7 presents the p-values obtained after applying an ANOVA+post HOC statistical analysis for CE, MQE and TE, respectively (Beta-SIM against each algorithm).

After the statistical analysis presented in Tables 5–7, it can be concluded that in general terms, the novel Beta-SIM algorithm obtains better results when the number of classes of the datasets increases, that is to say, for complex high dimensional datasets. When the number of classes is larger than 7, in general, Beta-SIM obtains a significant improvement in relation to the CE and MQE measures, with the improvement being of greater magnitude in the MQE measures. However, as can be expected, the TE measure is worse than the other algorithms (except GNG). This is due to the better adaptation to the datasets forcing a deformation of the final neural grid. Nevertheless, when the number of classes is lower than 7, Beta-SIM results are statistically similar to the ones obtained by the other state-of-the-art algorithms.

The results of the experiments performed over these benchmark datasets help to draw broader conclusions about the behavior of the novel proposed method, showing how it improves the other models based on such three well-known quality measures (CE, MQE and TE).

Table A3 (Appendix A) summarizes the results of Tables 5–7 in terms of percentage (%) of improvement:

1. Positive values are the "%" of improvement of Beta-SIM in comparison with a specific algorithm, in relation to the three well-known quality measures.
2. Negative values are the "%" of deterioration of Beta-SIM in comparison with a specific algorithm, in relation to the three well-known quality measures.

Finally, based on the parameters selection for the Beta-SIM algorithm (Table A1 Appendix), it can be observed that when the number of classes is large (more than 7 classes) the best combination of α and β parameters is obtained for $\alpha > \beta$, and otherwise when the number of classes is small (less than 7 classes), the best combination is obtained when $\alpha=\beta$, or even $\alpha < \beta$ when the number of classes are very small (i.e. 2 or 3 classes). Therefore, it is worthy to apply Beta-SIM when analyzing complex high dimensional datasets.

9. Conclusions and future work

A novel algorithm called Beta-SIM has been presented and thoroughly analyzed in this study. Beta-SIM aims to obtain the best topology preserving map possible, in order to be used as a reliable tool in data visualization. Due to the inherent capabilities of the SIM, their combination with the BHL algorithm improves adaptation and visualization of datasets with a radial structure, as has been successfully shown in the tests. The main improvement of the algorithm is the capacity to adapt to sparse clusters or to neglect outliers depending on the combination of values of α and β and the task to be carried out.

Beta-SIM is therefore a powerful new tool for the data mining and big data communities and should take its place along with existing topology preserving maps.

Future work includes the application of the BHL rule to other topology preserving models, such as Self Organizing Maps (SOM), Visualization Inducted SOM (ViSOM) (Corchado and Baruque, 2012), and its combination with the use of ensembles to boost model performance (Baruque and Corchado, 2014; Haixiang et al., 2016). Also, Beta-SIM could be applied to analyze challenging datasets in order to solve problems in the field of the electric vehicles, energy efficiency, cybersecurity, big data, etc.

Acknowledgements

This work was supported by a FPU grant at the Spanish Ministry of Education (FPU13/03459).

Appendix A

See Tables A1–A3.

Table A1
Average testing CE/MQE/TE ± STD over the 14 benchmark datasets.

Dataset	Quality Measure	SOM	ViSOM	SIM	MLHL-SIM	GNG	Beta-SIM
1	CE	0.412 ± 0.0866	0.472 ± 0.1004	0.474 ± 0.0904	0.461 ± 0.0858	0.446 ± 0.0690	0.469 ± 0.0725
	MQE	0.314 ± 0.0352	0.326 ± 0.0320	0.272 ± 0.0282	0.265 ± 0.0225	0.256 ± 0.0223	0.263 ± 0.0308
	TE	0.090 ± 0.0456	0.064 ± 0.0467	0.313 ± 0.0735	0.348 ± 0.0646	0.713 ± 0.0724	0.308 ± 0.0595
2	CE	0.210 ± 0.1595	0.180 ± 0.1033	0.180 ± 0.1229	0.210 ± 0.1287	0.220 ± 0.1476	0.210 ± 0.1229
	MQE	1.226 ± 0.0824	1.035 ± 0.1118	1.037 ± 0.1110	1.055 ± 0.1264	0.937 ± 0.0968	1.121 ± 0.1381
	TE	0.030 ± 0.0483	0.090 ± 0.1287	0.090 ± 0.0568	0.030 ± 0.0675	0.440 ± 0.1430	0.090 ± 0.1101
3	CE	0.117 ± 0.0039	0.117 ± 0.0046	0.116 ± 0.0046	0.117 ± 0.0040	0.116 ± 0.0047	0.117 ± 0.0048
	MQE	1.223 ± 0.0103	1.005 ± 0.0094	1.057 ± 0.0170	1.037 ± 0.016	0.993 ± 0.0131	1.178 ± 0.0701
	TE	0.023 ± 0.0123	0.026 ± 0.0065	0.053 ± 0.0148	0.055 ± 0.0297	0.309 ± 0.0425	0.136 ± 0.0466
4	CE	0.047 ± 0.0632	0.047 ± 0.0549	0.060 ± 0.0734	0.067 ± 0.0544	0.047 ± 0.0549	0.046 ± 0.0820
	MQE	0.222 ± 0.0388	0.157 ± 0.0209	0.177 ± 0.0310	0.181 ± 0.0331	0.149 ± 0.0143	0.166 ± 0.0250
	TE	0.200 ± 0.0770	0.120 ± 0.0322	0.260 ± 0.1313	0.320 ± 0.1209	0.587 ± 0.1565	0.280 ± 0.1363
5	CE	0.045 ± 0.0353	0.045 ± 0.0365	0.045 ± 0.0450	0.079 ± 0.0614	0.051 ± 0.0513	0.044 ± 0.0511
	MQE	0.795 ± 0.0836	0.720 ± 0.0743	0.710 ± 0.0822	0.700 ± 0.0807	0.669 ± 0.0422	0.767 ± 0.0909
	TE	0.090 ± 0.0390	0.062 ± 0.0720	0.190 ± 0.0865	0.225 ± 0.0950	0.668 ± 0.1129	0.400 ± 0.1124
6	CE	0.538 ± 0.0448	0.541 ± 0.0571	0.546 ± 0.0416	0.538 ± 0.0536	0.560 ± 0.0432	0.550 ± 0.0417
	MQE	0.870 ± 0.0342	0.699 ± 0.0338	0.742 ± 0.0378	0.717 ± 0.0342	0.493 ± 0.0242	0.654 ± 0.0360
	TE	0.040 ± 0.0255	0.028 ± 0.0088	0.081 ± 0.0286	0.073 ± 0.0505	0.529 ± 0.0469	0.125 ± 0.0364
7	CE	0.210 ± 0.0499	0.231 ± 0.0413	0.207 ± 0.0287	0.211 ± 0.0413	0.192 ± 0.0491	0.177 ± 0.0268
	MQE	1.206 ± 0.0243	1.045 ± 0.0305	1.083 ± 0.0257	1.036 ± 0.0252	0.851 ± 0.0097	1.086 ± 0.0286
	TE	0.046 ± 0.0142	0.063 ± 0.0177	0.136 ± 0.0266	0.120 ± 0.0243	0.667 ± 0.0217	0.443 ± 0.0393
8	CE	0.031 ± 0.0406	0.081 ± 0.0403	0.061 ± 0.0677	0.104 ± 0.0722	0.131 ± 0.0514	0.050 ± 0.0318
	MQE	1.818 ± 0.0638	1.834 ± 0.0683	1.783 ± 0.0585	1.800 ± 0.0677	2.261 ± 0.0606	1.679 ± 0.0452
	TE	0.053 ± 0.0243	0.031 ± 0.0337	0.132 ± 0.0708	0.162 ± 0.0721	0.070 ± 0.0509	0.167 ± 0.0818
9	CE	0.143 ± 0.0262	0.195 ± 0.0237	0.165 ± 0.0249	0.147 ± 0.0246	0.151 ± 0.0187	0.098 ± 0.0162
	MQE	0.553 ± 0.0197	0.532 ± 0.0298	0.487 ± 0.0227	0.464 ± 0.0265	0.511 ± 0.0183	0.381 ± 0.0206
	TE	0.071 ± 0.0195	0.040 ± 0.0246	0.162 ± 0.0350	0.161 ± 0.0364	0.345 ± 0.0362	0.158 ± 0.0404
10	CE	0.158 ± 0.0106	0.157 ± 0.0100	0.155 ± 0.0216	0.148 ± 0.0204	0.157 ± 0.0081	0.132 ± 0.0125
	MQE	0.644 ± 0.0084	0.655 ± 0.0103	0.597 ± 0.0197	0.592 ± 0.0083	0.681 ± 0.0114	0.489 ± 0.0075
	TE	0.120 ± 0.0184	0.063 ± 0.0132	0.197 ± 0.0460	0.243 ± 0.0606	0.380 ± 0.0252	0.392 ± 0.0600
11	CE	0.500 ± 0.0359	0.489 ± 0.0472	0.462 ± 0.0538	0.468 ± 0.0394	0.478 ± 0.0427	0.451 ± 0.0530
	MQE	0.321 ± 0.0274	0.292 ± 0.0170	0.279 ± 0.0207	0.276 ± 0.0190	0.286 ± 0.0203	0.267 ± 0.0217
	TE	0.059 ± 0.0174	0.051 ± 0.0193	0.297 ± 0.0383	0.333 ± 0.0598	0.520 ± 0.0527	0.330 ± 0.0340
12	CE	0.106 ± 0.0123	0.109 ± 0.0147	0.098 ± 0.0150	0.096 ± 0.0101	0.093 ± 0.0135	0.081 ± 0.0114
	MQE	0.883 ± 0.0102	0.897 ± 0.0115	0.753 ± 0.0129	0.741 ± 0.0138	0.901 ± 0.0123	0.668 ± 0.0152
	TE	0.055 ± 0.0090	0.042 ± 0.0037	0.088 ± 0.0136	0.082 ± 0.0135	0.290 ± 0.0268	0.105 ± 0.0176
13	CE	0.105 ± 0.0098	0.120 ± 0.0146	0.092 ± 0.0103	0.091 ± 0.0115	0.087 ± 0.0178	0.064 ± 0.0128
	MQE	2.703 ± 0.0228	2.811 ± 0.0173	2.450 ± 0.0203	2.454 ± 0.0213	2.701 ± 0.0229	2.247 ± 0.0289
	TE	0.038 ± 0.0092	0.037 ± 0.0082	0.063 ± 0.0123	0.058 ± 0.0121	0.293 ± 0.0273	0.091 ± 0.0095
14	CE	0.545 ± 0.0169	0.516 ± 0.0141	0.503 ± 0.0101	0.497 ± 0.0139	0.363 ± 0.0175	0.362 ± 0.0122
	MQE	0.605 ± 0.0044	0.568 ± 0.0045	0.527 ± 0.0028	0.521 ± 0.0044	0.448 ± 0.0031	0.432 ± 0.0043
	TE	0.074 ± 0.0109	0.075 ± 0.0109	0.193 ± 0.0129	0.204 ± 0.0196	0.483 ± 0.0154	0.247 ± 0.0191

Table A2
Algorithms parameters for each Benchmark dataset.

Dataset	Parameters	SOM	ViSOM	SIM	MLHL-SIM	GNG	Beta-SIM
1	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	30,000	20,000	30,000	30,000	30,000	30,000
	Learning rate	0.1	0.05	0.05	0.05	–	0.05
	Neighborhood	12	10	10	10	–	6
	p	–	–	–	0.5	–	–
	α;β	–	–	–	–	–	*α=3;β=4*
	λ_{ViSOM}	–	0,25	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	200; 0.5; 0.9; 5; 0.01; 0.001	–
2	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	5,000	5,000	5,000	5,000	10,000	5,000
	Learning rate	0.1	0.05	0.1	0.1	–	0.1
	Neighborhood	12	10	10	10	–	6
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=3;β=3.5*
	λ_{ViSOM}	–	0,25	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	200; 0.5; 0.9; 5; 0.01; 0.001	–
3	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	30,000	30,000	30,000	30,000	30,000	50,000
	Learning rate	0.05	0.05	0.05	0.05	–	0.05
	Neighborhood	12	10	10	10	–	6
	p	–	–	–	0.5	–	–
	α;β	–	–	–	–	–	*α=3;β=3.5*
	λ_{ViSOM}	–	0,25	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.01; 0.001	–
4	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	5,000	5,000	5,000	5,000	5,000	5,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	10		5	10	–	12
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=3;β=3*
	λ_{ViSOM}	–	0,1	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.1; 0.001	–
5	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	5,000	5,000	5,000	5,000	5,000	5,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	12	10	–	12
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=4;β=4*
	λ_{ViSOM}	–	0.1	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	50; 0.5; 0.9; 5; 0.1; 0.0001	–
6	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[15,10]
	Iterations	10,000	10,000	10,000	10,000	10,000	50,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	12	10	–	5
	p	–	–	–	1.2	–	–
	α;β	–	–	–	–	–	*α=4;β=4*
	λ_{ViSOM}	–	0.2	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	50; 0.5; 0.9; 5; 0.1; 0.0001	–
7	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	10,000	5,000	10,000	10,000	10,000	10,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	10	10	12	10	–	6
	p	–		–	0.9	–	–
	α;β	–		–	–	–	*α=5;β=3*
	λ_{ViSOM}	–	0.2	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.01; 0.0001	–
8	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	5,000	5,000	5,000	5,000	5,000	5,000
	Learning rate	0.1	0.1	0.05	0.05	–	0.1
	Neighborhood	10	10	10	10	–	6
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.3	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	50; 0.5; 0.9; 5; 0.01; 0.0001	–
9	**Size**	[15,20]	[15,20]	[15,20]	[15,20]	–	[15,20]
	Iterations	10,000	5,000	10,000	10,000	10,000	10,000
	Learning rate	0.1	0.1	0.05	0.1	–	0.1
	Neighborhood	10	10	10	10	–	6
	p	–		–	0.9	–	–
	α;β	–		–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.25	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.01; 0.0001	–
10	**Size**	[15,20]	[15,20]	[15,20]	[15,20]	–	[15,20]
	Iterations	30,000	30,000	30,000	30,000	30,000	30,000

(continued on next page)

Table A2 *(continued)*

Dataset	Parameters	SOM	ViSOM	SIM	MLHL-SIM	GNG	Beta-SIM
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	10	10	10	10	–	6
	p	–	–	–	1.1	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.25	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.01; 0.0001	–
11	**Size**	[10,15]	[10,15]	[10,15]	[10,15]	–	[10,15]
	Iterations	10,000	10,000	10,000	10,000	10,000	10,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	10	10	–	6
	p	–	–	–	0.5	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.1	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	50; 0.5; 0.9; 5; 0.01; 0.0001	–
12	**Size**	[15,20]	[15,20]	[15,20]	[15,20]	–	[15,20]
	Iterations	50,000	50,000	50,000	50,000	50,000	50,000
	Learning rate	0. 1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	8	8	–	6
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.1	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.001; 0.0001	–
13	**Size**	[10,20]	[10,20]	[10,20]	[10,20]	–	[10,20]
	Iterations	40,000	40,000	40,000	40,000	40,000	40,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	12	12	–	6
	p	–	–	–	0.9	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	–	–	–	100; 0.5; 0.9; 5; 0.001; 0.0001	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***						
14	**Size**	[20,25]	[20,25]	[20,25]	[20,25]	–	[20,25]
	Iterations	80,000	80,000	80,000	80,000	80,000	80,000
	Learning rate	0.1	0.1	0.1	0.1	–	0.1
	Neighborhood	12	10	12	12	–	6
	p	–	–	–	0.5	–	–
	α;β	–	–	–	–	–	*α=4;β=3*
	λ_{ViSOM}	–	0.1	–	–	–	–
	λ_{GNG}; α_{GNG};β_{GNG}; a_{max}; ***Wl; Wn***	–	–	–	–	100; 0.5; 0.9; 5; 0.005; 0.0001	–

Table A3
Summary of the results in terms of the three well-known quality measures for each benchmark dataset.

Dataset	Quality Measure	SOM	ViSOM	SIM	MLHL-SIM	GNG
	CE	-6%	0%	1%	-1%	-2%
1	MQE	**16%**	**19%**	3%	1%	-3%
	TE	-22%	-24%	1%	4%	**41%**
	CE	0%	-3%	-3%	0%	**1%**
2	MQE	9%	-8%	-8%	-6%	-20%
	TE	-6%	0%	0%	-6%	**35%**
	CE	0%	0%	0%	0%	0%
3	MQE	**4%**	**-17%**	**-11%**	**-14%**	**-19%**
	TE	-11%	-11%	-8%	-8%	**17%**
	CE	0%	0%	1%	2%	0%
4	MQE	**25%**	-6%	6%	8%	-11%
	TE	-8%	-16%	-2%	4%	**31%**
	CE	0%	0%	0%	4%	1%
5	MQE	4%	-7%	-8%	-10%	-15%
	TE	-31%	-34%	-21%	-18%	**27%**
	CE	-1%	-1%	0%	-1%	1%
6	MQE	**25%**	**6%**	12%	9%	-33%
	TE	-9%	-10%	-4%	-5%	**40%**
	CE	3%	**5%**	3%	3%	2%
7	MQE	**10%**	**-4%**	0%	**-5%**	**-28%**
	TE	-40%	-38%	-31%	-32%	**22%**
	CE	-2%	3%	1%	5%	**8%**
8	MQE	**8%**	**8%**	**6%**	**7%**	**26%**
	TE	-11%	-14%	-4%	-1%	-10%
	CE	**5%**	**10%**	**7%**	**5%**	**5%**
9	MQE	**31%**	**28%**	**22%**	**18%**	**25%**
	TE	-9%	-12%	0%	0%	**19%**
	CE	**3%**	**3%**	**2%**	**2%**	**3%**
10	MQE	**24%**	**25%**	**18%**	**17%**	**28%**
	TE	-27%	-33%	-20%	-15%	-1%
	CE	5%	4%	1%	2%	3%
11	MQE	**17%**	9%	4%	3%	7%
	TE	-27%	-28%	-3%	0%	**19%**
	CE	**3%**	**3%**	**2%**	2%	1%
12	MQE	**24%**	**26%**	**11%**	**10%**	**26%**
	TE	-5%	-6%	-2%	-2%	**19%**
	CE	**4%**	**6%**	**3%**	**3%**	**2%**
13	MQE	**17%**	**20%**	**8%**	**8%**	**17%**
	TE	-5%	-5%	-3%	-3%	**20%**
	CE	**18%**	**15%**	**14%**	**14%**	0%
14	MQE	**29%**	**24%**	**18%**	**17%**	**4%**
	TE	-17%	-17%	-5%	-4%	**24%**

Red color: Beta-SIM is significantly better (significance level of 0.05) than the other model.
Blue color: Beta-SIM is significantly worse (significance level of 0.05) than the other model.

References

Akhand, M.A.H., Murase, K., 2012. Ensembles of neural networks based on the alteration of input feature values. Int. J. Neural Syst. 22 (1), 77–87.

Baruque, B., Corchado, E., 2009. A novel ensemble of scale-invariant feature maps. In: Kurzynski, M., Wozniak, M. (Eds.), Computer Recognition Systems 3. Springer Berlin Heidelberg, Berlin, Heidelberg, 265–273.

Baruque, B., Corchado, E., 2011. Fusion Methods for Unsupervised Learning Ensembles 322. Springerhttp://www.springer.com/us/book/9781852338831.

Baruque, B., Corchado, E., 2014. WeVoS scale invariant map. Inf. Sci. 280, 307–321.

Baruque, B., Corchado, E., Yin, H., 2011. The s2-ensemble fusion algorithm. Int. J. Neural Syst. 21 (6), 505–525.

Bishop, C.M., Svensén, M., Williams, C.K.I., 1998. GTM: the generative topographic mapping. Neural Comput. 10 (1), 215–234.

Chen, N., Ribeiro, B., Vieira, A., Chen, A., 2013. Clustering and visualization of bankruptcy trajectory using self-organizing map. Expert Syst. Appl. 40 (1), 385–393.

Cho, S.-B., 2000. Ensemble of structure-adaptive self-organizing maps for high performance classification. Inf. Sci. 123 (1–2), 103–114.

Corchado, C., Fyfe, E., 2002. The Scale Invariant Map And Maximum Likelihood Hebbian Learning. vol. 82, pp. 245–249.

Corchado, E., Fyfe, C., 2002. The Scale Invariant Map and Maximum Likelihood Hebbian Learning. Presented at the International Conference on Knowledge-Based & Intelligent Information & Engineering Systems. vol. 82

Corchado, E., Baruque, B., 2012. WeVoS-ViSOM: an ensemble summarization algorithm for enhanced data visualization. Neurocomputing 75 (1), 171–184.

Corchado, E., MacDonald, D., Fyfe, C., 2004. Maximum and minimum likelihood Hebbian learning for exploratory projection pursuit. Data Min. Knowl. Discov. 8 (3), 203–225.

Dietterich, T.G., 2000. Ensemble methods in machine learning. In: Multiple Classifier Systems: First International Workshop, MCS 2000 Cagliari, Italy, June 21–23, 2000 Proceedings, Springer, Berlin, Heidelberg, pp. 1–15.

Fritzke, B., 1994. Growing cell structures—a self-organizing network for unsupervised and supervised learning. Neural Netw. 7 (9), 1441–1460.

Fritzke, B., 1995. A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems 7, pp. 625–632.

Fuertes, J.J., Domínguez, M., Reguera, P., Prada, M.A., Díaz, I., Cuadrado, A.A., 2010. Visual dynamic model based on self-organizing maps for supervision and fault detection in industrial processes. Eng. Appl. Artif. Intell. 23 (1), 8–17.

Fyfe, C., 1997. A neural network for PCA and beyond. Neural Process. Lett. 6 (1–2), 33–41.

Fyfe, C., 2005. Hebbian Learning and Negative Feedback Networks. Springer, http://www.springer.com/us/book/9783642162046.

Fyfe, C., 2005. The topographic product of experts. In: Artificial Neural Networks: Biological Inspirations - Icann 2005, Pt 1, Proceedings, Berlin, vol. 3696, pp. 397–402.
Ghassany, M., Bennani, Y., 2015. Collaborative fuzzy clustering of variational Bayesian generative topographic mapping. Int. J. Comput. Intell. Appl. 14 (1), 1550001.
Haimoudi, E.K., Fakhouri, H., Cherrat, L., Ezziyyani, M., 2016. Towards a new approach to improve the classification accuracy of the Kohonen's self-organizing map during learning process. Int. J. Adv. Comput. Sci. Appl 7 (3), 230–236.
Haixiang, G., Yijing, L., Yanan, L., Xiao, L., Jinling, L., 2016. BPSO-Adaboost-KNN ensemble learning algorithm for multi-class imbalanced data classification. Eng. Appl. Artif. Intell. 49, 176–193.
Huang, W., Yin, H., 2009. ViSOM for dimensionality reduction in face recognition. In: Advances in Self-Organizing Maps, 7th International Workshop, WSOM 2009, St. Augustine, FL, USA, June 8–10, 2009. Proceedings, pp. 107–115.
Jeong, Y., Lee, K., Yoon, B., Phaal, R., 2015. Development of a patent roadmap through the generative topographic mapping and bass diffusion model. J. Eng. Technol. Manag 38, 53–70.
Kohonen, T., 1998. The self-organizing map. Neurocomputing 21 (1–3), 1–6.
Kohonen, T., 2013. Essentials of the self-organizing map. Neural Netw. 37, 52–65.
Lichman, M., 2013. UCI Machine Learning Repository (School of Information and Computer Sciences). University of California, Irvine.
Martinetz, T., 1993. Competitive Hebbian Learning Rule Forms Perfectly Topology Preserving Maps. In: Gielen, S., Kappen, B., (Eds.), ICANN '93: Proceedings of the International Conference on Artificial Neural Networks Amsterdam, The Netherlands 13–16 September 1993, Springer, London, pp. 427–434.
Martinetz, T.M., Berkovich, S.G., Schulten, K.J., 1993. 'Neural-gas' network for vector quantization and its application to time-series prediction. Neural Netw. IEEE Trans. On 4 (4), 558–569.
Mendel, J.M. (Ed.), 1994. A Prelude to Neural Networks: Adaptive and Learning Systems. Prentice Hall Press, Upper Saddle River, NJ, USA.
Mohebi, E., Bagirov, A., 2016. Constrained self organizing maps for data clusters visualization. Neural Process. Lett. 43 (3), 849–869.
Pölzlbauer, G., 2004. Survey and comparison of quality measures for self-organizing maps. In: Proceedings of the Fifth Workshop on Data Analysis (WDA'04), Sliezsky dom, Vysoké Tatry, Slovakia, pp. 67–82.
Redondo, R., Sedano, J., Vera, V., Hernando, B., Corchado, E., 2015. A novel hybrid intelligent system for multi-objective machine parameter optimization. Pattern Anal. Appl 18 (1), 31–44.
Vera, V., Corchado, E., Redondo, R., Sedano, J., García, Á.E., 2013. Applying soft computing techniques to optimise a dental milling process. Neurocomputing 109, 94–104.
Wang, X., Gupta, A., 2015. Unsupervised learning of visual representations using videos. In: The IEEE International Conference on Computer Vision (ICCV).
Wu, Y., Doyle, T.K., Fyfe, C., 2011. Multi-layer topology preserving mapping for K-means clustering. In: Yin, H., Wang, W., RaywardSmith, V. (Eds.), Intelligent Data Engineering and Automated Learning – Ideal 2011 6936. Springer-Verlag, Berlin, 84–91.
Zapater, M., Fraga, D., Malagon, P., Bankovic, Z., Moya, J.M., 2015. Self-organizing maps versus growing neural gas in detecting anomalies in data centres. Log. J. Igpl 23 (3), 495–505.