Improving the quality of self-organizing maps by self-intersection avoidance

Ezequiel López-Rubio

Abstract

The quality of self-organizing maps has always been a key issue to practitioners. Smooth maps convey information about input datasets in a clear manner. Here a method is presented to modify the learning algorithm of self-organizing maps to reduce the number of topology errors, so that the obtained map has a better quality at the expense of increased quantization error. It is based on avoiding maps which self-intersect or nearly so, since these states are related to low quality. Our approach is tested with synthetic data and real data from visualization, pattern recognition and computer vision applications, with satisfactory results.

Index Terms
self-organizing map topologies, self-organizing map quality, self-intersection, visualization

I. INTRODUCTION

SINCE the proposal of the self-organizing map [31], [32] there has been a keen interest in the quality of the obtained maps. In this context, quality means that the map defines a smooth transformation from the lattice to the input space and vice versa, so that the topology relations are preserved as much as possible [44], [62], [41]. This is advantageous since a good quality map provides a better insight to the structure of the input dataset [18]. However, this should not be done at the expense of spoiling the vector quantization property of the map [20], since it is necessary that the units of the map represent faithfully the clusters present in the dataset. Consequently, there is a fundamental balance between map quality and vector quantization which must guide the construction of this kind of neural networks [25].

Much work has been done on the evaluation of trained maps. One of the earliest quality measures is the topographic product [2], which is sensitive to large scale topology disorder. It was soon followed by the topology error [30], which tests whether the first and second best matching units for an input sample are neighbors; and the topographic function [62], which analyzes the receptive fields of the units to account for nonlinear data manifolds. It was realized that topology errors lead to tensions in the map structure, which hampers learning [12]. The visualization of complex datasets is particularly sensitive to these problems [33], [58]. It must also be pointed out that the assessment can be done from the point of view of dimensionality reduction, which is a broad framework that can be used to evaluate other manifold learning models [37], [68].

A common denominator of this research is that the quality check is done after the map has been trained. That is, the emphasis is on evaluating different maps or learning algorithms in terms of their topology preservation capability. Even in the cases which study the evolution of the disorder over training time [12], the aim is to understand the behavior of the learning algorithm without modification, i.e. no attempt is done to enhance the performance of the algorithm in terms of map quality. A possible reason is that many assessment measures involve large computational requirements which prevents their use during the training process [61]. Here our goal is the development of a way to improve already existing self-organizing models by decreasing the topology errors of the generated maps.

The structure of this paper is as follows. First some fundamental concepts are reviewed; they will be used through the paper (Section II). Then the method to improve the quality of self-organizing maps is presented (Section III). Experiments with synthetic and real data are shown in Section IV. Some important properties of the proposal are discussed in Section V. Finally, Section VI is devoted to conclusions.

II. BASIC CONCEPTS

In this section the fundamental concepts which this work is based on are reviewed. First a brief outline of the original Kohonen’s Self-Organizing Feature Map (SOFM, [31]) with rectangular topology is presented, which is used as the base model (Subsection II-A). We also consider the well known Visualization-Induced Self Organizing Map (ViSOM, [65]). It must be highlighted that the proposal could be applied to many other self-organizing map models, with very little changes. Then some non equivalent definitions of topology error for self-organizing maps which have been proposed in literature (Subsection II-B) are explored. This leads us to the concept of self-intersection, which is at the root of the proposed approach (Subsection II-C).
A. Review of two self-organizing map models

Next we are going to review the original Kohonen’s SOFM and the ViSOM to present the notation that will be used through the paper. Let \( M \) be the number of neurons of the self-organizing map, which are arranged in a rectangular lattice of size \( a \times b \), where \( M = ab \). The topological distance between the units \( i \) and \( i' \), located at positions \((y_1, y_2)\) and \((y'_1, y'_2)\), is given by:

\[
d(i, i') = \sqrt{(y_1 - y'_1)^2 + (y_2 - y'_2)^2}
\]  

(1)

Every neuron \( i \) has a prototype vector \( w_i \) which represents a cluster of input samples. Please note that \( w_i \in \mathbb{R}^D \), where \( D \) is the dimension of the input space. At time step \( n \), a new sample \( x(n) \) is presented to the network, and a winner unit is declared:

\[
Winner(n) = \arg \min_{j \in \{1, \ldots, M\}} \| x(n) - w_j(n) \|
\]  

(2)

Then the prototypes of all the units are adjusted, for \( i \in \{1, \ldots, M\} \):

\[
w_i(n+1) = w_i(n) + \eta(n) \Lambda(i, Winner(n)) (x(n) - w_i(n))
\]  

(3)

where \( \eta(n) \) is a decaying learning rate and the neighborhood function \( \Lambda \) varies with the time step \( n \) and depends on a decaying neighborhood radius \( \Delta(n) \):

\[
\eta(n+1) \leq \eta(n)
\]  

(4)

\[
\Lambda(i, Winner(n)) = \exp \left( - \left( \frac{d(i, Winner(n))}{\Delta(n)} \right)^2 \right)
\]  

(5)

\[
\Delta(n+1) \leq \Delta(n)
\]  

(6)

The receptive field of unit \( i \), i.e. the region of the input space which is represented by \( i \), is defined as:

\[
F_i = \{ x(k) \mid i = Winner(k) \}
\]  

(7)

This concept will be very helpful in what follows. The ViSOM is similar to the above presented SOFM, but with a different prototype update equation instead of (3):

\[
w_i(n+1) = w_i(n) + \eta(n) \Lambda(i, Winner(n)) \left[ (x(n) - w_{Winner(n)}(n)) + (w_{Winner(n)}(n) - w_i(n)) \right]
\]

\[
\frac{\|w_{Winner(n)}(n) - w_i(n)\| - \lambda d(i, Winner(n))}{\lambda d(i, Winner(n))}
\]  

(8)

where \( \lambda > 0 \) is a prespecified resolution parameter.

B. Types of topology errors

The topographic error \( (TE, [30]) \) is a standard measure of self-organizing map quality, both for the assessment of new models and strategies \([20],[45],[1],[8],[25],[24],[27],[26],[3]\) and for the development of practical applications \([9],[22],[36]\). It is defined as the fraction of test samples whose best matching unit is not a topological neighbor of the second best matching unit:

\[
Winner(k) = \arg \min_{j \in \{1, \ldots, M\}} \| x(k) - w_j \|
\]  

(9)

\[
Second(k) = \arg \min_{j \in \{1, \ldots, M\}, j \neq Winner(k)} \| x(k) - w_j \|
\]  

(10)
\[ \text{TE} = \frac{1}{K} \sum_{k=1}^{K} I(\text{d(Winner}(k), \text{Second}(k)) = 1) \]  

(11)

where \( I \) stands for the indicator function:

\[ I : \{\text{false}, \text{true}\} \rightarrow \{0, 1\} \]  
(12)

\[ I(\text{false}) = 0 \]  
(13)

\[ I(\text{true}) = 1 \]  
(14)

Please note that \( \text{TE} \in [0, 1] \) (lower is better). A high value of \( \text{TE} \) means that the computational map is not well ordered.

Now, the problem is to capture the badly ordered maps (those with a high \( \text{TE} \)) with a definition of topology error. Some non equivalent definitions of topology errors can be found in literature. Next we present the Definition 1 by Villmann et al [62], and the Definition 2 by Su et al [57]:

**Definition 1.** A self-organizing map is not topology preserving if and only if there are two units \( i, j \) which are adjacent in the map lattice such that their receptive fields \( F_i, F_j \) are not adjacent, i.e. \( F_i, F_j \) do not share a common boundary.

**Definition 2.** A self-organizing map is not topologically ordered if and only if if there are two units \( i, j \) which are adjacent in the map lattice, and a unit \( k \) which is not adjacent to \( i \) in the map lattice, such that

\[ ||w_i - w_j|| > ||w_i - w_k|| \]  
(15)

These two definitions are not equivalent. Even worse, they incorrectly include some well ordered maps. We illustrate the situation on Figure 1, where a bidimensional input space is used, \( D = 2 \), and the map topology is rectangular. According to Definition 1 [62], Subfigure 1a is a non topology preserving map, since the two units marked in a darker tone are adjacent in the lattice, while their receptive fields are not adjacent; the receptive field boundaries are shown as thick dashed lines. However, it is a topologically ordered map according to Definition 2 [57]. In fact, Su et al state that this kind of map 'may be claimed to be a topologically ordered one', at least intuitively.

Now, Subfigure 1b depicts a map which is not topologically ordered according to Definition 2 (due to the three units marked in a darker tone), while it is a topology preserving map according to Definition 1. These two examples show that Definitions 1 and 2 are too weak, i.e. they fail to exclude all well ordered maps.
C. Self-intersections

From the preceding subsection we may conclude that an algorithm to produce good quality maps should be based on a definition of topology error which is strong enough to exclude well ordered maps such as the two examples of Figure 1. We propose a definition of this kind as follows:

**Definition 3.** A map is self-intersecting if and only if there two triples of adjacent units \(\{i, j, k\}\) and \(\{r, s, t\}\) such that the two following conditions hold:

\[
\{i, j, k\} \cap \{r, s, t\} = \emptyset \tag{16}
\]

\[
(\triangle w_i w_j w_k) \cap (\triangle w_r w_s w_t) \neq \emptyset \tag{17}
\]

where \(\triangle abc\) is the triangle defined by vertices \(a, b, c \in \mathbb{R}^D\):

\[
\triangle abc = \{(1 - u - v)a + ub + v c \mid 0 \leq u + v \leq 1\} \tag{18}
\]

Definition 3 is based on a discretization of the self-organizing map into triangles formed by adjacent units. The map is self-intersecting whenever two triangles which have no common vertices in the map lattice intersect in the input space. An example of self-intersection is shown in Figure 2; the offending connections are shown with thicker lines. Please note that neither of the two maps in Figure 1 is self-intersecting.

![Figure 2: Example of self-intersection.](image)

The connection between the notion of self-intersection (Definition 3) and the topographic error (equation 11) can be seen as follows. The points inside the triangle \(\triangle w_i w_j w_k\) should have their first and second best matching units in the set \(\{i, j, k\}\) to avoid topographic errors. Analogously, the points inside the triangle \(\triangle w_r w_s w_t\) should have their first and second best matching units in the set \(\{r, s, t\}\). But if both triangles intersect, then it is likely that some of the points on the intersection line yield topology errors, since they should have their best matching units in both sets at the same time, which is impossible since they are disjoint (16). More errors can arise from those points near to an intersection line, since the distance function is continuous.

Surface self-intersection detection and removal plays an important role in computational geometry. These techniques are restricted to surfaces embedded in the tridimensional space, \(D = 3\). There are two main kinds of strategies: algebraic methods, which need that the surface is defined by parametric equations [63], [14]; and discrete methods, where the surface is discretized into sets of small planar surfaces, usually triangles [51], [5]. In our case only the latter are relevant, since a self-organizing map is defined by a discrete lattice.

III. SELF-INTERSECTION AVOIDANCE

This section contains our approach to use the concept of self-intersection to improve the quality of a self-organizing map. We propose to rollback the learning steps which lead to an undesirable state of the map. Next the undesirability condition is defined, and then an explanation is given on how to check for it.

In order to take advantage of discrete methods for detecting triangle mesh self-intersections, it is necessary to generate a triangle set from a self-organizing map. Definition 3 includes all possible triangles that can be formed from adjacent units. But
it is not necessary to include all of them to discretize the self-organizing map surface. Consequently, we restrict the checks to the triangles depicted in Subfigure 3a. Moreover, for maps with sizes larger than $8 \times 8$ units, it has been found that the checks can be carried out at a larger scale to save computation time. Hence, for these large maps the checks are done by $2 \times 2$ unit blocks, as shown in Subfigure 3b. A map is in an undesirable state if and only if there is an intersection between two triangles of the above specified triangle set which do not share any common vertex.

The effect of rolling back the learning whenever an intersection is detected depends heavily on the dimension of the space where the intersections are computed. We have found that for a bidimensional space the map can not progress since the undesirable states are too frequent. On the other hand, if the dimension of the space is larger than 3, then the undesirable states are very infrequent, so there is no impact in the learning process and the map quality is not improved. This leaves us with a three dimensional space, where the frequency of undesirable states is moderate. In order to accommodate for input samples of dimension $D > 3$, we compute the intersections in the three dimensional principal subspace obtained by a global Principal Components Analysis (PCA) on the input dataset. This implies that our approach can not be applied for input dimension $D = 2$. Please note that the dimensionality reduction to three dimensions is done inside the self-intersection check subroutine only, while the prototype update runs just as in the standard SOFM, with the full dimension original input data.

Now we see how to check for intersections. Here we follow the fast triangle-triangle intersection test proposed in [46]. Let us consider a first triangle defined by the triple of points $p_0$, $p_1$ and $p_2$; and a second triangle defined by $q_0$, $q_1$ and $q_2$. First we compute the equation of the plane $\pi_2$ where the second triangle lies:

$$\pi_2 : n_2^T x + \delta_2 = 0$$  \hspace{1cm} (19)

$$n_2 = (q_1 - q_0) \times (q_2 - q_0)$$  \hspace{1cm} (20)

$$\delta_2 = -n_2^T q_0$$  \hspace{1cm} (21)

Then the signed distances from the vertices of the first triangle to $\pi_2$ (multiplied by a constant $n_2^T n_2$) are obtained by inserting the vertices into (19). For $i \in \{0, 1, 2\}$ we have:

$$\delta_{p_i} = n_2^T p_i + \delta_2$$  \hspace{1cm} (22)

Now, if all $\delta_{p_i} \neq 0$ and all have the same sign, then the first triangle lies on one side of $\pi_2$ and the triangles can not overlap. The same is done for the second triangle and the plane $\pi_1$ where the first triangle lies. If both tests are negative, then there is a line of direction $n_1 \times n_2$ which meets both triangles. The intersections of the triangles with this line form two intervals on it, and the triangles intersect if these intervals overlap. There is also a special test for coplanar triangles; the reader is kindly directed to the original paper [46].
Finally we are ready to present our map quality improvement method. The learning algorithm of the original self-organizing map model (either Kohonen’s SOFM or ViSOM) is modified in the following way:

- We initialize the map by setting up the prototypes over a rectangular grid on the two-dimensional principal subspace obtained by a global Principal Components Analysis (PCA) on the input dataset. This initial state can not be undesirable.
- Every $N_{\text{check}}$ learning steps, if the current map state is undesirable, then we rollback to the last known desirable state.

This modification can be seen as an algorithm to solve the following constrained optimization problem:

$$\min \mathcal{E}(M) \quad \text{subject to } M \in \mathcal{D}$$

where $M$ is the self-organizing map, $\mathcal{D}$ is the set of desirable map states and $\mathcal{E}$ is the energy function of $M$ [34], [19]:

$$\mathcal{E}(M) = \sum_{i=1}^{M} \int_{P} \sum_{j=1}^{M} \Lambda(i,j) \|x - w_j\|^2 p(x) \, dx$$

IV. EXPERIMENTS

In order to assess the ideas we have presented, we have designed two kinds of experiments. First of all, we have carried out a quantitative evaluation over synthetic data with respect to quantization error and map quality performance measures (Subsection IV-A). This way we check the behavior of our approach. Then we have chosen three case studies from different application domains for a qualitative evaluation (Subsection IV-B). These are examples of what the practitioner would find in a practical application.

The quantization error, also known as mean squared error ($MSE$), is the most acknowledged way to measure the vector quantization performance of a SOM [20], [66], [3], [13]:

$$MSE = \frac{1}{K} \sum_{k=1}^{K} \min_{j \in \{1, \ldots, M\}} \|x(k) - w_j\|^2$$

where $K$ is the number of test samples. Lower is better, and its value is expected to decrease as the number of units increases, since there are more prototypes to represent the input data. On the other hand, the map quality has been measured by the already mentioned topographic error $TE$ (11).

The parameter selection strategy for the self-organizing maps has been the same in all the experiments. Let $N$ be the overall number of time steps of the training process. We have divided the training process into an ordering phase and a convergence phase with the same number of time steps, i.e. $N/2$ time steps each. During the ordering phase, the learning rate and the neighborhood radius experience a linear decay:

$$\eta(n) = \eta_0 \left(1 - \frac{n}{N}\right)$$

$$\Delta(n) = \Delta_0 \left(1 - \frac{n - 1}{N}\right)$$

During the convergence phase, constant values have been used to carry out the fine tuning of the maps: $\eta(n) = \eta_c$, $\Delta(n) = \Delta_c$. Hence, the set of parameters to be chosen is: $\eta_0, \Delta_0, \eta_c, \Delta_c$, and the resolution parameter $\lambda$ for ViSOM (8). We have considered an objective function $\varphi$ to be minimized which takes into account both the vector quantization and the map quality:

$$\varphi = \frac{MSE}{MSE_0} + TE$$

where $MSE_0$ is the mean squared error for a single prototype placed at the mean of the dataset. The Nelder-Mead optimization method [48] has been used, which is quite robust with respect to noise in the objective function. To this end we split the available set of samples into a training set (90% of the data) and a validation set (the remaining 10%). The the map is trained with the training set and then $\varphi$ is evaluated over the validation set.

As our approach includes extra computation with respect to the original Kohonen’s SOFM and ViSOM, namely the self-intersection check, it is also important to measure the increase of the computation time. The CPU time results are always attached to a specific hardware and software, but we include them to convey a general idea of the computational load associated with the method. We have implemented our proposal in Matlab, and it has been run on a single core of a 3GHz CPU with 64 bit architecture and 2GB of RAM. Our approach is marked in the plots as ‘NSI’, which stands for ‘non self-intersecting’; this leaves four competing approaches: SOFM, NSI-SOFM, ViSOM and NSI-ViSOM.

The source code and demos of our proposal are available at http://www.lcc.uma.es/%7Eezeqlr/nsisom/nsisom.html
A. Synthetic data

We have chosen as input the uniform distribution over the unit hypercube for input dimensions \( D = 3, \ldots, 20 \). Please note that \( D = 3 \) is the minimum input dimension for our approach. Self-organizing maps have been tested with sizes \( 4 \times 4, 5 \times 5, 6 \times 6, 7 \times 7 \) and \( 8 \times 8 \); the number of time steps has been set to \( N = 100,000 \). For every input dimension and map size a 10-fold cross validation has been carried out over 100,000 randomly drawn samples to yield the mean and the standard deviation of the three measures \( TE \) (equation 11), \( MSE \) (equation 25) and CPU time. The results are shown in Figures 4, 5, and 6, respectively. The standard deviations are shown as error bars.

A study of the statistical significance of the differences among the methods has been carried out. The nonparametric Friedman test with the corresponding post-hoc Dunn test have been selected, which are robust for multi-way comparisons [11]. An asterisk is shown in the figures whenever the difference among the non self-intersecting (NSI) and the standard versions of both SOFM and ViSOM are significant with 95% confidence (\( \alpha = 0.05 \)).

![Figure 4: Synthetic data topographic error results for different map sizes: (a) \( 4 \times 4 \), (b) \( 5 \times 5 \), (c) \( 6 \times 6 \), (d) \( 7 \times 7 \), (e) \( 8 \times 8 \). An asterisk is shown whenever the difference among the NSI and the standard versions are statistically significant.](image)

The map quality results reveal that our approach is better, except for the \( 4 \times 4 \) and \( 5 \times 5 \) map sizes where the difference is negligible. On the other hand, the vector quantization performance features a small advantage for the original models which does not scale with the input dimension. This means that there is a price to pay to enforce a good ordering of the map, since the units are less free to move through the input space. The slight decrease of the MSE with increasing SOM sizes for all tested methods is a consequence of the curse of dimensionality: the mean distance to the closest unit does not decrease much when we add more units.

The greater picture is that our proposal differs little from the standard model for small map sizes both in quantization error and topographic error. This is because in a small map there is less chance to run into a self-intersecting map state. Conversely, large maps have a more complex structure which is prone to self-intersection because it can bend itself more easily. In these cases, the NSI versions yield better results.
An asterisk is shown whenever the difference among the NSI and the standard versions are statistically significant.

cases our approach attains larger differences.

Finally, the computational overhead of our method is not statistically significant in any case when compared to the original map models. Like before, the difference does not scale with the input dimension. As we will see in Subsection IV-B, the difference is also negligible for high dimensional inputs. Also, it is worthwhile to note that ViSOM is slower due to its more complex update equation (8).

B. Real data

Here we aim to show the relevance to practitioners of the improvement in the map quality that have been demonstrated in Subsection IV-A. To this end three different application domains have been selected where self-organizing maps are typically used, namely data visualization, computer vision and pattern recognition.

We have employed three kinds of graphical representations of the trained maps:

- The first of them is the unified distance matrix, or U-matrix [59], [65], [10], [40]. It depicts the distances between every pair of units with different tones, so that the units which are too far from its topological neighbors can be easily spotted.
- The second one is the silhouette plot, which is specifically designed to assess the quality of the clusters [21], [28], [64]. Here we have considered clusters of $4 \times 4$ units to produce readable plots. Let $\sigma_k$ be the average distance from sample $x(k)$ to the other points in its own cluster, and $\sigma_{j,k}$ the average distance from $x(k)$ to points in another cluster $j$. The silhouette value for a sample $SV(x(k)) \in [-1,1]$ (higher is better) and the average silhouette value $MSV$ have been computed, where:

$$SV(x(k)) = \frac{-\sigma_k + \min_j \sigma_{j,k}}{\max \{\sigma_k, \min_j \sigma_{j,k}\}}$$

- The third representation is specifically designed to highlight the differences of our approach with respect to the original algorithms. We project the prototypes and some randomly chosen input samples on the three dimensional principal subspace obtained by a global Principal Components Analysis (PCA) on the input dataset. Then we generate a 3D plot where the samples are shown as dots and the topological connections between the units as lines. This way we can check visually whether the maps are topologically ordered no matter how large is the input space dimension $D$. To further enhance the
visualization, we plot the samples corresponding to topographic errors in a darker tone, so that we can spot the most difficult parts of the dataset.

Next the qualitative results for the three selected case studies are presented, and then the quantitative results corresponding to all of them are presented.

1) Data visualization: One of the most common usages of self-organizing maps is for multidimensional data visualization [29], [20], [8], [15], [16], [17]. We have chosen a set of global climate data from the NASA Earth Observatory [47]. In particular, we have obtained images corresponding to February 2011 for the following nine climate parameters: aerosol optical depth, aerosol size, chlorophyll concentration, cloud fraction, land surface temperature, vegetation, water vapor, carbon monoxide, and net radiation. Please refer to the source of the dataset for a description of these parameters. The images are 3600×1800 pixels in size, and represent the entire surface of the Earth; they have been reduced to 900×450 pixels to yield a smaller dataset. The pixels have a precision of 8 bits, with values in the range [0, 1]. Consequently, the input dataset for the self-organizing maps has 405,000 samples (one per pixel), each with nine components (one per climate parameter). That is, the input dimensionality is $D = 9$. A map with $8 \times 8$ units has been trained for each of the four proposals for $N = 100,000$ time steps.

The result of this training is that every unit of a map represents a set of points on the surface of the Earth with similar climate parameters (Figure 7). We have assigned a different color to each unit, so that neighboring units have similar colors (Subfigure 7e). Then we have generated four Earth images (one per competing approach) where every pixel is plotted in the color of the receptive field it belongs to; this amounts to an unsupervised segmentation of Earth surface. Consequently, these images should show Earth regions with similar climate parameters painted with similar colors, as learned by the associated self-organizing map. However, it can be seen in Subfigure 7a that the original SOFM assigns similar colors to Brazil and...
Figure 7: Earth climate segmentation. (a) SOFM, (b) NSI-SOFM, (c) ViSOM, (d) NSI-ViSOM, (e) color associated to each unit.

Antarctica. This is a topographic failure corresponding to neighbor units which represent distant input samples, and would lead
to the erroneous interpretation that these two regions have a similar climate. Also, the original ViSOM assigns different colors to North Pole and Antarctica, and large color changes are observed in North Greenland (Subfigure 7c). These kind of failures are not observed in our approach (Subfigures 7b and 7d). The principal subspace plots (Figure 8) indicate that the original models yield twisted maps, which account for the topographic errors described before.

The conclusions outlined above are confirmed by examination of the silhouette plots (Figure 9) and the U-matrices (Figure 10). The silhouette plots reveal that our approach generates good quality clusters. Moreover, the U-matrix of our proposal is
homogeneous and the inter-unit distances are low. Meanwhile, the U-matrix of the original algorithm shows higher distances between neighbors, which means that the mapping is not smooth.

2) Computer vision: Computer vision has taken advantage of self-organizing models for various purposes [6], [42], [49], [52], [55]. Here we are interested in their capability to model a manifold of images [60]. We aim to produce maps which represent the most common contents of the frames contained in a video, i.e. a model of normal behavior which could be used to build an anomaly detection system [4], [54] or a background modeling procedure [70], [38], [23], [56]. We have chosen the Fountain video sequence, which is freely available [39]. This sequence depicts a fountain with flowing water, and some pedestrians passing in front of it; at a certain point a pedestrian stops for a while. There are 523 frames (input samples), each of size $160 \times 128$ pixels. The frames have been converted to grayscale as a preprocessing stage. The pixels have a precision of 8 bits, with values in the range $[0, 255]$. With these arrangements the dimension of the input samples is $D = 20480$. A map with $8 \times 8$ units has been trained for each of the competing approaches for $N = 10,000$ time steps; in this case less steps are needed because there are few samples.

The topological ordering of the prototypes learned by the maps can be seen on Figure 11. The original SOFM algorithm produces a rather unordered final map state, while the original ViSOM performs better. On the other hand, NSI-SOFM and NSI-ViSOM unfold correctly, yielding good unit arrangements.

The silhouette plots (Figure 12) show that the differences between the map unfoldings considered before correspond to a worse clustering in the case of the original algorithms. The NSI-ViSOM algorithm clearly outperforms all the others in this case. An ordered map translates to small inter unit distances (Figure 13), and vice versa.

3) Pattern recognition: Our last set of experiments is devoted to the well known problem of handwritten digit recognition, which has also been managed with the help of self-organizing models [67], [7], [50], [43], [69]. We have selected a standard benchmark, namely the MNIST Handwritten Digit Database [35]. This database contains 60,000 grayscale images of handwritten digits, each of size $28 \times 28$ pixels, so that the input dimension is $D = 784$. The pixels have 8 bit precision, with values in the
range $[0,1]$. A map with $8 \times 8$ units has been trained for each of the two proposals. This time we have used $N = 200,000$ time steps, since the number of input samples is higher than in the previous experiments.

Smoorher transitions among digits reveal that our maps are smoother (Figure 14). For this dataset, the NSI-SOFM algorithm yields the best cluster quality, as seen in the silhouette plots (Figure 15), which demonstrate that NSI-SOFM outputs clusters of well balanced size and higher inter-cluster separation. These conclusions are further confirmed by the U-matrices (Figure
Figure 13: Fountain video U-matrices. (a) SOFM, (b) NSI-SOFM, (c) ViSOM, (d) NSI-ViSOM, (e) inter-unit distance scale.

Figure 14: MNIST handwritten digit prototypes. (a) SOFM, (b) NSI-SOFM, (c) ViSOM, (d) NSI-ViSOM.
Figure 15: MNIST handwritten digit silhouette plots. (a) SOFM, $MSV = 0.0546$; (b) NSI-SOFM, $MSV = 0.0629$; (c) ViSOM, $MSV = 0.0583$; (d) NSI-ViSOM, $MSV = 0.0443$.

16), where it is shown that our proposals produce substantially smaller inter-unit distances.

4) Quantitative results: The quantitative performance measures for the three case studies we have just examined can be found in Tables I, II and III. They have been obtained by 10-fold cross validation over the available set of samples in each case. As in Subsection IV-A, the mean squared errors are pretty similar, while the topographic errors of our approach are clearly lower than those of the original ones. In addition to this, the CPU times are almost equal. This implies that the computational overhead of our method is not significant due to the fast self-intersection check.

<table>
<thead>
<tr>
<th></th>
<th>Earth</th>
<th>Fountain</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOFM</td>
<td>0.3504 (0.0246)</td>
<td>0.1918 (0.0596)</td>
<td>0.3841 (0.0190)</td>
</tr>
<tr>
<td>NSI-SOFM</td>
<td>0.2485 (0.0290)</td>
<td>0.1156 (0.0447)</td>
<td>0.2875 (0.0507)</td>
</tr>
<tr>
<td>ViSOM</td>
<td>0.3168 (0.0198)</td>
<td>0.1752 (0.0574)</td>
<td>0.3844 (0.0190)</td>
</tr>
<tr>
<td>NSI-ViSOM</td>
<td>0.2193 (0.0325)</td>
<td>0.1423 (0.0488)</td>
<td>0.2691 (0.0380)</td>
</tr>
</tbody>
</table>

Table I: Topographic error for the three case studies.

<table>
<thead>
<tr>
<th></th>
<th>Earth</th>
<th>Fountain</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOFM</td>
<td>0.0299 (0.0005)</td>
<td>1.0234 (0.1487)</td>
<td>28.3837 (0.1141)</td>
</tr>
<tr>
<td>NSI-SOFM</td>
<td>0.0426 (0.0029)</td>
<td>1.0821 (0.1373)</td>
<td>30.6729 (0.8643)</td>
</tr>
<tr>
<td>ViSOM</td>
<td>0.0296 (0.0006)</td>
<td>1.0198 (0.1427)</td>
<td>28.8290 (0.1154)</td>
</tr>
<tr>
<td>NSI-ViSOM</td>
<td>0.0399 (0.0036)</td>
<td>1.0516 (0.1424)</td>
<td>30.7865 (0.6609)</td>
</tr>
</tbody>
</table>

Table II: Mean squared error for the three case studies.

V. DISCUSSION

In this section some important features of our proposal are studied:

- Most of the work done on self-organizing map quality has been oriented towards the evaluation of already trained networks [8]. In some cases these evaluations have been used to find out the intrinsic dimensionality of the input dataset [2]. Here
Figure 16: MNIST handwritten digit U-matrices. (a) SOFM, (b) NSI-SOFM, (c) ViSOM, (d) NSI-ViSOM, (e) inter-unit distance scale.

<table>
<thead>
<tr>
<th></th>
<th>Earth</th>
<th>Fountain</th>
<th>MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOFM</td>
<td>19.7206 (0.0395)</td>
<td>1367.09 (31.84)</td>
<td>1761.02 (82.98)</td>
</tr>
<tr>
<td>NSI-SOFM</td>
<td>21.4573 (0.0735)</td>
<td>1389.63 (27.70)</td>
<td>1847.15 (151.11)</td>
</tr>
<tr>
<td>ViSOM</td>
<td>37.8440 (0.0199)</td>
<td>2404.11 (12.01)</td>
<td>3482.40 (101.99)</td>
</tr>
<tr>
<td>NSI-ViSOM</td>
<td>39.3348 (0.0890)</td>
<td>2526.87 (32.94)</td>
<td>3588.80 (43.93)</td>
</tr>
</tbody>
</table>

Table III: CPU time (in seconds) for the three case studies.

Our goal is different, since we use topographic quality concepts to establish a modified learning process which produces better maps. As noted in [57], some quality measures such as the topographic product involve a heavy computational load, which prevents their use in the learning algorithm.

- Our proposal can be applied to many self-organizing map models. It is based on the geometrical properties of the lattice formed by the prototypes. Consequently, it is only needed that the model at hand defines a prototype \( w_i \in \mathbb{R}^D \) for each unit \( i \). Even for hierarchical self-organizing map models [53], it can be applied separately to each map of the hierarchy.

- The practitioners can make use of this method without having to learn a new self-organizing model, since our approach modifies the training algorithm, but not the model structure. That is, people who are not aware of the details of the learning algorithms of self-organizing maps can take advantage of this approach, since the interpretation of the maps does not change.

- This approach should be used with care for classification applications, which are supervised tasks. If the classification performance is the only goal, then it is possible that a heavily twisted map attains the best results because it gets as close as possible to the training data corresponding to a certain class. In these cases, an improvement of the map quality in terms of topographic error might decrease the classification accuracy. Conversely, the silhouette values obtained for the real datasets (Subsection IV-B) suggest that our approach attains better results for unsupervised clustering. This is because self-intersection avoidance tends to exclude map states where two distant sections of the map are trying to model the same cluster of data.
VI. Conclusions

We have presented a new method to improve the quality of self-organizing maps. It relies on the avoidance of undesirable map states, which are defined as a superset of self-intersecting states. The maps trained with this approach exhibit less topology errors at the expense of a larger quantization error, as shown in the experiments. The practical use of the method has been illustrated with three real case studies. Furthermore, the procedure can be easily extended to many self-organizing neural networks, and it does not change the structure of the original model.

ACKNOWLEDGMENTS

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under grant TIN2011-24141, project name Detection of anomalous activities in video sequences by self-organizing neural systems, and the Autonomous Government of Andalusia (Spain) under project P12-TIC-6213, project name Development of Self-Organizing Neural Networks for Information Technologies. The author thankfully acknowledges the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Malaga.

REFERENCES


Ezequiel López-Rubio (born 1976) received his MSc and PhD (honors) degrees in Computer Engineering from the University of Málaga, Spain, in 1999 and 2002, respectively. He joined the Department of Computer Languages and Computer Science, University of Málaga, in 2000, where he is currently an Associate Professor of Computer Science and Artificial Intelligence. His technical interests are in unsupervised learning, pattern recognition and image processing.