On cluster analysis via neuron proximity in monitored self-organizing maps

Susana Vegas-Azcárate and Jorge Muruzábal

Statistics and Decision Sciences Group
University Rey Juan Carlos, 28933 Móstoles, Spain
{susana.vegas,jorge.muruzabal}@urjc.es

Abstract. A potential application of self-organizing or topographic maps is clustering and visualization of high-dimensional data. It is well-known that an appropriate choice of the degree of smoothness in topographic maps is crucial for obtaining sensible results. Indeed, experimental evidence suggests that suitably monitored topographic maps should be preferred as they lead to more accurate performance. This paper reconsiders the basic toolkit for cluster analysis —based on the relative distance from each pointer to its immediate neighbours on the network— from this monitoring perspective. It is shown that the idea works nicely, that is, much useful information can be encoded and recovered via the trained map alone (ignoring any possible density estimate available). Moreover, the fact that a topographic map is not restricted to metric vector spaces makes this learning structure a perfect tool to deal with biological data, such as DNA or protein sequences of living organisms, for which only a similarity measure is readily available.

Key words: Topographic maps, monitoring, neuron interdistances.

1 Introduction

We are living the blast of the silicon-based biology era, an era in which investigation of complete genomes is viable for the first time. We need computer-based technologies to cope with the vast quantities of data generated by genome projects, seeking not only increased facilities in data storage and access, but also assistance in computational manipulation and post-processing. Without methods that help us to analyze this deluge of data, the information it contains becomes useless.

The Self-Organizing Map (SOM) [1] is a popular neural network model for unsupervised learning that tries to 'imitate' the *self-organization* process taking place in the sensory cortex of the human brain (by which neighboring neurons will typically be activated by similar stimuli). This model develops a mapping from a d-dimensional input space, into an equal or lower-dimensional discrete lattice with regular, fixed topology. Thanks to a simple competitive learning process — whereby only weights connected to the winner (or best matching

unit) and its neighbours are updated—, the SOM structure is often organized (topologically ordered).

We are interested in applying the SOM neural structure for sound clustering of biological signals. We have a particular application in mind, namely, clustering proteins. This problem has some tradition in the literature [2, 3], and it prompts a number of interesting issues, beginning with the issue of the 'vectorial vs. nonvectorial' data representation. However, in this paper we consider only the vectorial case and pursue some foundational research (i) focusing on the use of interneuron proximity information *alone*, and (ii) emphasizing the need to monitor the training process. While both theory and applications have developed substantially in the SOM literature, there is probably not a wide awareness yet (among practitioners) of the technical requirements for the extracted maps, nor there seems to be a wide consensus on how to best monitor, train or even analyze the SOM structure.

For clustering purposes, a good density estimate of the sampling distribution can of course be very valuable, and certain kernel-based learning algorithms are naturally suited to yield such estimates. While conventional kernel-based estimates [4, 5] seem to provide generally good results, drawbacks in higher dimensions should still occur in the multivariate case [6]. Recent approaches to SOM training usually incorporate some statistical machinery yielding richer models and more principled fitting algorithms (the standard framework lacks a statistical model for the data and thus provides no density estimate).

As regards monitoring, we have introduced an early-stopping criterion called UDL (for uniform data load) and we have shown that it provides sensible density estimates in a wide array of cases [7]. Here we show that the UDL criterion is also useful for the usual basic proximity summaries (available in all training algorithms). To this end, four algorithms are tested. Specifically, the batch version (SOM-B) [1] and a convex adjustment (SOM-Cx) [8] of the standard SOM algorithm are compared to two kernel-based learning rules: the generative topographic mapping (GTM) [9] and the kernel-based maximum entropy learning rule (kMER) [10]. The latter three tend to achieve the 'equiprobabilistic state' that motivates the UDL criterion [7]; it appears unlikely that SOM-B can achieve this state, but it is still monitored in the same way for the sake of reference.

The organization is as follows. Section 2 briefly describes the four topographic map formation algorithms considered in the paper. Section 3 spells out the particular training and testing strategies examined in the experiments reported in section 4. Some conclusions are drawn in section 5.

2 SOM training algorithms

Before we actually describe the details of the algorithms, it is appropriate to begin with a bit of background for the work presented here. As often noted, the quality of the SOM fit will be assessed in the first place by the extent to which the organization or topology preservation property holds. A substantial amount of research has been devoted to the formalization and quantification of

this idea. Proposed approaches vary from the early measures based entirely on the map pointers [11] to increasingly sophisticated variants that also incorporate some aspects of the data into the analysis [12–15]. While much progress has been done in the area, there is no universally accepted methodology for practitioners to follow. Many proposed measures are not easy to interpret, and the available implementations are scarce. At any rate, empirical confirmation is needed in each case.

Determination of the most sensible criterion is also complicated by the nature of the training algorithms. Most importantly, algorithms differ in their *magnification factors*, that is, in the ability to reflect exactly the true density generating the data. When the match between pointer density and data density is exact (at least asymptotically) we talk of equiprobabilistic maps [16]. These have been argued to overcome the output unit underutilization problem found in the standard SOM training algorithm. But even among theoretically equiprobabilistic algorithms there are, as we shall see, substantial differences in practical behaviour.

Finally, training algorithms vary also in the amount of modelling machinery involved. As noted above, the traditional SOM structure lacks a statistical model for the data, whereas modern training algorithms like GTM and kMER provide an explicit density estimate that can be very useful for clustering purposes. These estimates enhance the framework and raise questions about the preferred training strategy.

Denote the SOM weight or pointer vectors as $w_i \in \mathrm{IR}^d, i = 1, ..., N$, and the data as $v_m \in \mathrm{IR}^d, m = 1, ..., M$. In this paper we consider 2D SOMs only; besides, we restrict consideration to squared maps equipped with the standard topology.

2.1 SOM-batch (SOM-B)

The batch version of Kohonen's SOM training algorithm (SOM-B) is defined by the recursive update [1]

$$w_i = \frac{\sum_{m=1}^{M} v_m H_{i_m^* i}}{\sum_{m=1}^{M} H_{i_m^* i}},$$

where H is the neighborhood function, typically chosen with a Gaussian shape and a monotonically decreasing range, and $i_m^* = \arg\min_i \{\|v_m - w_i\|\}$ is the best matching unit for the input data vector v_m. Since SOM-B contains no learning rate parameter, no convergence problems arise, and more stable asymptotic values for the weights w_i's are obtained [1].

2.2 Convex adjustment (SOM-Cx)

A convex adjustment for the original SOM algorithm has been studied by Zheng and Greenleaf [8]. They actually present two nonlinear models of weight adjustments. One of them uses a *convex* transformation to adjust weights. This is seen to provide more efficient data representation for vector quantization, whereas

the convergence rate is comparable to that of the linear model. Specifically, the standard competitive learning rule

$$\Delta w_i = \eta H_{i^*_m i}(v_m - w_i),$$

becomes

$$\Delta w_i = \eta H_{i^*_m i}(v_m - w_i)^{\frac{1}{\kappa}},$$

where κ is a positive odd integer [8] and η represents the learning rate (which can also be a monotonically decreasing function of time [1]).

2.3 Generative Topographic Mapping (GTM)

GTM [9] defines a non-linear mapping $y(x, W)$ from an L-dimensional latent space to a d-dimensional data space, where $L < d$ (and typically equals 2). By suitably constraining the model to a lattice in latent space, a posterior distribution over the latent grid is readily obtained using Bayes' theorem for each data point. More specifically, GTM training is based on a standard EM procedure aimed at the standard Gaussian-mixture log-likelihood [9]

$$\log \ell = \sum_{m=1}^{M} \log \left\{ \sum_{i=1}^{N} p(v_m|i) P(x_i) \right\}, \tag{1}$$

where $P(x_i)$ is the prior mass at each point in the latent grid and $p(\cdot|i)$ is the Gaussian density centered at $y_i = y(x_i, W)$ (equal, of course, to our more common w_i) and spherical covariance with common variance $\sigma_i^2 = \beta^{-1}$. A generalized linear regression model is typically chosen for the embedding map, namely $y(x, W) = W\Phi(x)$, where $\Phi = \Phi(x)$ is a matrix containing the scores by B fixed basis functions and W is a free matrix to be optimized together with β.

Note that the (optimized) quantity in Eq. 1 provides a standard measure on which a GTM model can be compared to other generative models. Note also that the topology-preserving nature of the GTM mapping is an automatic consequence of the choice of a continuous function $y(x, W)$ [9]. Basis functions parameters explicitly govern the smoothness of the fitted manifold.

2.4 Kernel-based Maximum Entropy learning Rule (kMER)

kMER [10] was introduced as an unsupervised competitive learning rule for non-parametric density estimation, whose main purpose is to obtain equiprobabilistic topographic maps on regular, fixed-topology lattices. Here, the receptive fields of neurons are (overlapping) radially symmetric kernels, whose radii are adapted to the local input density together with the weight vectors that define the kernel centroids. A neuron w_i is 'activated' by an input data v_m if it is contained within the hypersphere S_i centered at w_i and with radius σ_i. Since hyperspheres are allowed to overlap, several neurons can be active for a given input vector. An

online together with a batch version of kMER are developed in [10]. We focus here on the online version of kMER, that is,

$$\Delta w_i = \eta \sum_{j=1}^{N} H_{ji} \Xi_j(v) Sgn(v_m - w_i),$$

where $Sgn(\cdot)$ is the sign function taken componentwise, Ξ is a fuzzy code membership function and H is the time-decreasing neighborhood function. Note that, unlike GTM, kMER derives a different standard deviation σ_i for each mixture Gaussian component. Specifically, the kernel radii σ_i are adjusted so as to verify, at convergence, that the probability of each neuron i to be active is given by a fixed scale factor ρ (which controls the degree of overlap between receptive fields).

It can be seen that the receptive field weight centers and its radii are adapted to achieve a topographic map maximizing the unconditional information-theoretic entropy [16]. Furthermore, the density estimate output by kMER can be written in terms of a mixture distribution where the kernel functions represent the component Gaussian densities with equal prior probabilities, providing an heteroscedastic, homogeneous mixture density model [16] whose log-likelihood function can be computed just like in the GTM case (see Eq. 1 above).

3 Analysis

Once we have trained the SOM, a number of summaries of its structure are routinely extracted and analyzed. In particular, here we consider *Sammon's projections*, *median interneuron distances*, and *dataloads*. We now review each of this basic tools in turn.

To visualize high-dimensional SOM structures, use of Sammon's projection is customary. Sammon's map provides a useful global image while estimating all pairwise Euclidean distances among SOM pointers and projecting them directly onto 2D space. Thus, since pointer concentrations in data space will tend to be maintained in the projected image, we can proceed to identify high-density regions directly on the projected SOM. Furthermore, by displaying the set of projections together with the connections between immediate neighbours, the degree of self-organization in the underlying SOM structure can be assessed intuitively in terms of the amount of overcrossing connections.

Interneuron distance or proximity information has also been traditionally used for cluster detection in the SOM literature. Inspection of pointer interdistances was pioneered by Ultsch, who defined the *unified-matrix* (U-matrix) to visualize Euclidean distances between neuron weights in Kohonen's SOM. Here we consider the similar *median interneuron distance* (MID) matrix. Each MID entry is the median of the Euclidean distances between the corresponding pointer and all pointers belonging to a star-shaped, fixed-radius neighborhood containing typically eight units. The median can be seen as a conservative choice; more radical options based on extremes can also be implemented. To facilitate the

visualization of pointer concentrations, a linear transformation onto a 256-tone gray scale is standard (the interpretation here is that the lower the value, the darker the cell).

On the other hand, the number of data vectors projecting onto (won by) each unit, namely the neuron *dataload*, is the main quantity of interest for UDL monitoring purposes. Again, to easily visualize the dataload distribution over the map, a similar gray image is computed, namely, the DL-matrix (note that, in this case, darker means higher). The main idea in UDL is that, in the truly equiprobabilistic case, each neuron would cover about the same proportion of data, that is, a (nearly) uniform DL-matrix should be obtained. Hence, training is stopped as soon as the first signs of having reached this state are noticed [7]. Note that we use the UDL stopping policy as a heuristic for the optimal value of the final adaptation radius in SOM-B and SOM-Cx.

The training strategy for cluster analysis is thus formally described as follows. First train the SOM network until a (nearly) uniform DL-matrix is obtained and Sammom's projection shows a good level of organization. Compute the MID and DL matrices associated to this map. We stress that we do not use the maps obtained by training all the way (which yield much worse results).

Now, a cluster detection strategy based on MID would first isolate all local maxima on the MID surface and identify each such a mode with a specific cluster in the data. We can also consider a minimum Euclidean labelling scheme, in which each neuron is marked with the label that occurs most within its activation region. We denote this as the *labels* matrix below.

4 Experimental work

We now summarize our main experimental results. We first analyze a trimodal 2D data set with two of the modes close enough to illustrate the finer detail in our algorithms. Next we examine a mixture of gaussians with clusters relatively apart from each other. Finally, we consider a real data set with ten clusters in high-dimensional input space.

4.1 Three modes in 2D space (3M-2D)

Figure 1 shows good organization and suitable DL matrices in all cases (albeit more uniform DL matrices can be seen in the case of GTM and kMER). As a result, the three clusters are correctly identified via MID analysis, yet we note that kMER and SOM-Cx provide the cleanest assignments.

4.2 Seven modes in 5D space (7M-5D)

Data are generated in a two-step process. We first sample the locations of the cluster centroids, then sample each cluster in term. All Gaussians are spherical, and all clusters have the same size. The standard deviation of the centroid distribution is much larger than that of the data (clusters are well separated).

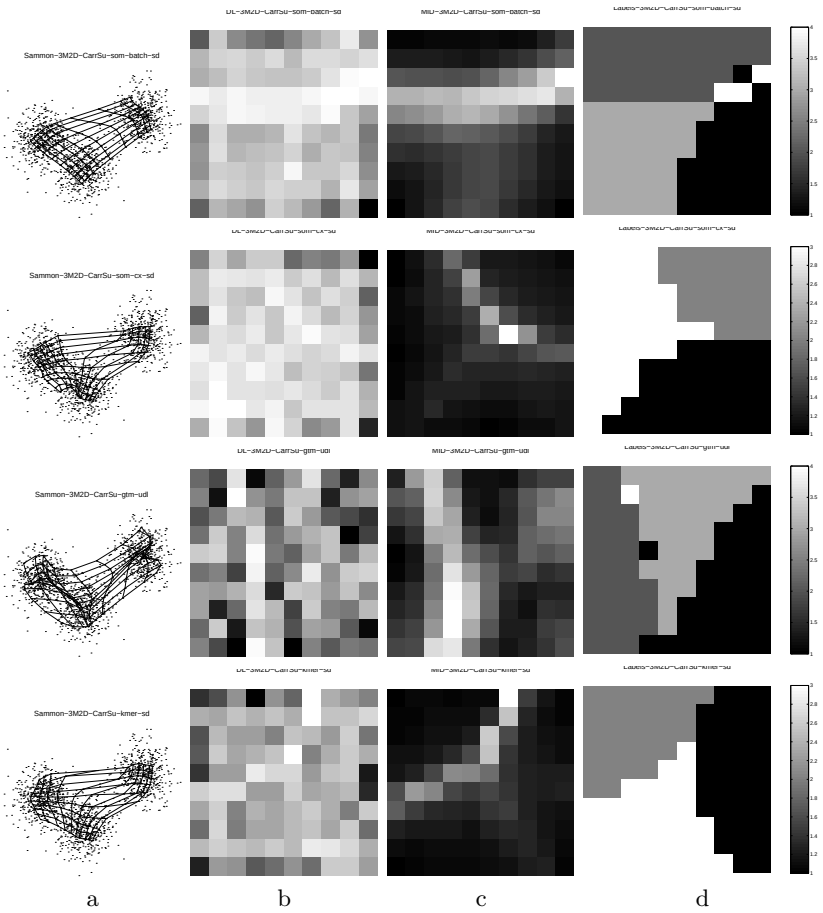

Fig. 1. Performance by SOM-B (top), SOM-Cx (middle-up), GTM (middle-down) and kMER (bottom) on 3M-2D data set: (a) trained maps with data set highlighted; (b) DL matrices; (c) MID matrices; (d) Labels matrices

Figure 2 shows generally nice behaviour except in the GTM case, where a relatively high number of dead units is unexpectedly observed. In all other cases, the seven clusters are exhibited very clearly by the MID matrices.

4.3 Real-world example

We now take up the Multiple Features database from the well-known UCI repository, which we call the Mfeat data. Here $d = 649$ and there are $M = 2,000$ training vectors available. As Figure 3 shows, interestingly organized maps obtain in all cases. These maps involve rather uniform DL matrices and result in pretty good cluster recognition, see also [7]. Here we see GTM ranking worst in terms of cluster separation.

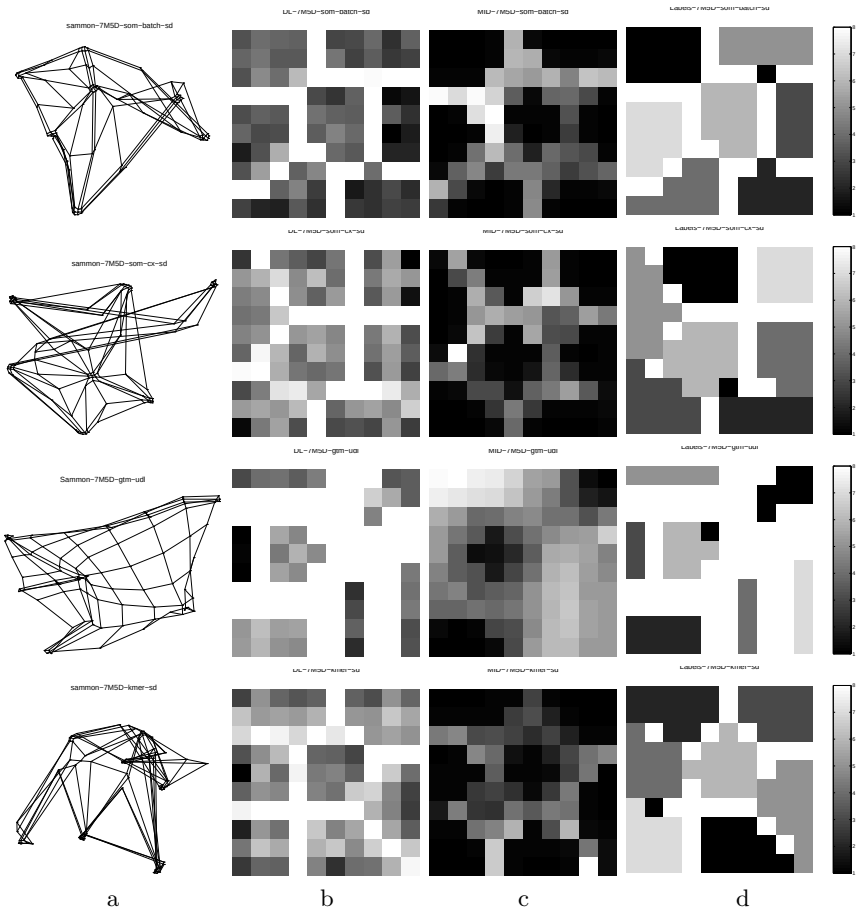

Fig. 2. Performance by SOM-B (top), SOM-Cx (middle-up), GTM (middle-down) and kMER (bottom) on 7M-5D data set: (a) Sammon projected maps; (b) DL matrices; (c) MID matrices; (d) Labels matrices

5 Summary and conclusions

We have revisited a universal approach to cluster detection based on the SOM structure and neuron proximity. We have shown that good results are obtained most of the time when maps are monitored (stopped early) via the UDL criterion introduced in [7]. Early stopping seems indeed almost a requirement if the ultimate goal of the analysis depends on having a faithful approximation to the data-generating distribution. Other stopping criteria can be seen in [13].

Perhaps the Gaussian kernels in GTM are too constrained by the transformation from lattice to input space, for it appears that these kernels cannot move freely when needed at some point along the training process. We have also seen that many of the previous drawbacks are avoided by kMER, which produces

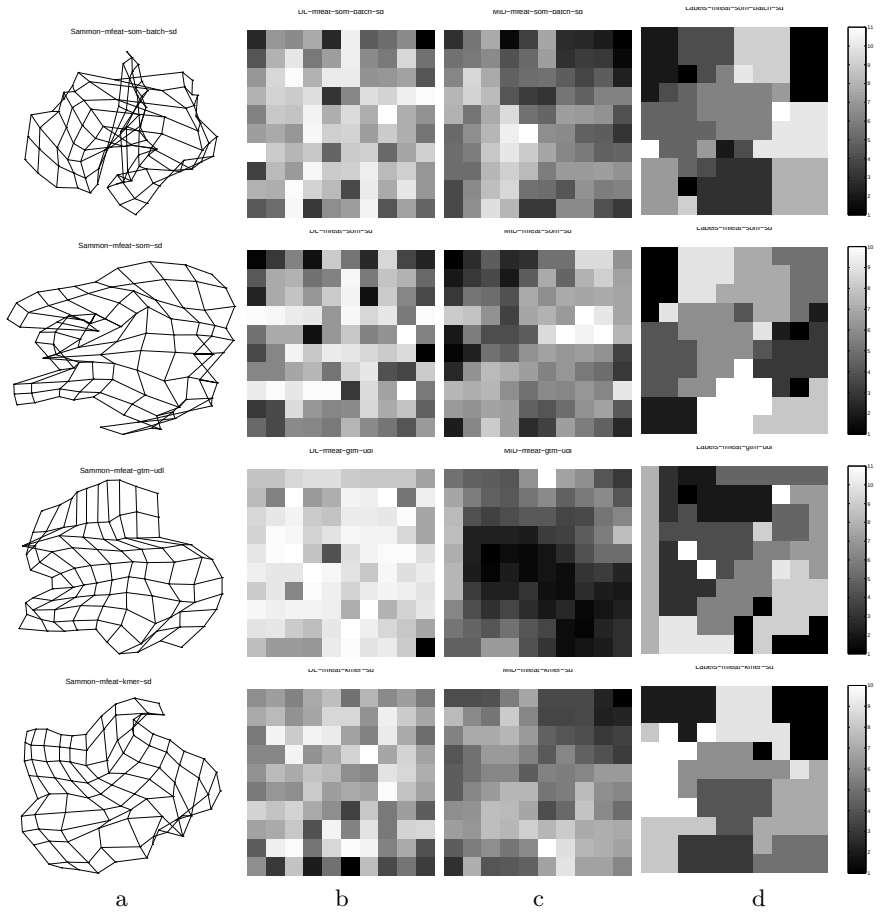

Fig. 3. Performance by SOM-B (top), SOM-Cx (middle-up), GTM (middle-down) and kMER (bottom) on Mfeat data set: (a) Sammon projected maps; (b) DL matrices; (c) MID matrices; (d) Labels matrices

more flexible and more effective maps, yet SOM-B and SOM-Cx seem also very well behaved and quite useful in the cases studied.

The scope of the above ideas for SOM-based biosignal clustering is important in as much as vectorial data keep on being worked out by researchers. Our results should be reassuring for practitioners following strategies based on neuron proximities, but they should also be recalled of the need to monitor map formation closely.

References

1. T. Kohonen, *Self-Organizing Maps.* Berlin: Springer-Verlag, 3rd extended ed., 2001.

2. E. A. Ferrán and P. Ferrara, "Topological maps of protein sequences," *Biological Cybernetics*, vol. 65, pp. 451–458, 1991.
3. T. Kohonen and P. Somervuo, "How to make large self-organizing maps for non-vectorial data," *Neural Networks*, vol. 15, pp. 945–952, 2002.
4. A. G. Gray and A. W. Moore, "Nonparametric density estimation: Toward computational tractability," in *SDM: Proceedings of the Third SIAM International Conference on Data Mining, San Francisco, CA, USA, May 1-3, 2003*, D. Barbará and C. Kamath, Eds. SIAM, 2003.
5. P. L. Davies and A. Kovac, "Densities, spectral densities and modality," *Annals of Statistics*, vol. 32, pp. 1093–1136, 2004.
6. D. W. Scott and W. F. Szewczyk, "The stochastic mode tree and clustering," *Journal of Computational and Graphical Statistics*, 2000.
7. J. Muruzábal and S. Vegas-Azcárate, "On equiprobabilistic maps and plausible density estimation," in *5th Workshop On Self-Organizing Maps, Paris*, 2005.
8. Y. Zheng and J. F. Greenleaf, "The effect of concave and convex weight adjustements on self-organizing maps," in *IEEE Transactions on Neural Networks*, vol. 7-1, 1996, pp. 87–96.
9. C. M. Bishop, M. Svensén, and C. K. I. Williams, "Gtm: The generative topographic mapping," *Neural Computation*, vol. 10, pp. 215–235, 1997.
10. M. M. Van Hulle, "Kernel-based equiprobabilistic topographic map formation," *Neural Computation*, vol. 10(7), pp. 1847–1871, 1998.
11. H.-U. Bauer and K. Pawelzik, "Quantifying the neighborhood preservation of self-organizing feature maps," *IEEE Trans. Neural Networks*, vol. 3, pp. 570–579, 1992.
12. S. Kaski and K. Lagus, "Comparing self-organizing maps," in *Proceedings of ICANN'96, International Conference on Artificial Neural Networks, Lecture Notes in Computer Science*, v. S. W. V. J. C. von der Malsburg, C. and B. Sendhoff, Eds., vol. 1112. Springer, Berlin, 1996, pp. 809–814.
13. T. Villmann, R. Der, M. Herrmann, and T. Martinetz, "Topology preservation in self-organizing feature maps: Exact definition and measurement," *IEEE Trans. Neural Networks*, vol. 8(2), pp. 256–266, 1997.
14. J. Lampinen and T. Kostiainen, "Overtraining and model selection with the self-organizing map," in *Proc. IJCNN'99, Washington, DC, USA*, 1999.
15. K. Haese and G. J. Goodhill, "Auto-som: Recursive parameter estimation for guidance of self-organizing feature maps," *Neural Computation*, vol. 13, pp. 595–619, 2001.
16. M. M. Van Hulle, *Faithful representations and topographic maps: From distortion-to information-based self-organization*. New York: Wiley, 2000.