Application of Self-Organizing Maps to the Maritime Environment

Victor J.A.S. Lobo[1],

[1] Portuguese Naval Academy,
Alfeite, 2810-001 Almada, Portugal
vlobo@isegi.unl.pt

Abstract. Self-Organizing Maps, or Kohonen networks, are a widely used neural network architecture. This paper starts with a brief overview of how self-organizing maps can be used in different types of problems. A simple and intuitive explanation of how a self-organizing map is trained is provided, together with a formal explanation of the algorithm, and some of the more important parameters are discussed. Finally an overview of different applications of SOMs in maritime problems is presented.

Keywords: Self-Organizing Maps, SOM, Kohonen Networks.

1   Introduction

Although the term "Self-Organizing Map" has been used to designate a number of different entities, it generally refers to Kohonen's Self Organizing Map [1], or SOM for short. These maps are also referred to as "Kohonen Neural Networks"[2], "Topological Neural Networks"[3], "Self-organizing Feature Maps-SOFM", or "Topology preserving feature maps" [1], or some variant of these names.

   Professor Kohonen worked on auto-associative memory during the 70's and early 80's, and presented his self-organizing map algorithm in 1982 [4]. However, it was not until the publication of the second edition of his book "Self-Organization and Associative Memory" in 1988 [5], and his paper named "The Neural Phonetic Typewriter" on IEEE Computer [5] that his work became widely known. Since then there have been many excellent papers and books on SOM, but his 2001 book [1]is generally regarded as the main reference on the subject. This book has had very flattering reviews, presenting a thorough covering of the mathematical background for SOM, it's physiological interpretation, the basic SOM, developments and applications.

Although Professor Kohonen has retired, his research group maintains a very good web-site at Helsinki's Technical University at "http://www.cis.hut.fi/research". That site contains public domain software, various manuals, papers, technical reports, and a very thorough and searchable list of papers dealing with SOM (available at "http://www.cis.hut.fi/research/som-bibl" and containing a total of 7718 references in December 2008). The som_pak programs, that are available with source code, and the Somtoolbox for Matlab are of particular interest to anyone wanting to experiment with SOM. We strongly recommend a visit to these sites.

Kohonen himself describes SOM as a "visualization and analysis tool for high dimensional data". These are indeed the two most attractive characteristics of SOM, but, as we shall see, it can be used many other applications.

1.1 - What can a SOM do?

Despite de simplicity of the SOM algorithm, it can and has been used to perform many different tasks, the most common of which are:

1. Clustering (k-means type clustering) – This is probably the most common application of SOM, albeit probably not the best. In this context, the SOM is used as an alternative to k-means clustering [6-8], *i.e.*, given a fixed number k of clusters, the SOM will partition the available data into k different groups. As an example, we may want to divide customers into 4 different groups according to their characteristics, for marketing purposes. The main advantage of o SOM in this case is that it is less prone to local minima than the traditional k-means clustering algorithm, and thus can act as a good initialization algorithm for that method. In fact, it can substitute k-means altogether, for as noted in [9], the final stages of the SOM training algorithm are exactly the same as the k-means algorithm. An extra bonus of the SOM algorithm is that the clusters obtained are topologically ordered, *i.e.*, similar clusters are (usually) grouped together.

2. Exploratory data analysis and visualization – This is, arguably, the most important application of SOM. In this case, the SOM is used as a non-linear projection algorithm, mapping n-dimensional data onto a 1 or 2 dimensional grid. The SOM can thus be an alternative to PCA projections, Principal curves, or Multi-dimensional Scaling (MDS) algorithms such as Sammon Mappings [10]. Different projection algorithms perform different trade-offs when mapping from high to low dimension, since in all but the most trivial cases some information will be lost. The main advantage of projecting multidimensional data onto 1 or 2 dimensions is that we can easily visualize the data in these dimensions. From this visualization we can identify outliers (data points that are far from other data), identify data that is similar to a given reference, or generally compare different data. If we project data onto 1 dimension, we may then plot histograms, and thus identify "natural" clusters of data. A similar result may be obtained with a technique closely related to SOM called U-Matrix [11] that can be extended to visualize what can loosely be interpreted as a 2-dimensional histogram.

3. Ordering of multidimensional data – This type of application makes use of the topological ordering of the SOM to organize a given set of data vectors according to some criteria. As an example, a 1-dimensional SOM can be used to solve the well-known travelling salesman or related problems [12]. Another interesting use of this ordering capacity of a SOM is to create colour palettes from pictures.

4. Supervised data classification – The SOM is not meant to be a classifier, and a related technique called LVQ-Linear Vector Quantization [1] is best suited for this task. However, just like the centroids obtained by a k-means algorithm, a SOM may also be used supervised classification by labeling the neurons (or units) with the classes of the data that are mapped to it.

5. Sampling – The units of a SOM have a probability distribution that is a function of the probability distribution of the data used for training. Generally, the SOM will over represent regions of the input space that have a low density, but that is many times an advantage since it helps detect outliers and novel data patterns.

6. Feature extraction – Since the SOM performs a mapping from a high-dimensional space to a low dimensional one, it may be used for feature extraction. In the simpler case, the new features are simply the coordinates of the mapped data point. This is one of the few cases where SOMs with a dimension greater than 2 are easy to use.

7. Control and/or data sensitive processing – A SOM can be used to select, based on available data, the best model, controller, or data processor for a given situation. The main idea behind this type of application is that instead of designing a rather complex controller, multiple simple controllers may be used, each one tuned to a particular type of situation. During the training of the SOM the input data is partitioned into various Voronoi regions, and each of these is used to train or define the parameters of a different controller.

8. Data interpolation – When using the SOM to interpolate data, the output space of the SOM will have the same dimension as the input space, but since the units are ordered on a regular grid, that grid provides a locally linear interpolator for the data.

Beyond these more typical applications of SOM, there have been many others, and a complete list is not practical or indeed interesting. An example of an unexpected application is the use of SOM to draw cartograms [13].

2 Basic Principles

A SOM is single layer neural network. The name neural network, or more correctly artificial neural network, is due to the historical fact that they were originally inspired by the way biological neurons where believed to work. Although this analogy is, generally speaking, still valid, developments in artificial neural networks and in our knowledge of how biological neurons actually work, have let many researchers to refer to the basic computing units of artificial neural networks not as "neurons", but as "units". In this paper, to stress the difference between the mathematical model of a biological neuron and our computational units, we will follow the more recent conventions, and refer to them simply as "units".

There are also many terms used to designate the data that is used to train the network, or later to use it. In this paper we will follow the term most used in the pattern recognition community, which is simply "pattern" or "data pattern". Different communities will call it "sample", "instance", "point", or "entity".

In a SOM, the units are set along an n-dimensional grid. In most applications this grid is 2-dimensional and rectangular, though many applications use hexagonal grids, and 1, 3, or more dimensional spaces. In this grid we can define neighborhoods in what we call the output space, as opposed to the input space of the data patterns.
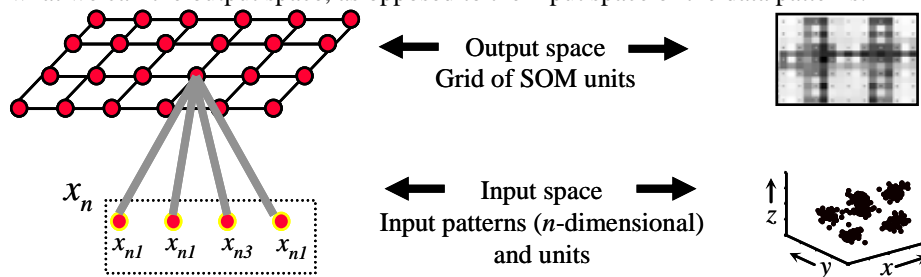

Fig. 1 - Basic SOM architecture. On the bottom, the input patterns are shown as a 4 dimensional vector (left) or 3 dimensional point (right). The units are also points in this input space. On the top, the grid of units is shown (left) together with a U-matrix coloring of a SOM.

Each unit, being an input layer unit, has as many weights or coefficients as the input patterns, and can thus be regarded as a vector in the same space as the patterns. When we train or use a SOM with a given input pattern, we calculate the distance between that pattern and every unit in the network. We then select the unit that is closest as the winning unit (or Best Matching Unit – BMU), and say that the pattern is mapped onto that unit. If the SOM has been trained successfully, then patterns that are close in the input space will be mapped to units that are close (or the same) in the output space and, hopefully, vice-versa. Thus, SOM is "topology preserving" in the sense that (as far as possible) neighborhoods are preserved through the mapping process.

Generally, no matter how much we train the network, there will always be some difference between any given input pattern and the unit it is mapped to. This is a situation identical to vector quantization, where there is some difference between a pattern and its code-book vector representation. This difference is called quantization error, and is used as a measure of how well map units represent the input patterns.

We can look at a SOM as a "rubber surface" that is stretched and bent all over the input space, so as to be close to all the training points in that space. In this sense, a SOM is similar to the input layer of a Radial Basis Function neural network (*e.g.* [14]), a neural gas model [15], or a *k*-means algorithm. The big difference is that while in these methods there is no notion of "output space" neighborhood (all units are "independent" from each other), in a SOM the units are "tied together" in the output space. It thus imposes an ordering of the units that is not present in the other methods. These ties are equivalent to a strong lateral feedback, common in other competitive learning algorithms.

Let us imagine a very simple example, where we have 4 clusters of 3 dimensional training patterns, centered at four of the vertices of the unit cube: (0,0,0), (0,0,1), (1,1,0), and (1,1,1). If we trained a 2 dimensional, 4 node map, we would expect to obtain units centered at those vertices. If we use a larger map, with 16 nodes, for example, we would expect to obtain a map where the units are grouped in clusters of 4 nodes on each of the vertices (see Fig. 2).
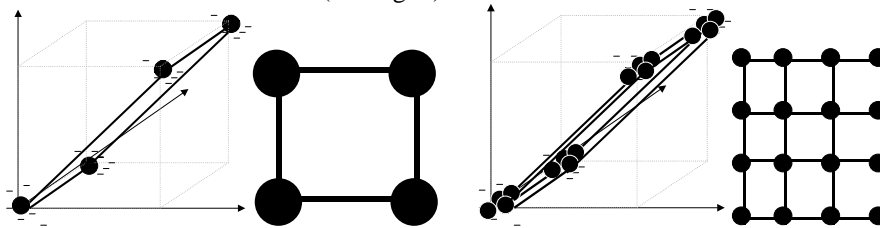

Fig. 2. Left: a 4 unit 2D SOM clustering some vertices of a 3D unit cube. On the far left we can see the units in the input (data) space, and center left in the output (grid) space. Right: a 16 unit SOM clustering the same data.

Before training, the units may be initialized randomly. During the first part of training, they are "spread out", and pulled towards the general area (in the input space) where they will stay. This is usually called the unfolding phase of training. After this phase, the general shape of the network in the input space is defined, and we can then proceed to the fine tuning phase, where we will match the units as far as possible to the input patterns, thus decreasing the quantization error.

To visualize the training process, let us follow a 2-dimensional to 1-dimensional mapping presented in [1]. In this problem, 2-dimensional data points are uniformly distributed in a triangle, and a 1-dimensional SOM is trained with these patterns. Figure 4 represents the evolution of the units in the input space. As training proceeds, the line first unfolds (steps 1 to 100), and then fine-tunes itself to cover the input space.
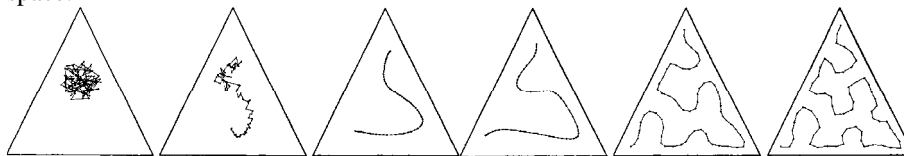

Fig. 3 -2D to 1D mapping by a SOM, from [1]. The number of iterations used increases from the left to right, from 0 to 25000.

3 - Description of the training algorithm

3.1 - The algorithm

Let x_k (with $k=1$ to the number of training patterns N) be the n-dimensional training patterns. Let w_{ij} be the unit in position (i,j). Let $0 \leq \alpha \leq 1$ be the learning rate (sometimes referred to as η), and $h(w_{ij}, w_{mn}, r)$ be the neighborhood function (sometimes referred to as Λ or Nc). This neighborhood function has values in [0,1] and is high for units that are close in the output space, and small (or 0) for units far away. It is usual to select a function that is 1 if $w_{ij}=w_{mn}$, monotonically decreases as the distance in the grid between them increases up to a radius r (called neighborhood radius) and is zero from there onwards. Let w_{bmu} be the best matching unit for a given input pattern.

The algorithm for training the network is then:

For each input pattern x_k:
1. Calculate the distances between the pattern x_k and all units w_{ij}: $d_{ij} = \| x_k - w_{ij} \|$
2. Select the nearest unit w_{ij} as best matching unit $w_{bmu} = w_{ij} : d_{ij} = \min(d_{mn})$
3. Update each unit w_{ij} according to the rule $w_{ij} = w_{ij} + \alpha \ h(w_{bmu}, w_{ij}, r) \ \| x_k - w_{ij} \|$
4. Repeat the process until a certain stopping criterion is met. Usually, the stopping criterion is a fixed number of iterations. To guarantee convergence and stability of the map, the learning rate α and neighborhood radius r are decreased in each iteration, thus converging to zero.

The distance measure between the vectors is usually the Euclidean distance, but many others can and are used, such as norm based Minkowski metrics, dot products, director cosines, Tanimoto measures, or Hausdorff distances.
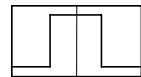
3.2 - Neighborhood functions

The neighborhood function provides a bond between a unit and its neighbors, and is responsible for the topological ordering of the map. In fact, without this neighborhood function (or when it's radius is zero), the SOM training algorithm is exactly the same as the incremental k-means algorithm [6]. The two most common neighborhood functions are the Gaussian and the square (or bubble) functions:

$$h_g(w_{ij}, w_{mn}, r) = e^{-\frac{1}{2}\left(\frac{\sqrt{(i-n)^2+(j-m)^2}}{r}\right)^2}$$
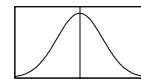
$$h_s(w_{ij}, w_{mn}, r) = \begin{cases} 1 \Leftarrow \sqrt{(i-n)^2+(j-m)^2} \leq r \\ 0 \Leftarrow \sqrt{(i-n)^2+(j-m)^2} > r \end{cases}$$

In both cases, r decreases to 0 or 1 during training. If r→ 0 the final quantization error will be minimized, but the topological ordering may be lost, since the algorithm is performing a k-means clustering. On the other hand, forcing r→ 1 will preserve the ordering of the units, but the quantization error will not be minimized. Moreover, in this case there will be a boarder effect, by which units close to the border will be dragged to the center, and present higher quantization errors.

The algorithm is surprisingly robust to changes in the neighborhood function, and our experience is that it will usually converge to approximately the same final map, whatever our choice, providing the radius and learning rate decrease to 0. The Gaussian neighborhood tends to be more reliable (different initializations tend to converge to the same map), while the bubble neighborhood leads to smaller quantization errors, and is computationally much faster. A theoretic discussion of the effect of neighborhood functions (although only for the 1-dimensional case) can be found in [17], and a less rigorous but more general one in [18].

3.3 – Other parameters and training options

As mentioned before, training is usually done in two phases: the unfolding phase, and the fine-tuning phase. The algorithm is exactly the same in both cases, but while in the first phase the neighborhood radius and learning rate have rather high values (to allow for a general orientation of the map), in the second phase they will have smaller values, to perform only fine adjustments on the unit's positions. As a rule of thumb, the initial radius for the first phase should be roughly the length of the smaller side of the map, while for the second it should be the radius of the expected size of clusters in the output space.

The size of the map will depend a lot on the particular problem at hand and on the data available. If the SOM is to be used as alternative to k-means, one unit per desired cluster should be used. For that type of application, a 1-dimensional SOM will usually provide the best results [9]. For exploratory data analysis, a larger map should be used. These are sometimes called Emergent-SOM or ESOM [19]. Depending on the amount and variability of available data, a rule of thumb could be to use one unit for each 4 to 20 or more data patterns, but in some cases one might use more units than data patterns (to obtain very clear cut U-Matrices).

3.4 – U-Matrices

U-Matrices where introduced by Ultsch [11], and are one of the most popular and useful ways of visualizing clusters with a SOM. A U-Matrix is obtained by computing the distance in the input space of units that are neighbors in the output space. If these differences are small, it means that the units are close together, and thus there is a cluster of similar data in that region of the input space. On the other hand, if the distances are large, the units are far apart, and there isn't much data in that region of the input space. The U-Matrix can thus be seen as a sort of extension of an inverted histogram for multidimensional data projected on a lower dimensional space: low

values indicate large concentrations of data, and high values indicate sparse regions. U-Matrices are usually presented as color-coded maps: white regions indicate low values (and thus clusters), while dark regions indicate separations between clusters.

4 - SOM variants

Many different variants of the basic SOM algorithm have been proposed, and a complete review of these is beyond the scope of this paper. Some reviews of these variants have been published [20, 21], and we will overview some of them to show how the basic algorithm can be adapted to different problems.

The original SOM algorithm and most of its variants deal with vector data only. Some variants for non-vector data have also been proposed namely the Dissimilarity SOM [22], the Kohonen Multiple Correspondence Analysis and the Kohonen algorithm on Disjuctive Table [23]. For the simpler case of binary valued data, both the original algorithm using 0 and 1 as real numbers, and binary variants of SOM produce good results [24, 25].

SOMs have frequently been used to analyze temporal data, such as EEG or Stock Exchange data. In most cases time can be imbedded into the data vector, and a standard SOM algorithm is used, treating that vector as a simple input pattern. More interesting uses of SOM have been made by changing the learning rule or by changing the topology or structure of the network so as to explicitly take time into consideration. In the former case, the learning rule may, for example, consider only the neighbors of the last BMU as candidates for the next input pattern, or separate the time variable from the rest when computing the similarity. As for changes in topology and structure, some approaches use hierarchical SOMs with different time frames, or include time delay memories in the units. A review of the different ways in which this has been done, together with a proposal for a taxonomy of temporal SOMs is available in [26].

Geographical information science problems also have a special variable (special location) that should, like time, be treated in a different way. To this end a variant called GeoSOM has also been developed [21, 27, 28].

Hierarchical SOMs [29, 30] combine several SOMs to process data a low level, and then use their outputs as inputs to a high level SOM that fuses the results.

In some applications, the notion of output grid is substituted by a more general graph, such as happens in the Minimum Spanning Tree SOM [20], Tree-Structured SOM [29], or Growing Cells [31, 32]. The links and concept of output space may even disappear, as happens in the Neural Gas model [15, 33, 34].

Another important type of variants on the basic SOM algorithm are those that try and overcome the theoretical obstacles raised by the fact that the SOM does not minimize a global energy function. One solution is to change the learning rule slightly, as was done in [35]. Another solution is to use a variation of Gaussian Mixture Models to derive a topologically ordered map, as is done with the Generative Topographic Mapping [36]. However, despite the theoretical soundness of these methods, they do not provide significantly better results and are computationally more complex than the original algorithm.

5 - Applications in maritime environment

Given the wide range of capabilities of the SOM there have been many applications of this technique on maritime problems.

SOMs have been used quite frequently to cluster and classify satellite images [3, 37-41]. In most cases, the SOM is basically used as a classifier, and each pixel of the satellite image forms a data pattern. When analyzing satellite images, the ground truth (*i.e.*, the real class of a given pixel) is usually established by an expert, and is rather slow, expensive, and prone to errors. Therefore not many classified pixels are available. One advantage of the SOM in this case is that it may be trained with all the data, including non-classified pixels, and then labeled with only the classified ones. This labeling may then be extended to other units that belong to the same cluster, improving the classification capabilities of the system. Very similar approaches have been made with data that combines satellite images with other data [42], data obtained by radars [43], data obtained by meteorological stations [44], airborne lasers [45], or even data obtained by simulators. The common factor in all these cases is that a 2-dimensional map with pixels that are multidimensional vectors is presented to a SOM for clustering and classification. Let us see one of these with a little more detail, and then overview the problems where these approaches were successfully applied.

One application of SOM to satellite images, that concerns reflectance spectra of ocean waters, is presented in [3]. In this case, a 20x20 unit probabilistic SOM (or more precisely PSOM) is trained with 43000 6-dimensional vectors. Each of these corresponds to sampled pixels of a satellite image with 5 pre-processed frequency bands, and an extra value corresponding to the spatial standard deviation of one of those measurements. A human expert will then label some of the pixels, and these are used to label the SOM units, either directly or indirectly after these are clustered with a hierarchical clustering algorithm. The authors point out that the method used provides a good overall classification of the data, in part due to the fact that that the probabilistic nature of PSOM allows for a confidence level to be assigned to each classification. The PSOM is also considered useful by showing that a lot of resources are dedicated to separating clouds from other pixels, thus leading to the suggestion that the images be pre-processed to remove these clouds. The author's main interest is in the characterization of Sahara dust, clouds, and other aerosols present over the ocean, and they do not go into great detail on the parameterization of the PSOM. It could be argued that a non-square map would lead to a better stabilization of the training process, and that the use of a U-Matrix would help define larger clusters (instead of using hierarchical clustering), but the authors did not follow that path.

The SOM has been used in a similar way (*i.e.* for clustering and classifying data contained in 2-dimensional maps or images), in many applications of environmental science, climatology, geology, and oceanography. These include analyzing sea surface temperature [46-49], plankton [50, 51], ocean current patterns [43, 52], estuary and basin dynamics [53], sediment structure [54], atmospheric pressure [55, 56], wind patterns [39], storm systems [41], the El Niño weather conditions [42], clouds [57], ice [53, 58, 59], rainfall [44, 60, 61], oil spills [45], the influence of ocean conditions in droughts [62], and the relationship between sardine abundance and upwelling phenomena [40].

Data concerning fisheries was analyzed in different perspectives using a SOM in [63]. The use of SOM in this case clearly shows the existence of well defined changes in fisheries along time, and relationships between different species.

A more creative use of SOM in shown in [64], where the SOM is used to segment maps of the seafloor obtained with multibeam sonars. The segmented data is then classified with specialized classifiers for each segment. The SOM is thus used to pre-process the data so that multiple simpler or more precise classifiers can be used to obtain the desired results.

Although classical harmonical methods can provide good sea level predictions in most cases, those predictions can have rather large errors in basins, estuaries, or regions where weather conditions have a large influence. In those cases, SOMs have been used to predict sea levels with greater accuracy in [65].

Following an approach common in several problems in robotics [66], the SOM has been used to control a Underwater Autonomous Vehicle (AUV) [67-69]. The basic idea in this type of application is that the SOM receives the sensor inputs, and based on that chooses a unit that will provide the guidance for the AUV. The main advantage of the SOM in this case is that each of the units has a quite simple control law (as opposed to a complicated non-linear controller), and the topological ordering of the SOM makes it relatively robust to noise in the inputs.

With the increase in maritme traffic, the consequences of accidents, and the availability of Vessel Traffic Sytems (VTS), the automatic detection of anomalous behavious of ships became a pressing problem. This problem was addressed, in [70], where track data (heading, speed, etc) from navy exercises was used to train a SOM. Clusters where then identified on that SOM, and both suspicious behavior clusters and outliers where flagged as potential threats. The same problem was tackled in a similar way in [71]. In this case the emphasis in more on visualization of the data, and on estimating the probability of a given situation occurring in the dataset.

Also related with ship trajectories, SOMs have been used to plan patrol trajectories of naval vessels in [72]. The approach followed was basically the one used to solve the traveling salesman problem with a SOM (*e.g.* [12]). In this case, the geographical locations of "incidents" (accidents and illegal fishing) where used as training patterns, and the trajectory obtained tries to maximize the probability of passing in the area where there were "incidents" in the past.

In underwater acoustics SOMs have been used extensively to analyze passive sonar recordings [73-76]. Although ship noise or transient recognition is basically a supervised task, it is very important to detect novelties, and to relate those novelties with known causes. The SOM can provide this by using large maps which will have many unlabeled units. Additionally it provides an easy to use and understand interface for the operators.

Also concerning fluids, although not directly applied to the maritime environment, an interesting use of SOM is given in [77, 78] for analyzing movement in fluids by tracking particles in suspension. The idea is to use successive images of the fluid for training a map, and then infer the movement by observing how the units change from one step to the next.

6 - Conclusions

An introduction to how a Self-Organizing Map (SOM) works and how it can be used was presented. Despite its simplicity, the SOM can be used for a wide variety of applications. Some of its shortcomings where also pointed out, as well as the main issues that must be taken into consideration when using them.

An overview of applications in the marine environment was given, showing that it has successfully been used in many real maritime problems. I believe that its use in this field is still at a preliminary stage, and more and more powerful uses will be given to SOM. It is many times used simply for k-means type clustering and supervised classification. While those types of applications are useful, I think that the greatest potential of SOM is its ability do project and visualize multidimensional data. Many authors have criticized clustering through visualization as too subjective for engineering purposes. I would argue that clustering is intrinsically a subjective problem, and that the human eye and judgment are the best tools available for that task. The computer algorithms should only present the data in a suitable way, which is exactly what a SOM does. I also believe that there is still a lot of potential for using SOM in non-linear control and routing or piping problems aboard ships. As SOMs become more mainstream, and software for their use becomes more widespread, they will probably be used in creative ways in even more problems.

References

1. Kohonen, T., Self-Organizing Maps. 3rd ed. Information Sciences. 2001, Berlin-Heidelberg: Springer. 501.
2. Fu, L., Neural Networks in Computer Intelligence. 1994, Singapore: McGraw Hill. 460.
3. Niang, A., et al., Automatic neural classification of ocean color reflectance spectra at the top of the atmosfphere with introduction of expert knowledge. Remote Sensing of Environment, 2003(86): p. 257-271.
4. Kohonen, T. Clustering, Taxonomy, and Topological Maps of Patterns. in Proceedings of the 6th International Conference on Pattern Recognition. 1982.
5. Kohonen, T., The 'neural' phonetic typewriter. IEEE Computer, 1988. 21(3): p. 11-22.
6. MacQueen, J. Some methods for classification and analysis of multivariate observation. in 5th Berkeley Symposium on Mathematical Statistics and Probability. 1967: University of California Press.
7. Loyd, S.P., Least Squares quantization in PCM. IEEE Transactions on Information Theory, 1982. 28(2): p. 129-137.
8. Selim, S.Z. and M.A. Ismail, k-means type algorithms: a generalized convergence theorem and characterization of local optimality. IEEE Trans. Pattern Analysis and Machine Intelligence, 1984. 6: p. 81-87.
9. Bacao, F., V. Lobo, and M. Painho, Self-organizing Maps as Substitutes for K-Means Clustering, in Lecture Notes in Computer Science,, V.S. Sunderam, et al., Editors. 2005, Springer-Verlag: Berlin Heidelberg. p. 476-483.

10. Sammon, J.W.J., A Nonlinear Mapping for Data Structure Analysis. IEEE Transactions on Computers, 1969. C-18(5): p. 401-409.
11. Ultsch, A. and H.P. Siemon. Kohonen´s Self-Organizing Neural Networks for Exploratory Data Analysis. in Intl. Neural Network Conf. INNC90. 1990. Paris.
12. Altinel, I.K., N. Aras, and B.J. Oommen, Fast, Efficiente and accurate solutions to the Hamiltonian path problem using neural approaches. Computers & Operations Research, 2000. 27: p. 461-494.
13. Henriques, R., Cartogram creation using self-organizing maps, in ISEGI. 2006, New University of Lisbon: Lisbon. p. 144.
14. Haykin, S., Neural Networks: A Comprehensive Foundation. 2 ed. 1999.
15. Martinetz, T.M., S.G. Berkovich, and K.J. Schulten, Neural-Gas network for vector quantization and its application to time-series prediction. IEEE Transactions on Neural Networks, 1993. 4(4): p. 558-569.
16. Kohonen, T., Self-Organizing Maps. 1st ed. 1995, Berlin-Heidelberg: Springer. 362.
17. Erwin, E., K. Obermeyer, and K. Schulten, Convergence properties of self-organizing maps, in Artificial Neural Networks, T. Kohonen, et al., Editors. 1991, Elsevier. p. 409-414.
18. Ritter, H., T.M. Martinetz, and K. Schulten, Neural Computation and Self-Organizing Maps: an introduction. 1992: Addison-Wesley.
19. Ultsch, A. Clustering with SOM: U*C. in WSOM 2005. 2005. Paris.
20. Kangas, J.A., T.K. Kohonen, and J.T. Laaksonem, Variants of Self-Organizing Maps. IEEE Transactions on Neural Networks, 1990. 1(1): p. 93-99.
21. Bação, F., V. Lobo, and M. Painho, The Self-Organizing Map, the Geo-SOM, and relevant variants for geosciences. Computers and Geosciences, 2005. 31(2): p. 155-163.
22. Ambroise, C., et al. Analyzing dissimilarity matrices via Kohonen maps. in 5th Conference of the International Federation of Classification Societies (IFCS 1996). 1996. Kobe (Japan).
23. Cottrell, M., S. Ibbou, and P. Letremy, SOM-based algorithms for qualitative variables. Neural Networks, 2004. 17(8-9): p. 1149-1167.
24. Lobo, V., N. Bandeira, and F. Moura-Pires. Distributed Kohonen networks for Passive Sonar Based Classification. in FUSION 98. 1998. Las Vegas, NV, USA.
25. Lourenço, F., V. Lobo, and F. Bação. Binary-based similarity measures for categorical data and their application in Self-Organizing Maps. in JOCLAD 2004 - XI Jornadas de Classificação e Análise de Dados. 2004. Lisbon.
26. Guimarães, G., V. Lobo, and F. Moura-Pires, A taxonomy of Self-organizing Maps for temporal sequence processing. Intelligent Data Analysis, 2002. 7(4).
27. Bacao, F., V. Lobo, and M. Painho, Applications of Different Self-Organizing Map Variants to Geographical Information Science Problems, in Self-Organizing Maps - Applications in Geographic Information Science, P. Agarwal and A. Skupin, Editors. 2008, John Wiley & Sons: Chichester. p. 205.
28. Bação, F., V. Lobo, and M. Painho. Geo-SOM and its integration with geographic information systems. in WSOM 05, 5th Workshop On Self-Organizing Maps. 2005. Paris.

29. Koikkalainen, P. and E. Oja. Self-organizing hierarchical feature maps in IJCNN'90 - International Joint Conference on Neural Networks. 1990. Washington, DC, USA.

30. Kemke, C. and A. Wichert. Hierarchical Self-Organizing Feature Maps for Speech Recognition. in WCNN'93 - World Conference on Neural Networks. 1993: Lawrence Erlbaum, Hillsdale.

31. Fritzke, B. Let it Grow - Self-organizing Feature Maps With Problem Dependent Cell Structure. in ICANN-91. 1991. Helsinki: Elsevier Science Publ.

32. Fritzke, B. Growing Self-organizing Networks - Why? in ESANN'96 European Symposium on Artificial Neural Networks. 1996.

33. Fritzke, B., A growing neural gas network learns topologies, in Advances in Neural Information Processing Systems, G. Tesauro, D.S. Touretzky, and T.K. Leen, Editors. 1995, MIT Press: Cambridge MA. p. 625-632.

34. Hammer, B., A. Hasenfuss, and T. Villmann, Magnification control for batch neural gas. Neurocomputing, 2007. 70(7-9): p. 1225-1234.

35. Heskes, T., Energy Functions for Self-Organizing Maps, in Kohonen Maps, E. Oja and S. Kaski, Editors. 1999, Elsvier: Amsterdam. p. 303-316.

36. Bishop, C.M., M. Svensen, and C.K.I. Williams, GTM: The Generative Topographic Mapping. Neural Computation, 1998. 10(1): p. 215-234.

37. Mather, P.M., B. Tso, and M. Koch, An evaluation of Landsat TM spectral data and SAR-derived textural information for lithological discrimination in the Red Sea Hills, Sudan. International Journal of Remote Sensing, 1998. 19(4): p. 587-604.

38. Villmann, T., E. Merenyi, and B. Hammer, Neural maps in remote sensing image analysis. Neural Networks, 2003. 16(3-4): p. 389-403.

39. Richardson, A.J., C. Risien, and F.A. Shillington, Using self-organizing maps to identify patterns in satellite imagery. Progress in Oceanography, 2003. 59(2-3): p. 223-239.

40. Hardman-Mountford, N.J., et al., Relating sardine recruitment in the Northern Benguela to satellite-derived sea surface height using a neural network pattern recognition approach. Progress in Oceanography, 2003. 59(2-3): p. 241-255.

41. Parikh, J.A., et al., An evolutionary system for recognition and tracking of synoptic-scale storm systems. Pattern Recognition Letters, 1999. 20(11-13): p. 1389-1396.

42. Leloup, J.A., et al., Detecting decadal changes in ENSO using neural networks. Climate Dynamics, 2007. 28(2-3): p. 147-162.

43. Liu, Y., R.H. Weisberg, and L. Shay, Current Patterns on the West Florida Shelf from Joint Self-Organizing Map Analyses of HF Radar and ADCP Data. Journal of Atmospheric and Oceanic Technology, 2007. 24: p. 702-712.

44. Cavazos, T., Using self-organizing maps to investigate extreme climate events: An application to wintertime precipitation in the Balkans. Journal of Climate, 2000. 13(10): p. 1718-1732.

45. Lin, B., et al. Neural networks in data analysis and modeling for detecting littoral oil-spills by airborne laser fluorosensor remote sensing. in Conference on Ocean Remote Sensing and Applications. 2002. Hangzhou, Peoples R China: Spie-Int Soc Optical Engineering.

46. Liu, Y.G., R.H. Weisberg, and R.Y. He, Sea surface temperature patterns on the West Florida Shelf using growing hierarchical self-organizing maps. Journal of Atmospheric and Oceanic Technology, 2006. 23(2): p. 325-338.

47. Liu, Y.G., R.H. Weisberg, and Y.C. Yuan, Patterns of upper layer circulation variability in the South China Sea from satellite altimetry using the self-organizing map. Acta Oceanologica Sinica, 2008. 27: p. 129-144.

48. Tozuka, T., et al., Tropical Indian Ocean variability revealed by self-organizing maps. Climate Dynamics, 2008. 31(2-3): p. 333-343.

49. Marques, N.C. and N. Chen. Border detection on remote sensing satellite data using self-organizing maps. in 11th Portuguese Conference on Artificial Intelligence. 2003. Beja, Portugal: Springer-Verlag Berlin.

50. Chazottes, A., et al., Statistical analysis of a database of absorption spectra of phytoplankton and pigment concentrations using self-organizing maps. Applied Optics, 2006. 45(31): p. 8102-8115.

51. Solidoro, C., et al., Understanding dynamic of biogeochemical properties in the northern Adriatic Sea by using self-organizing maps and k-means clustering. Journal of Geophysical Research-Oceans, 2007. 112(C7): p. 13.

52. Liu, Y.G. and R.H. Weisberg, Patterns of ocean current variability on the West Florida Shelf using the self-organizing map. Journal of Geophysical Research-Oceans, 2005. 110(C6): p. 12.

53. Reusch, D.B. and R.B. Alley. Antarctic sea ice: a self-organizing map-based perspective. in International Symposium on Cryospheric Indicators of Global Climate Change. 2006. Cambridge, ENGLAND: Int Glaciological Soc.

54. Kropp, J. and T. Klenke, Phenomenological pattern recognition in the dynamical structures of tidal sediments from the German Wadden Sea. Ecological Modelling, 1997. 103(2-3): p. 151-170.

55. Cassano, E.N., et al., Classification of synoptic patterns in the western Arctic associated with extreme events at Barrow, Alaska, USA. Climate Research, 2006. 30(2): p. 83-97.

56. Hewitson, B.C. and R.G. Crane, Self-organizing maps: applications to synoptic climatology. Climate Research, 2002. 22(1): p. 13-26.

57. Kubo, M., K. Muramoto, and Ieee. Classification of clouds in the Japan Sea area using NOAA AVHRR satellite images and self-organizing map. in IEEE International Geoscience and Remote Sensing Symposium (IGARSS). 2007. Barcelona, SPAIN: Ieee.

58. Reusch, D.B., R.B. Alley, and B.C. Hewitson, North Atlantic climate variability from a self-organizing map perspective. Journal of Geophysical Research-Atmospheres, 2007. 112(D2): p. 20.

59. Fukumi, M., et al. Drift ice detection using a self-organizing neural network. in 9th International Conference on Knowledge-Based Intelligent Information and Engineering Systems. 2005. Melbourne, AUSTRALIA: Springer-Verlag Berlin.

60. Uotila, P., et al., Changes in Antarctic net precipitation in the 21st century based on Intergovernmental Panel on Climate Change (IPCC) model scenarios. Journal of Geophysical Research-Atmospheres, 2007. 112(D10): p. 19.

61. Chandrasekar, V. and ieee. SOM of space borne precipitation radar rain profiles on global scale. in IEEE International Geoscience and Remote Sensing Symposium. 2004. Anchorage, AK: Ieee.

62. Barros, A.P. and G.J. Bowden, Toward long-lead operational forecasts of drought: An experimental study in the Murray-Darling River Basin. Journal of Hydrology, 2008. 357(3-4): p. 349-367.

63. Hyun, K., et al., Using an artificial neural network to patternize long-term fisheries data from South Korea. Aquatic Sciences, 2005. 67(3): p. 382-389.

64. Chakraborty, B., et al., Application of artificial neural networks to segmentation and classification of topographic profiles of ridge-flank seafloor. Current Science, 2003. 85(3): p. 306-312.

65. Ultsch, A. and F. Roske, Self-organizing feature maps predicting sea levels. Information Sciences, 2002. 144(1-4): p. 91-125.

66. Barreto, G.d.A., A.F.R. Araújo, and H.J. Ritter, Self-Organizing Feature Maps for Modeling and Control of Robotic Manipulators. ááJournal of Intelligent and Robotic Systems 2003. 36(4): p. 407-450.

67. Nishida, S., et al. Adaptive learning to environment using self-organizing map and its application for underwater vehicles. in 4th International Symposium on Underwater Technology. 2004. Taipei, TAIWAN: Ieee.

68. Ishii, K., et al. A self-organizing map based navigation system for an underwater robot. in IEEE International Conference on Robotics and Automation. 2004. New Orleans, LA: Ieee.

69. Nishida, S., et al. Self-organizing decision-making system for AUV. in 5th International Symposium on Underwater Technology/5th Workshop on Scientific Use of Submarine Cables and Related Technologies. 2007. Tokyo, JAPAN: Ieee.

70. Patton, R., M. Webb, and R. Gaj, Covert Operations Detection for maritime applications. Canadian Journal of Remote Sensing, 2001. 27(4): p. 306-319.

71. Riveiro, M., G. Falkman, and T. Ziemke. Visual analytics for the detection of anomalous maritime behavior. in 12th International Conference Information Visualisation 2008. 2008. London, ENGLAND: Ieee Computer Soc.

72. Lobo, V. and F. Bacao. One dimensional Self-Organizing Maps to optimize marine patrol activities. in Oceans 2005 Europe International Conference. 2005. Brest, FRANCE: Ieee.

73. Lobo, V. and F. Moura-Pires. Ship noise classification using Kohonen Networks. in EANN 95. 1995. Helsinki, Finland.

74. Lobo, V., N. Bandeira, and F. Moura-Pires. Ship Recognition using Distributed Self Organizing Maps,. in EANN 98. 1998. Gibraltar.

75. Lobo, V., Ship noise classification: a contrinution to prototype based classifier design, in Departamento de Informatica. 2002, Universidade Nova de Lisboa: Lisbon.

76. Oliveira, P.M., et al. Detection and Classification of Underwater Transients with Data Driven Methods Based on Time-Frequency Distributions and Non-Parametric Classifiers. in MTS/IEEE Oceans'02. 2002. Biloxi, Mississipi, USA.

77. Labonté, G. A SOM neural network that reveals continuous displacement fields. in IEEE World Congress on Computational Intelligence, 1998. 1998. Anchorage, AK, USA.

78. Ohmia, K., SOM-Based particle matching algorithm for 3D particle tracking velocimetry Applied Mathematics and Computation, 2008. 205(2): p. 890-898