Limitations of the SOM and the GTM

Elias Pampalk
e9625173@student.tuwien.ac.at
February 13th, 2001 (2nd version)

In this paper I present the results of my project work for the course *"Connectionism Laboratory"* at the Department of Medical Cybernetics and Artificial Intelligence, University of Vienna, held by professor Georg Dorffner.

A summary is given of the Self-Organizing Map (SOM) and the Generative Topographic Mapping (GTM) algorithms. They are statistic tools used for clustering and multidimensional scaling. I will discuss their limitations as well as those of an alternative which combines classical algorithms to achieve similar results.

---

1. INTRODUCTION

Datamining is becoming more and more popular thanks to the rapid development of computers and the need to extract information out of increasingly large data collections.

Within datamining one interesting field is to visualize the data to obtain a better understanding. One common approach is clustering with topology preservation, which can be achieved with the very popular SOM algorithm. Very similar results are generated by the recently introduced GTM algorithm. An alternative approach is to use multidimensional scaling techniques such as Sammon's mapping in combination with classical clustering algorithms such as k-means. This paper gives a summary of the different algorithms and compares them briefly.

The remainder of this paper is structured as follows. In Section 2 I describe the SOM algorithm and in Section 3 the GTM algorithm will be introduced. In Section 4 I discuss their limitations concerning assumptions, applicability, result representation, parameters and computational cost. An alternative approach for visualization can be found in Section 5 and conclusions are presented in Section 6.

2. SELF-ORGANIZING MAP

The Self-Organizing Map (SOM) has been introduced in [Kohonen 1982] since then it has been analyzed and employed extensively. A recent overview can be found in [Kohonen 1997].

The SOM and its variants have been employed many times in a wide variety of domains, such as financial, medical or time series data analysis [Kohonen et al. 1996; Deboeck and Kohonen 1998; Simula et al. 1999].

2.1 Relation to Biological Models

The SOM is inspired by neuroscience. The two main connections are *competitive learning* and *topographic maps* as observed within nervous systems.

Figure 1[1] depicts a simple Winner-Takes-All circuit. Each cell inhibits all other

[1]These images have been taken from [Kohonen et al. 1997] without permission.

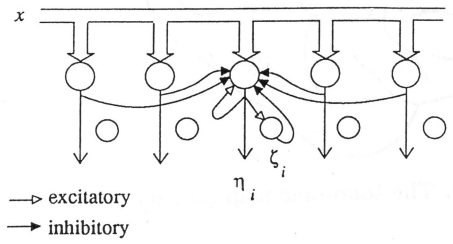

Fig. 1. Winner-Takes-All Circuit.

cells through the lateral connections and excites itself. Only the cell that receives the largest input wins and remains active.

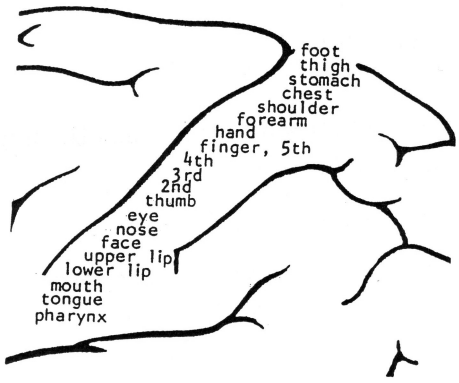

Fig. 2. Somatotopic Map.

There is a fine structure in many areas: for instance, the visual, somatosensory, etc. response signals are obtained in the same topographical order on the cortex in which they were received at the sensory organs; see, e.g., the somatotopic map shown in Figure 2[1]). For example stimulating the nerves of the thumb will activate neurons in the brain which are spatially located closely to those which respond if the nerves of the forefinger are stimulated.

2.2 Algorithm

The SOM is an unsupervised neural network mapping high dimensional input data onto a usually two-dimensional output space while preserving relations between the data items.

The SOM consists of units (neurons), which are arranged as a two-dimensional rectangular or hexagonal grid. These units represent certain vectors in the data space and can be initialized, for example, randomly or using principle component analysis. During the training process vectors from the dataset are presented to the map in random order. The unit with the highest response to a chosen vector (this unit is the so called winner) and its neighborhood are adapted in such a way as to make them more responsive if the present input and, as a result of ordering, similar inputs are presented again. Often the Euclidean distance is used to calculate similarities and a Gaussian-like function determines how strongly the neighborhood of the winner is adapted. This neighborhood function is declined during the training process as well as the learning rate which regulates the overall adaptation.

The trained SOM provides a mapping of the data space onto a two-dimensional map in such a way that similar data points are located close to each other.

2.3 Enhancements

Additional visualization techniques such as the U-Matrix [Ultsch 1993], Adaptive Coordinates [Merkl and Rauber 1998], cluster connections [Merkl and Rauber 1997], or local factors [Kaski et al. 1998] aid the user in understanding the cluster structure.

Furthermore methods like LabelSOM [Rauber 1999] allow the automatic extraction of cluster descriptions based on the attributes.

There have been many suggestions to improve the algorithm regarding certain shortcomings when employing the SOM in special domains and data mining tasks. Some of these variants are the tree-structured SOM [Miikkulainen 1990], the growing SOM [Fritzke 1996], combinations of these two [Dittenbach et al. 2000], or the adaptive-subspace Self-Organizing Map [Kohonen et al. 1997].

3. GENERATIVE TOPOGRAPHIC MAPPING

The Generative Topographic Mapping (GTM) algorithm is a new probabilistic reformulation of the SOM introduced in [Bishop et al. 1997]. It addresses some limitations of the SOM such as the lack of a cost function, the lack of a theoretical basis for parameters, lack of a proof of convergence and that hard assignments are used instead of soft ones (probabilities). Unlike the SOM, GTM has not been developed in the context of neural networks. Instead it is embedded in a statistic framework. The currently most complete description can be found in [Svensén 1998]. Further information as well as a MATLAB implementation can be found on the Web[2].

Being a rather new development it has not yet been employed extensively on real world problems.

3.1 Basic Concepts

GTM consists of a constrained mixture of Gaussians in which the model parameters are determined by maximum likelihood using the Expectation Maximization (EM) algorithm. It is defined by specifying a set of points $\{\mathbf{x}_i\}$ in latent space, together with a set of basis functions $\{\mathbf{\Phi}_j(\mathbf{x})\}$, which map the latent points continuously and non-linearly into the data space, and together with the adaptive parameters $\mathbf{W}$ and β define a constrained mixture of Gaussians with centers $\mathbf{W}\mathbf{\Phi}(\mathbf{x}_i)$ and a common covariance matrix given by $\beta^{-1}\mathbf{I}$. After initializing $\mathbf{W}$ and β, training involves alternating between the E-step in which the posterior probabilities are evaluated, and the M-step in which $\mathbf{W}$ and β are re-estimated.

3.2 Model

The continuous function $\mathbf{y}(\mathbf{x};\mathbf{W})$ defines a mapping from the latent space into the data space. Where $\mathbf{x}$ is a 2-dimensional latent variable and the matrix $\mathbf{W}$ is the parameter of the mapping. The transformation $\mathbf{y}(\mathbf{x};\mathbf{W})$ maps the latent-variable space to a non-linear Euclidean manifold embedded within the data space.

Defining a probability distribution $p(\mathbf{x})$ on the latent-variable space induces a corresponding distribution $p(\mathbf{y}|\mathbf{W})$ in the data space.

Since in reality the data $\mathbf{t} = (t_1, \ldots, t_D)$ will only approximately live on a lower-dimensional manifold, it is appropriate to include a noise model. A radially-

[2]http://www.ncrg.aston.ac.uk/GTM

symmetric Gaussian distribution centered on $\mathbf{y}(\mathbf{x};\mathbf{W})$ is chosen

$$p(\mathbf{t}|\mathbf{x},\mathbf{W},\beta) = (\frac{\beta}{2\pi})^{D/2} exp\{-\frac{\beta}{2}||\mathbf{y}(\mathbf{x};\mathbf{W}) - \mathbf{t}||^2\}$$

where D is the dimension of the data space and β^{-1} is the variance of the Gaussian distribution.

The distribution in the data space, for a given value of $\mathbf{W}$, is then obtained by integration over the $\mathbf{x}$-distribution

$$p(\mathbf{t}|\mathbf{W},\beta) = \int p(\mathbf{t}|\mathbf{x},\mathbf{W},\beta)p(\mathbf{x})dx.$$

For a given data set $\mathbf{D} = \mathbf{t_1},\ldots,\mathbf{t}_N$ of N data points, the parameter matrix $\mathbf{W}$, and the inverse variance β can be determined using maximum likelihood. The log likelihood function is given by

$$L(\mathbf{W},\beta) = ln \prod_{n=1}^{N} p(t_n|\mathbf{W},\beta).$$

The (prior) distribution $p(\mathbf{x})$ is chosen to be a sum of K delta functions centered on the nodes of a regular grid in the latent space so that the integral over $\mathbf{x}$ can be solved

$$p(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K}\delta(\mathbf{x} - \mathbf{x}_i).$$

The distribution function in the data space now takes the form

$$p(\mathbf{t}|\mathbf{W},\beta) = \frac{1}{K}\sum_{i=1}^{K} p(\mathbf{t}|\mathbf{x}_i,\mathbf{W},\beta)$$

and the log likelihood function becomes

$$L(\mathbf{W},\beta) = \sum_{n=1}^{N} ln\{\frac{1}{K}\sum_{i=1}^{K} p(\mathbf{t}_n|\mathbf{x}_i,\mathbf{W},\beta)\}.$$

This corresponds to a constrained Gaussian mixture model, since the centers of the Gaussians, given by $\mathbf{y}(\mathbf{x}_i,\mathbf{W})$, cannot move independently but are related through the function $\mathbf{y}(\mathbf{x};\mathbf{W})$. If the mapping function $\mathbf{y}(\mathbf{x};\mathbf{W})$ is smooth and continuous, the projected points $\mathbf{y}(\mathbf{x}_i;\mathbf{W})$ will necessarily have a topographic ordering in the data space.

3.3 Optimizing

If a particular parameterized form for $\mathbf{y}(\mathbf{x};\mathbf{W})$ is chosen, which is a differentiable function of $\mathbf{W}$ (for example a feed-forward network with sigmoidal hidden units) then any standard non-linear optimization method, such as conjugate gradients can be used. However, since the model consists of a mixture distribution the EM algorithm is used.

The function is chosen to be given by a generalized linear regression model of the form

$$\mathbf{y}(\mathbf{x};\mathbf{W}) = \mathbf{W}\mathbf{\Phi}(\mathbf{x})$$

where the elements of $\mathbf{\Phi}(\mathbf{x})$ consist of M fixed basis functions $\Phi_j(\mathbf{x})$, and $\mathbf{W}$ is a DxM matrix. Generalized linear regression models possess the same universal approximation capabilities as multi-layer adaptive networks.

The E-step for the GTM is the same as for a general Gaussian mixture model, computing the *responsibilities*,

$$r_{kn} = p(\mathbf{x}_k|\mathbf{t}_n, \mathbf{W}, \beta) = \frac{p(\mathbf{t}_n|\mathbf{x}_k, \mathbf{W}, beta)p(\mathbf{x}_k)}{\sum_{k'} p(\mathbf{t}_n|W, \beta)p(\mathbf{x}_{k'})}$$

assumed by the kth component of the Gaussian mixture for the nth data point, for each possible pair of k and n. Maximizing the complete-data log likelihood in the form of

$$\mathcal{L}_{comp}(\mathbf{W}, \beta) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{kn}(\mathbf{W}_{old}, \beta_{old}) ln\{p(\mathbf{t}_n|\mathbf{x}_k, \mathbf{W}, \beta)\}$$

with respect to $\mathbf{W}$ leads to

$$\sum_{n=1}^{N}\sum_{k=1}^{K} r_{kn}(\mathbf{W}_{old}, \beta_{old})\{\mathbf{W}_{new}\mathbf{\Phi}(\mathbf{x}_i) - \mathbf{t}_n\}\mathbf{\Phi}^T(\mathbf{x}_i) = 0,$$

which can be calculated using standard matrix inversion techniques, and with respect to β leads to

$$\frac{1}{\beta_{new}} = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{kn}(\mathbf{W}_{old}, \beta_{old})||\mathbf{W}_{new}\mathbf{\Phi}(\mathbf{x}_i) - \mathbf{t}_n||^2.$$

The EM algorithm alternates between the E-step, calculating the responsibilities r_{kn}, and the M-step where the new values for the parameters β_{new} and $\mathbf{W}_{new}$ are calculated. Jensen's inequality can be used to show that with each iteration the cost function converges to a local minimum.

4. LIMITATIONS

As any other algorithm SOM and GTM have some limitations, which make them more or less appropriate for certain data mining problems.

4.1 Assumptions and Applicability

Both, GTM and SOM, are designed for unsupervised datamining. No information on existing clusters can be used and therefore other methods should be considered whenever this information is available.

As well as any other datamining tools, they assume that the input data contains the information that should be found. This leads to the non-trivial process of preprocessing and is directly linked to the distance measure used, which is normally the Euclidean distance in most standard implementations of GTM and SOM.

Both algorithms assume a low dimensional non-linear Euclidean manifold in the data space on which the data lies. Finding this manifold with its information on topology and clusters can aid the user in getting an intuitive understanding of the underlining data structure. If there is no such inherent manifold trying to find it will interfere with finding optimal clusters. If no topology information is desired, other methods such as k-means clustering instead of the SOM, a simple Gaussian mixture model with EM instead of GTM, or perhaps alternatives such as Bayesian classification (for example AutoClass [Cheesman and Stutz 1996]), or agglomerative hierarchical clustering methods should be considered. A recent review of data clustering can be found in [Jain et al. 1999].

On the other side if clustering is not the main task but rather projecting the data from a high-dimensional input space to a low-dimensional output space (mul-

tidimensional scaling) other methods such as Sammon's mapping [Sammon 1969] should be considered.

4.2 Result Representation

Visualization is the main advantage of the SOM and GTM algorithms. They provide the user with a intuitive 2-dimensional map of the data which represents the clusters and how they are related to each other (cf. Figures 3, 4). Clusters and their sub-clusters can easily be identified on the maps. Areas with a higher density are magnified while sparse areas occupy only a minimum of space on the map. Hence the user has all information on one 2-dimensional map, be it the top-level clusters or the smallest sub-clusters.

The main difference between the SOM and the GTM regarding their result presentation is that while SOM assigns each data point to exactly one place on the map GTM calculates a distribution for each data point on the map. These distributions might be very interesting for very small datasets or for single data points, but they are difficult to visualize. Hence the mean (arithmetic average) or the mode (maximum) of the distribution can be used for visualization, which leads to the loss of the additional information.

What is not directly represented on the map is the hierarchical structure of the data, which can be obtained using hierarchical agglomerative clustering or for example tree-structured SOMs.

What a trained SOM may look like - when using the MATLAB toolbox[3] and a simple animal[4] dataset - can be seen in Figure 3. In Figure 4 the same dataset was used with the GTM toolbox supplied by Svensen.

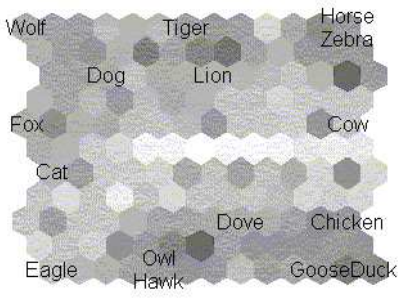

Fig. 3. The SOM trained with a simple animal dataset. The shading represents the clustering tendency (U-matrix): dark areas indicate centers of clusters and light areas the boarders. There is a clear devision between the animals with four legs on the top and the birds at the bottom.

4.3 Accuracy

GTM offers more information about the mapping than the SOM because of it calculates the distribution of single data points. The SOM can be seen as an approximation of the GTM. However it is not easy to visualize this additional information and often leads to simplifications in which this information is lost. One advantage GTM offers due to its statistical background is that the posterior probability of the GTM can be easily used to compare different models. Furthermore, this posterior probability in combination with the EM-algorithm insures that the algorithm will

[3] http://www.cis.hut.fi/projects/somtoolbox
[4] http://student.ifs.tuwien.ac.at/~elias/animals.tar.gz

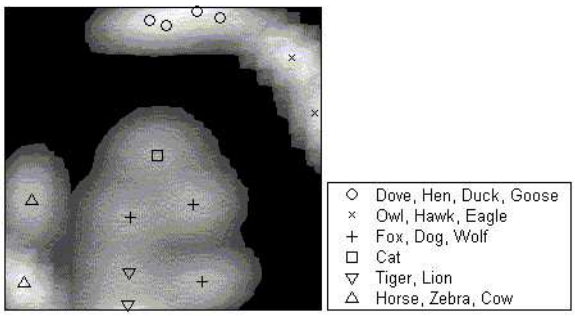

Fig. 4. The GTM trained with the same animal dataset. The shading represents the density function of the dataset: light areas indicate centers of clusters and dark areas the boarders. Again there is a clear division between the animals with four legs and the birds.

converge to a local minimum, while the SOM does not have any function that is minimized and so it is not possible to prove the convergence of the algorithm. (Although there exists a proof for the 1-dimensional SOM and in practice there have never been any convergence problems.)

It is necessary to keep in mind that a higher cluster accuracy can be achieved when using pure clustering algorithms, and multi-dimensional scaling algorithms might achieve better results when projecting data into lower dimensional spaces.

4.4 Parameters

Both algorithms assume hyper-spherical, Gaussians clusters, and provide a non-linear model for the hyperplane. The distance measure is normally the Euclidean. For the user to decide there are two more important parameters for SOM and GTM, which are the flexibility of the hyperplane and how accurate the hyperplane itself should be approximated. While GTM directly adjusts these parameters it is necessary to consider some additional parameters when using the SOM. If the flexibility is chosen too high it will lead to over-fitting while if the hyperplane is too stiff it might not be able to fit the data well enough. The accuracy has direct impact on the computational cost. It is up to the user to decide how much time and storage is available and how accurate the results need to be.

Using GTM the appropriate *number of cycles* can be determined observing the convergence of the cost function. Also for the SOM there have been several proposals for a function that calculates the quality of the mapping, hence determining the correct number of cycles is not a difficult decision.

4.4.1 *SOM.* There is no theoretical foundation for the parameters of the SOM. There is also no clear separation between the parameters which influence how flexible the hyperplane is and how well it should be approximated. But since it has been applied numerous times there is empirical evidence as to which parameters make sense in which context. Yet it might take the user a few tests until the best parameters can be determined.

The *learning rate* influences the ability of the map to learn something new. Normally it is set to decrease exponentially during training. High values in the beginning can be used to initialize the map. If the map has been initialized already for example through principal component analysis then lower values can be used to optimize training time.

The difference between k-means clustering and SOM is that the SOM trains the

neighborhood of a unit as well. This is adjusted with the *neighborhood function and rate*. Normally a Gaussian-like function to determine how strong the neighbors of a unit are influenced is used. During training this function is decreased. At the beginning high values are used to initialize the map and at the end the neighborhood of a unit is reduced to the unit itself, to optimize the cluster quality of the map. The size of the neighborhood influences the flexibility of the map. The smaller the neighborhood of a unit, the bigger can be the difference between units which are close together.

The *map size and shape* defines the number of units and in which way they are arranged. A hexagonal or rectangular grid is normally used with a shape similar to a square since it is easy to visualize. But it would also be possible to arrange the units, for example, on a sphere. The number of units has influence on the accuracy with which the hyperplane is approximated. As mentioned above, the definition of what is the neighborhood of a unit has an influence on the flexibility of the map. Since the neighborhood of a unit is relative to the overall map size it is not possible to adjust the flexibility without considering the number of units on the map.

4.4.2 *GTM.* In contrast to the SOM the GTM is statistically well founded. The parameters can be determined without test-runs. There is a clear separation between the accuracy of the approximation and the flexibility of the hyperplane.

When using the MATLAB implementation the map size and shape are preset to a square shape with a rectangular grid but can be adjusted if desired. The main decision the user is confronted with concerns the *amount of latent points*. This number determines how accurately the hyperplane is approximated.

To define the flexibility of the hyperplane the user can adjust the *number of basis functions* and *their relative width (sigma)*. A limited number of basis functions will necessarily restrict the possible forms the hyperplane can take. The width of the basis functions defines their overlap, which leads to a correlated response that causes the smoothness of the mapping. More or narrower basis functions will allow a more flexible hyperplane, fewer or broader functions will prescribe a smoother hyperplane.

4.5 Computational Cost

According to [Bishop et al. 1998] the GTM algorithm is by a third slower compared to the batch SOM algorithm. This dramatically increases when algorithmic shortcuts are used with the SOM. For example, it is possible to speed up the winner search by remembering the most recent winner for each data point and searching for the winner within its vicinity. Other possible examples are increasing the number of units through interpolation, which can be used for initialization, or using a parallel implementation on a multi-processor architecture.

The computational cost of the SOM is comparable with k-means. SOM can be applied to very large data sets. One practical example is the WebSOM[5]. Currently maps with a million units have been trained on a dataset (abstracts of patents) with about 7 million documents represented by about 70,000 dimensions (which were reduced to 1,000 prior to training). While training took a total of 6 weeks other algorithms would have failed completely. For example Sammon's mapping reaches its limits with datasets with more than about 1,000 vectors.

[5] http://www.cis.hut.fi/WebSOM/

4.6 Summary

The main difference between the SOM and the GTM algorithm is that SOM is faster while GTM is slightly more accurate. Both algorithms can be applied when a mapping is desired that maps datapoints from the high-dimensional dataspace onto a 2-dimensional map while preserving local distances, which in turn is used to visualize the dataset. A typical domain for visualization are text archives. The resulting maps enable the user to identify important topics and understand the relations between them. Visualization can also be used as one of the first steps in data-mining and, in combination with other methods, can be employed in almost any data-mining process.

5. AN ALTERNATIVE APPROACH FOR VISUALIZATION

As mentioned in Section 4.3 a higher cluster accuracy can be achieved when using pure clustering algorithms, and multi-dimensional scaling algorithms might achieve better results when projecting data.

Concerning this problem an interesting approach has been presented in [Flexer 1997; Flexer 1999] where k-means clustering is combined with Sammon's mapping.

First the data is clustered using k-means. The user needs to specify the number of clusters just as the number of units using the SOM need to be specified. Since k-means is not concerned with neighborhoods its clustering results are more accurate compared to the SOM. Then the cluster centers found by k-means are used as input for Sammon's mapping. One advantage of Sammon's mapping in respect to the SOM is that it does not create discrete distances between the clusters. That the cluster centers are not located on a fixed grid gives them the freedom to form any needed shape and in turn can generate more accurate results.

This approach has limitations as well. One is that the *maximum number of clusters* is limited by the Sammon's mapping algorithm and is far below the what the SOM can handle. Practically Sammon's mapping reaches its limits at about 1,000 vectors. But a map with 1,000 units might be more than enough for an average user before oversight is lost. If really necessary it is possible to hierarchically go into further depth.

Another problem might occur when the clusters found by k-means are not of equal size. Sammon's mapping will treat them equally while SOM for example treats clusters with only few a items more flexibly. This problem can easily be solved by adding weights for each cluster, so bigger clusters are more inertial and smaller ones can easily be moved around. The function which Sammon minimizes via steepest descent can be modified to

$$\sum_{j<i}^{N} \gamma_{ij} \frac{(d_{in}(\mathbf{t}_i, \mathbf{t}_j) - d_{out}(\mathbf{x}_i, \mathbf{x}_j))^2}{d_{in}(\mathbf{t}_i, \mathbf{t}_j)}$$

where γ_{ij} can, for example, be the inverse of the sum of data points represented by the two clusters $\mathbf{t}_i$ and $\mathbf{t}_j$, d_{in} denotes the distance between two data points in the input space, d_{out} denotes the distance in the output space. (The factor two is omitted but when added shows the relation to the original function.)

Sammon's mapping does not directly have parameters to adjust the flexibility of the mapping in opposition to SOM and GTM. The algorithm does focus on local distances, but the user cannot influence the rate at which this happens. This could be done for example during training by selecting the data points (cluster centers) not with a uniform distribution but, for example, by choosing pairs which are closer together with a higher probability to increase flexibility. The inability to adjust the

flexibility easily is the biggest drawback of this approach.

Nevertheless this approach demonstrates the potential that lies within rather traditional statistical methods. Combinations of these potentials deserve more detailed analysis and should be considered before applying newly developed algorithms in the process of data-mining.

6. CONCLUSION AND DISCUSSION

Reasons why the SOM is very popular might be because it has an easy to understand algorithm, it is simple to use, and it produces good and intuitive results. Furthermore being a neural network and in turn in some way a model for the human brain makes it attractive. And last but not least the SOM brought lots of visualization into an otherwise quite plain and number oriented process of mining data with a very efficient algorithm with can be run on all common computers. But it is necessary to realize the limitations of the SOM algorithm. Some of them have been addressed by the GTM algorithm and some simply cannot be solved because they are inherent to the model of mapping datapoints from a high dimensional data-space onto a 2-dimensional map while preserving local distances. This in turn can be overcome when the use of SOM or GTM is combined with other data-mining methods such as pure clustering or multi-dimensional scaling techniques.

REFERENCES

BISHOP, C. M., SVENSÉN, M., AND WILLIAMS, C. K. I. 1997. Gtm: A principled alternative to the self-organizing map. In M. C. MOZER, M. I. JORDAN, AND T. PETCHE Eds., *Advances in Neural Information Processing Systems*, Volume 9, pp. 354–360. MIT Press.

BISHOP, C. M., SVENSÉN, M., AND WILLIAMS, C. K. I. 1998. Gtm: The generative topographic mapping. *Neural Computation 10*, 1, 215–235.

CHEESMAN, P. AND STUTZ, J. 1996. Bayesian classification (autoclass): Theory and results. In *Advances in Knowlege Discovery and Data Mining*. AAAI Press/MIT Press.

DEBOECK, G. AND KOHONEN, T. 1998. *Visual Explorations in Finance* (2nd extended ed.). Springer, Berlin, Germany.

DITTENBACH, M., MERKL, D., AND RAUBER, A. 2000. The growing hierarchical self-organizing map. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2000)*.

FLEXER, A. 1997. Limitations of self-organizing maps for vector quantization and multidimensional scaling. In M. C. MOZER, M. I. JORDAN, AND T. PETCHE Eds., *Advances in Neural Information Processing Systems*, Volume 9, pp. 445–451. MIT Press/Bradford Books.

FLEXER, A. 1999. On the use of self-organizing maps for clustering and visualization. Technical Report TR-99-04, Austrian Research Institute for Artificial Intelligence, Vienna.

FRITZKE, B. 1996. Growing self-organizing networks - why? In *Proceedings ot the European Symposium on Artificial Neural Networks (ESANN96)*. Bruges, Belgium.

JAIN, A., M.N., M., AND P.J., F. 1999. Data clustering: A review. *ACM Computing Surveys 31*, 3, 264–323.

KASKI, S., NIKKLIÄ, J., AND KOHONEN, T. 1998. Methods for interpreting a self-organized map in data analysis. In *Proceedings of the 6th European Symposium of Artificial Neural Networks (ESANN98)*, pp. 185–190. Brussels, Belgium.

KOHONEN, T. 1982. Self-organized formation of topologically correct feature maps. *Biol. Cybern. 43*, 59–69.

KOHONEN, T. 1997. *Self-Organizing Maps* (2nd extended ed.). Springer, Berlin.

KOHONEN, T., KASKI, S., LAPPALAINEN, H., AND SALOJÄRVI, J. 1997. The adaptive-subspace self-organizing map (assom). In *Workshop on Self-Organizing Maps (WSOM'97)*, pp. 191–196. Espoo, Finland.

KOHONEN, T., OJA, E., SIMULA, O., VISA, A., AND KANGAS, J. 1996. Engeneering applications of the self-organizing map. In *Proceedings of the IEEE*, Volume 84 of *10*, pp. 1358–1384.

MERKL, D. AND RAUBER, A. 1997. Cluster connections – a visualization technique to reveal cluster boundaries in self-organizing maps. In *Proceedings of the 9th Italian Workshop on Neural Nets (WIRN97)*. Springer.

MERKL, D. AND RAUBER, A. 1998. Finding structure in text archives. In *Proceedings of the 6th European Symposium of Artificial Neural Networks (ESANN98)*.

MIIKKULAINEN, R. 1990. Script recognition with hierarchical feature maps. *Connection Science 2*.

RAUBER, A. 1999. Labelsom: On the labeling of self-organizing maps. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN99)*.

SAMMON, J. W. 1969. A nonlinear mapping for data structure analysis. *IEEE Transactions on Comp. C-18*, 5, 401–409.

SIMULA, O., VASARA, P., VESANTO, J., AND HELMINEN, R. 1999. The self-organizing map in industry analysis. In L. JAIN AND V. VEMURI Eds., *Industrial Applications of Neural Networks*. Washington, DC: CRC Press.

SVENSÉN, M. 1998. *GTM: The Generative Topographic Mapping*. Ph. D. thesis, Aston University, Birmingham, UK.

ULTSCH, A. 1993. Self-organizing neural networks for visualization and classification. In *Information and Classification. Concepts, Methods and Applications*. Springer.