Topographic Processing of Relational Data

Barbara Hammer[1], Alexander Hasenfuss[1], Fabrice Rossi[2], Marc Strickert[3]
1 - Clausthal University of Technology, Department of Informatics
{hammer|hasenfuss}@in.tu-clausthal.de
2 - Projet AxIS, INRIA, Le Chesnay Cedex, France
fabrice.rossi@apiacoa.org
3 - IPK Gatersleben, Pattern Recognition Group
stricker@ipk-gatersleben.de
Keywords: relational data, batch clustering, visualization, SOM, NG

***Abstract*— Recently, batch optimization schemes of the self-organizing map (SOM) and neural gas (NG) have been modified to so-called median variants which allow a transfer of these methods to arbitrary distance measures beyond the standard euclidean metric. This principle is particularly suitable for complex applications where data are compared by means of problem-specific, possibly discrete metrics such as protein sequences which are compared by FASTA or BLAST. However, median variants do not allow a continuous update of prototype locations and their capacity is thus restricted in particular for small data sets. In this contribution, we consider the relational dual of batch optimization which can be formulated in terms of pairwise distances only such that an application to distance matrices without known euclidean embedding becomes possible. For SOM, a direct visualization of data is given by means of the underlying (euclidean or hyperbolic) lattice structure. For NG, pairwise distances of prototypes can be computed based on a given data matrix only, such that subsequent mapping by means of multidimensional scaling can be applied. These algorithms are evaluated in several experiments.**

1 Introduction

The self organizing map as proposed by Kohonen [17] constitutes one of the most popular methods for visual data inspection and clustering: It is highly flexible and can be easily applied to data of interest. Alternative methods exists such as neural gas as presented by Martinetz [20] which focuses on the aspect of clustering and which can serve as inspection tool in combination with subsequent visualization e.g. using multidimensional scaling [16, 27].

The original online variants of SOM and NG, however, have only been proposed for euclidean data. Extensions to more general situations are inevitable if the euclidean distance measure is not appropriate or not applicable such as the case of discrete data. A variety of extensions of SOM has been proposed to deal with this situation including statistical variants [28], variants which built on generative models [30], and encoder-decoder frameworks [4, 5, 26]. These proposals have in common that the original intuitive version of SOM is not directly obtained in the case of standard euclidean data but modifications are necessary. Recently, batch optimization schemes have been extended to general proximity data by means of the generalized median [3, 18]. This restricts prototype locations to data points, thereby maintaining the standard batch optimization principle as far as possible. A theoretical background for this method as well as a convergence proof is provided in [3]: median variants can be interpreted as consecutive optimization of the cost function of NG resp. SOM with respect to assignments and prototype locations. Thus, median clustering is very similar to standard batch variants since it relies on the same cost function. However, it restricts prototype locations to the data space such that a severe loss of accuracy can be expected in particular for an only sparsely covered input space since no continuous update takes place.

Here we propose relational variants of SOM and NG which allow a continuous update of prototypes also in the case of given distance matrix. These versions optimize the same cost function as the standard batch versions, such that convergence is guaranteed. Relational clustering is quite well known for simple k-means clustering and fuzzy variants thereof [9, 10]. The article [12], for example, provides a reliable (but costly) solution to optimize the cost function of Relational k-means by means of statistical physics. For SOM and NG, online kernelized variants have been proposed e.g. in [23, 32] which are related to relational clustering since every kernel induces a general metric (but not vice versa). However, these proposals do not provide a fast batch optimization scheme and they require similarities instead of dissimilarities. We embed relational clustering into a general framework by means of a cost function such that convergence is guaranteed and extensions such as supervision [8] can be easily integrated.

The focus of this article lies on the usefulness of these methods for inspection and low-dimensional visualization of relational data characterized by pairwise distances. For this purpose, standard SOM with either euclidean or hyperbolic lattice as proposed in [24] can be used. Alternatively, we derive formulas which describe pairwise distances of prototypes in the relational models, such that also NG structures without fixed prior lattice can be embedded in low dimensional euclidean or hyperbolic space [16, 27, 31]. Thereby, depending on the complexity of the data, hyperbolic space might be better suited to cluster and visualize hierarchical structures and close connections of the data [1].

Now, we first introduce standard batch optimization of NG and SOM and extend this models to relational data. We discuss the possibility to incorporate supervision and visualization, and demonstrate the applicability in several experiments from bioinformatics.

Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM 2007)
Published by the Neuroinformatics Group, Bielefeld University, Germany
All contributions to WSOM 2007 are available online at: http://biecoll.ub.uni-bielefeld.de

2 Relational topographic maps

Classical neural topographic maps consider vectorial data $\vec{x} \in \mathbb{R}^n$ which are distributed according to an underlying distribution P in the euclidean plane. The goal of clustering is to distribute prototypes $\vec{w}^i \in \mathbb{R}^n$, $i = 1, \ldots, N$ faithfully among the data. A new data point $\vec{x}$ is assigned to the *winner* $\vec{w}^{I(\vec{x})}$ which is the prototype with smallest distance $\|\vec{w}^{I(\vec{x})} - \vec{x}\|^2$. This clusters the data space into the receptive fields of the prototypes.

Different popular variants of neural clustering have been proposed to learn prototype locations from given training data [17]. Assume the number of prototypes is fixed to N. Neural gas (NG) [20] has been introduced based on the cost function

$$E_{\mathrm{NG}}(\vec{w}) = \frac{1}{2C(\lambda)} \sum_{i=1}^{N} \int h_\lambda(k_i(\vec{x})) \cdot \|\vec{x} - \vec{w}^i\|^2 \, P(d\vec{x})$$

where

$$k_i(\vec{x}) = |\{\vec{w}^j \mid \|\vec{x} - \vec{w}^j\|^2 < \|\vec{x} - \vec{w}^i\|^2\}|$$

is the rank of the prototypes sorted according to the distances, $h_\lambda(t) = \exp(-t/\lambda)$ scales the neighborhood cooperation with neighborhood range $\lambda > 0$, and $C(\lambda)$ is the constant $\sum_{i=1}^{N} h_\lambda(k_i(\vec{x}))$which we neglect in the following for simplicity. Classical NG is optimized online by means of a stochastic gradient descent. For a fixed training set, an alternative fast batch optimization scheme is offered by the following algorithm, which in turn computes ranks, which are treated as hidden variables of the cost function, and optimum prototype locations [3]:

init $\vec{w}^i$
repeat
  compute ranks $k_i(\vec{x}^j) = |\{\vec{w}^k \mid \|\vec{x}^j - \vec{w}^k\|^2 < \|\vec{x}^j - \vec{w}^i\|^2\}|$
  compute new prototype locations $\vec{w}^i = \sum_j h_\lambda(k_i(\vec{x}^j)) \cdot \vec{x}^j / \sum_j h_\lambda(k_i(\vec{x}^j))$

Like k-means, NG can be used as a preprocessing step for data mining and visualization, followed e.g. by subsequent projection methods such as multidimensional scaling.

The self-organizing map (SOM) as proposed by Kohonen uses a fixed (usually low-dimensional and regular) lattice structure which determines the neighborhood cooperation. This restriction can induce topological mismatches if the data topology does not match the prior lattice. However, since often a two-dimensional regular lattice is chosen, this has the benefit that, apart from clustering, a direct visualization of the data results by a representation of the data in the regular lattice space. Thus SOM constitutes a direct data inspection and visualization method. SOM itself does not possess a cost function, but a slight variation thereof does, as proposed by Heskes [11]. The cost function is $E_{\mathrm{SOM}}(\vec{w}) =$

$$\frac{1}{2} \sum_{i=1}^{N} \int \delta_{i,I^*(\vec{x})} \cdot \sum_k h_\lambda(n(i,k)) \|\vec{x} - \vec{w}^k\|^2 \, P(d\vec{x})$$

where $n(i,j)$ denotes the neighborhood structure induced by the lattice and $h_\lambda(t) = \exp(-t/\lambda)$ scales the neighborhood degree by a Gaussian function. Thereby, the index $I^*(\vec{x})$ refers to a slightly altered winner notation: the neuron $I^*(\vec{x})$ becomes winner for $\vec{x}$ for which the average distance

$$\sum_k h_\lambda(n(I^*(\vec{x}),k)) \|\vec{x} - \vec{w}^k\|^2$$

is minimum. Often, neurons are arranged in a graph structure which defines the topology, e.g. a rectangular or hexagonal tessellation of the euclidean plane resp. a hyperbolic grid on the two-dimensional hyperbolic plane, the latter allowing a very dense connection of prototypes with exponentially increasing number of neighbors. In these cases, the function $n(i,j)$ denotes the length of a path connecting the prototypes number i and j in the lattice structure. Original SOM is optimized in an online fashion. For fixed training data, batch optimization is possible by subsequently optimizing assignments and prototype locations:

init
repeat
  compute winner assignments $I^*(\vec{x}^j)$ minimizing $\sum_k h_\lambda(n(I^*(\vec{x}^j),k)) \|\vec{x} - \vec{w}^k\|^2$
  compute new prototypes $\vec{w}^i = \sum_j h_\lambda(n(I^*(\vec{x}^j),i) \cdot \vec{x}^j / \sum_j h_\lambda(n(I^*(\vec{x}^j),i)$

It has been shown in e.g. [3] that these batch optimization schemes converge in a finite number of steps towards a (local) optimum of the cost function, provided the data points are not located at borders of receptive fields of the final prototype locations. In the latter case, convergence can still be guaranteed but the final solution can lie at the border of basins of attraction.

2.1 Relational data

Relational data x^i are not embedded in a vector space, rather, pairwise similarities or dissimilarities are available. Assume training data $x^1, \ldots, x^m$ are given by means of pairwise distances $d_{ij} = d(x^i, x^j)^2$. We assume that this stems from an unknown distance measure such that we can find (possibly high dimensional) euclidean points $\vec{x}^i$ such that $d_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$. Note that this notation includes a possibly nonlinear mapping (feature map) $x^i \mapsto \vec{x}^i$ corresponding to the embedding in a euclidean space. However, this embedding is not known, such that we cannot directly optimize the above cost functions in the embedding space.

Median clustering restricts prototype locations to given data points and determines the prototype locations in each iterative step in such a way that the corresponding part of the cost function (assumed fixed assignments) becomes minimum. These values are determined by extensive search, turning the linear complexity to a quadratic one w.r.t. the number of training data. This procedure has the severe drawback that only discrete adaptation steps can be performed and the result is usually worse compared to standard SOM or NG in the euclidean setting.

Relational learning overcomes this problem. The key observation consists in the fact that optimum prototype locations $\vec{w}^j$ can be expressed as linear combination of data points. Therefore, the unknown distances $\|\vec{x}^j - \vec{w}^i\|^2$ can be expressed in terms of known values d_{ij}.

More precisely, assume there exist points $\vec{x}^j$ such that $d_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$. Assume the prototypes can be expressed in terms of data points $\vec{w}^i = \sum_j \alpha_{ij}\vec{x}^j$ where $\sum_j \alpha_{ij} = 1$. Then

$$\|\vec{w}^i - \vec{x}^j\|^2 = (D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$$

where $D = (d_{ij})_{ij}$ constitutes the distance matrix and $\alpha_i = (\alpha_{ij})_j$ the coefficients.

Because of this fact, we can substitute all terms $\|\vec{x}^j - \vec{w}^i\|^2$ in batch optimization schemes. For optimum solutions we find the equality $\vec{w}^i = \sum_j \alpha_{ij}\vec{x}^j$ for batch NG and batch SOM as introduced above, whereby

(1) $\alpha_{ij} = h_\lambda(k_i(\vec{x}^j))/\sum_j h_\lambda(k_i(\vec{x}^j))$ for NG, and
(2) $\alpha_{ij} = h_\lambda(n(I^*(\vec{x}^j), i))/\sum_j h_\lambda(n(I^*(\vec{x}^j), i)$ for SOM.

This allows to reformulate the batch optimization schemes in terms of relational data. We obtain the algorithm

init α_{ij} with $\sum_j \alpha_{ij} = 1$
repeat
  compute $\|\vec{x}^j - \vec{w}^i\|^2$ as $(D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$
  compute optimum assignments based on these values
    $\tilde{\alpha}_{ij} = h_\lambda(k_i(\vec{x}^j))$ (for NG)
    $\tilde{\alpha}_{ij} = h_\lambda(n(I^*(\vec{x}^j), i))$ (for SOM)
  compute $\alpha_{ij} = \tilde{\alpha}_{ij}/\sum_j \tilde{\alpha}_{ij}$

Hence, prototype locations are computed only indirectly by means of the coefficients α_{ij}. Initialization can be done e.g. setting initial prototype locations to random data points, which is realized by a random selection of N rows from the given distance matrix.

Given a new data point x which can isometrically be embedded in euclidean space as $\vec{x}$, and pairwise distances $d_j = d(x, x^j)^2$ corresponding to the distance from x^j, the winner can be determined by using the equality

$$\|\vec{x} - \vec{w}^i\|^2 = (D(x)^t \cdot \alpha_i) - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$$

where $D(x)$ denotes the vector of distances $D(x) = (d_j)_j = (d(x, x^j)^2)_j$.

The quantization error can be expressed in terms of the given values d_{ij} by substituting $\|\vec{x}^j - \vec{w}^i\|^2$ by $(D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$. Interestingly, using the formula for optimum assignments of batch optimization, one can also derive relational dual cost functions for the algorithms. The relational dual of NG is

$$\sum_i \frac{\sum_{ll'} h_\lambda(k_i(\vec{x}^l))h_\lambda(k_i(\vec{x}^{l'}))d_{ll'}}{4\sum_l h_\lambda(k_i(\vec{x}^l))}.$$

For SOM, we obtain

$$\sum_i \frac{\sum_{ll'} h_\lambda(n(I^*(\vec{x}^l), i))h_\lambda(n(I^*(\vec{x}^{l'}), i))d_{ll'}}{4\sum_l h_\lambda n(I^*(\vec{x}^l), i)}.$$

Note that this relational learning gives exactly the same results as standard batch optimization provided the given relations stem from a euclidean metric. Hence convergence is guaranteed in this case since it holds for the standard batch versions. If the given distance matrix does not come from a euclidean metric, this equality does no longer hold and the terms $(D \cdot \alpha_i)_j - 1/2 \cdot \alpha_i^t \cdot D \cdot \alpha_i$ can become negative. In this case, one can correct the distance matrix by the γ-spread transform $D_\gamma = D + \gamma(\mathbf{1} - \mathbf{I})$ for sufficiently large γ where $\mathbf{1}$ equals 1 for each entry and $\mathbf{I}$ is the identity. Alternatively, one can directly apply the optimization scheme since it converges also for general symmetric and nonsingular matrix D towards a value of the dual cost function as shown in [7] for relational NG.

2.2 Supervision

The possibility to include further information, if available, is very important to get meaningful results for unsupervised learning. This can help to prevent the 'garbage in - garbage out' problem of unsupervised learning, as discussed e.g. in [14, 15]. We assume that additional label information is available which should be accounted for. Thereby, labels are embedded in $\mathbb{R}^d$. We assume that the label attached to x^j is denoted by $\vec{y}^j$. We equip a prototype w^i with a label $\vec{Y}^i \in \mathbb{R}^d$ which is adapted during learning. For the euclidean case, the basic idea consists in a substitution of the standard euclidean distance $\|\vec{x}^j - \vec{w}^i\|^2$ by a mixture

$$(1-\beta) \cdot \|\vec{x}^j - \vec{w}^i\|^2 + \beta \cdot \|\vec{y}^j - \vec{Y}^i\|^2$$

which takes the similarity of labels into account. $\beta \in [0, 1]$ controls the influence of the labels. This procedure has been proposed in [8] for euclidean clustering. One can use the same principles for relational clustering.

For discrete euclidean settings $\vec{x}^1, \ldots, \vec{x}^m$ cost functions and related batch optimization is as follows: $E_{\mathrm{NG}}(\vec{w}, \vec{Y}) =$

$$\sum_{ij} h_\lambda(k_i(\vec{x}^j)) \cdot \Big((1-\beta) \cdot \|\vec{x}^j - \vec{w}^i\|^2 + \beta \cdot \|\vec{y}^j - \vec{Y}^i\|^2\Big)$$

where $k_i(\vec{x}^j)$ denotes the rank of neuron i measured according to the distances $(1-\beta) \cdot \|\vec{x}^j - \vec{w}^i\|^2 + \beta \cdot \|\vec{y}^j - \vec{Y}^i\|^2$. This change in the computation of the rank is accompanied by the adaptation $\vec{Y}^i = \sum_j h_\lambda(\vec{x}^j)\vec{y}^j / \sum_j h_\lambda(\vec{x}^j)$ for the prototype labels for batch optimization In the same way, the cost function of SOM becomes

$$E_{\mathrm{SOM}}(\vec{w}, \vec{Y}) = \delta_{i, I^*(\vec{x}^j)} \sum_k h_\lambda(n(i,k)) \Big((1-\beta) \cdot \|\vec{x}^j - \vec{w}^k\|^2 + \beta \cdot \|\vec{y}^j - \vec{Y}^k\|^2\Big)$$

where $I^*(\vec{x}^j)$ denotes the generalization of the winner notation proposed by Heskes to the supervised setting, i.e. it minimizes $\sum_k h_\lambda(n(I^*\vec{x}^j), k))$ $\big((1-\beta) \cdot \|\vec{x}^j - \vec{w}^k\|^2 + \beta \cdot \|\vec{y}^j - \vec{Y}^k\|^2\Big)$. Batch optimization uses this winner and extends the updates by $\vec{Y}^i = \sum_j h_\lambda(n(i, I^*(\vec{x}^j))\vec{y}^j / \sum_j h_\lambda(n(i, I^*(\vec{x}^j))$. It has been shown in [8] that these procedures converge in a finite number of steps.

Relational learning becomes possible by substituting the distances $\|\vec{x}^j - \vec{w}^i\|^2$ using the identity $\vec{w}^i = \sum \alpha_{ij}\vec{x}^j$ for optimum assignments which still holds for these extensions. The same computation as beforehand yields to the algorithm for dissimilarity data:

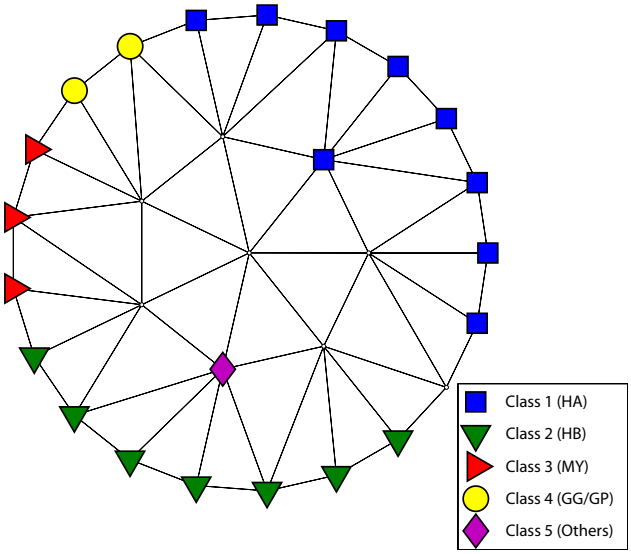

Figure 1: Mapping of the non-euclidean Protein dataset by a Relational SOM with hyperbolic grid structure.

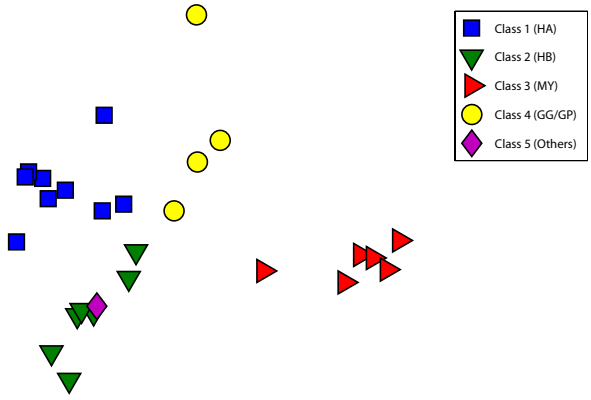

Figure 2: Mapping of the non-euclidean Protein dataset by Relational BNG with non-metric multidimensional scaling.

init α_{ij} with $\sum_j \alpha_{ij} = 1$
repeat
  compute the distances as $(1-\beta)\cdot$
    $((D\cdot\alpha_i)_j - 1/2\cdot\alpha_i^t D\alpha_i) + \beta\cdot\|\vec{Y}^i - \vec{y}^j\|^2$
  compute assignments $\tilde{\alpha}_{ij}$ based on these values
  compute $\alpha_{ij} = \tilde{\alpha}_{ij}/\sum_j \tilde{\alpha}_{ij}$
  compute prototype labels $\vec{Y}^i = \sum_j \alpha_{ij}\vec{y}^j$

Since this version is identical to the euclidean version for an euclidean distance matrix this procedure converges in a finite number of steps. Note that, for vanishing neighborhood size, the final prototype labels coincide with the averaged label taken over the receptive field of a prototype. For rapid learning, one can improve the classification result by setting the prototype labels to the averaged label of the receptive fields after training. Note that, still, the prototype locations are affected by the label information, unlike a pure unsupervised learning with posterior labeling. For the supervised setting, the prototypes are forced to follow the borders given by the class information.

2.3 Visualization

As discussed before, data can be clustered using trained prototypes. But it is not obvious whether we are able to create a visualization of such extracted information. For low-dimensional euclidean SOM, a direct visualization is given by an embedding of the data points to the respective positions of their winner in the lattice space. Also the hyperbolic SOM allows an embedding of the points by means of the Poincaré disk model [1] which embeds the hyperbolic space non-isometrically into the unit disk such that the focus of attention is put onto the point which is mapped into the center of the disk and an overall fish-eye effect results. Moving this focus allows to browse through the map.

For NG no direct visualization is given, but it can be easily reached by a subsequent embedding of the prototypes into the two-dimensional euclidean plane by means of distance preserving projection techniques of the prototypes such as multidimensional scaling. Given pairwise distances $\delta_{ij} := \|\vec{w}^i - \vec{w}^j\|$ of the prototypes (possibly nonlinearly preprocessed, i.e. $\delta_{ij} = f(\|\vec{w}^i - \vec{w}^j\|)$ where f is an appropriate weighting function), this model finds two-dimensional projections $p(\vec{w}^i)$ with pairwise distances $\Delta_{ij} := \|p(\vec{w}^i) - p(\vec{w}^j)\|$ such that the stress-function

$$\left(\frac{\sum_{i<j}(\delta_{ij}-\Delta_{ij})^2}{\sum_{i<j}\Delta_{ij}^2}\right)^2$$

or a similar objective function is minimized.

To apply these techniques, the pairwise distance of prototypes needs to be computed. As beforehand, we assume the identity $\vec{w}^i = \sum_l \alpha_{il}\vec{x}^l$ for optimum prototypes $\vec{w}^i$. We assume that pairwise distances $d_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$ are given, accumulated in a matrix D as beforehand. Then we find

$$\|\vec{w}^i - \vec{w}^j\|^2 = \alpha_j^t D\alpha_i - \frac{1}{2}\alpha_j^t D\alpha_j - \frac{1}{2}\alpha_i^t D\alpha_i$$

where α_i denotes the vector $(\alpha_{il})_l$.

3 Experiments

Protein Classification

The evolutionary distance of 226 globin proteins is determined by alignment as described in [21]. These samples originate from different protein families: hemoglobin-α, hemoglobin-β, myoglobin, etc. Here, we distinguish five classes as proposed in [6]: HA, HB, MY, GG/GP, and others. Table 1 shows the class distribution of the dataset.

For training we use 29 neurons for Relational Batch NG and Relational Hyperbolic SOM, and a 5x5 grid for standard Relational SOM, respectively. The number of neurons is derived from the hyperbolic grid of depth three (cf. figure 1). The neighborhood range is annealed starting from $N/2$ to 0, N being the number of neurons, in all experiments. The results reported in Table 2 are gained from repeated

Class No.	Count	Percentage
HA	72	31.86%
HB	72	31.86%
MY	39	17.26%
GG/GP	30	13.27%
Others	13	5.75%

Table 1: Class Statistics of the Protein Dataset

10-fold stratified cross-validation averaged over 100 repetitions and 150 epochs per run. Supervision is included in the cost function with mixing parameter 0.5.

Unlike the results reported in [6] for SVM which use one-versus-rest encoding, our setting gives one integrated clustering model. Depending on the choice of the kernel, [6] reports errors which approximately add up to 4% for the leave-one-out error. This result, however, is not comparable to our results due to the different error measure. A 1-nearest neighbor classifier yields an accuracy 91.6 for our setting (k-nearest neighbor for larger k is worse; [6]) which is comparable to our results. Thereby, continuous updates improve the results found by median clustering by 3%.

The projections of a Relational SOM with hyperbolic grid structure and of Relational BNG with non-metric multidimensional scaling using Kruskal's normalized stress1 criterion are shown in figure 1 and 2. The neurons are depicted according to majority vote. Obviously, the neurons arrange according to the associated class and a very clear two-dimensional representation of the data set is obtained.

Chromosome Images

The Copenhagen chromosomes database is a benchmark from cytogenetics [19]. A set of 4200 human nuclear chromosomes from 22 classes (the X resp. Y sex chromosome is not considered) are represented by the grey levels of their images and transferred to strings representing the profile of the chromosome by the thickness of their silhouettes. The edit distance is a typical distance measure for two strings of different length, as described in [13, 22]. In our application, distances of two strings are computed using the standard edit distance whereby substitution costs are given by the signed difference of the entries and insertion/deletion costs are given by 4.5 [22].

The algorithms are tested in repeated 2-fold stratified cross-validation using 85 neurons for Relational BNG and Relational HSOM (corresponding two three rings), and a 9x9 grid for the standard Relational SOM. The results

	Median Batch NG	Relational Batch NG	Standard Relational SOM	Relational Hyperbolic SOM
Accuracy Protein Data set				
Mean	89.5	92.4	91.5	91.5
StdDev	1.0	0.9	0.8	1.2
Accuracy Copenhagen Chromosome Database				
Mean	88.8	90.7	89.9	89.4
StdDev	1.2	0.5	0.6	0.7

Table 2: Classification accuracy on the Protein Data Set and Copenhagen Chromosome Database, respectively, for posterior labeling.

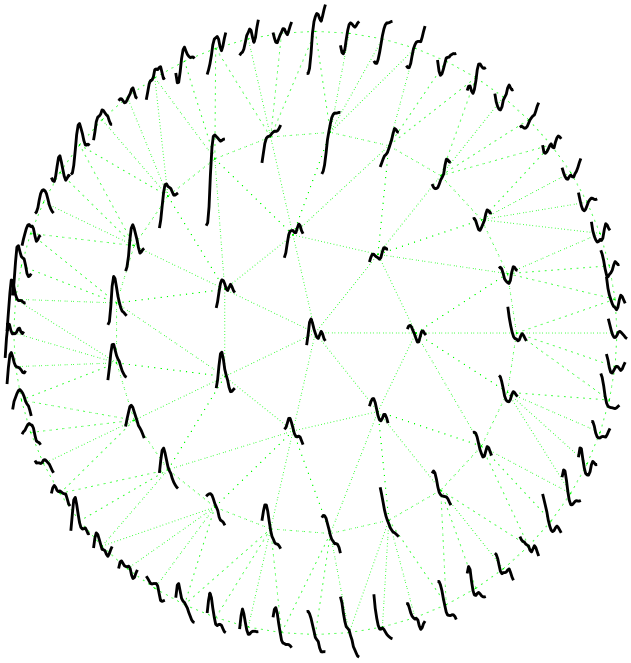

Figure 3: Visualization of the macroarray dataset by Relational SOM with with hyperbolic grid structure.

presented are the mean accuracy over 10 repetitions per method and 100 epochs per run (cf. [3]). Supervision is incorporated using the mixing parameter 0.9. Again, an improvement of 2% can be observed.

Macroarray Data

Barley is an important crop plant, thus, gene expression analysis of the temporal development of barley seeds is an important issue for the derivation of key metabolic processes during different stages of growth. Extensive gene expression measurements at 14 time points from day zero after flowering to day 26 in steps of two days were carried out using cDNA macroarray technology. High signal to noise ratios and high reproducibility between two independently taken experimental series led to a selection of 4824 genes out of 11786 genes available on the 12k macroarrays. Thus, a data matrix of 4824 (genes) by 14 (time points) is considered. As common in gene expression analysis, $\log_2$-transformed final expression values are considered. Coexpression analysis of the available 4824 gene expression time series is required to identify groups of commonly regulated genes that may have temporal impact on each other. Such a clustering helps to extract candidate genes responsible for triggering later events like, for example, the influence of cell wall degradation factors for lateral tissue nutrition or subsequent starch accumulation processes. Fig. 3 shows the arrangement which is obtained when training a HSOM with 85 neurons for 150 epochs on these data, thereby using a transformation of Pearson correlation of the expression patterns as distance measure which better accounts for the overall principled shape as described in [29]. Obviously, the map organizes the data according to the evolution of up- and down-regulation of the genes over time. Due to the hyperbolic structure, the map clearly sep-

arates opposite shapes and preserves the data topology.

4 Conclusions

The extension of SOM and NG to relational data offers a very intuitive and simple possibility to project data which are given by a distance matrix only into a low-dimensional space, thereby preserving the data topology. The method is directly based on the cost function of SOM and NG, respectively, thus opening the way towards further extensions such as supervision as demonstrated in our experiments or hyperbolic MDS as presented for euclidean data in [31]. Unlike alternative proposals such as [5, 12], relational visualization maintains the simplicity of original SOM and NG. The only drawback of the proposed method is given by the complexity of training, which scales with m^2, m being the number of data, instead of m such as original SOM and NG. This complexity can be drastically reduced using either exact methods such as introduced in the article [2], or low-dimensional approximations of the vectors which encode the prototypes as proposed in the approach [25].

References

[1] James W. Anderson (2005), Hyperbolic Geometry, second edition, Springer.

[2] B. Conan-Guez, F. Rossi, and A. El Golli (2006), Fast Algorithm and Implementation of Dissimilarity Self-Organizing Maps, *Neural Networks* 19:855-863.

[3] M. Cottrell, B. Hammer, A. Hasenfuss, and T. Villmann (2006), Batch and median neural gas, *Neural Networks* 19:762-771.

[4] T. Graepel, R. Herbrich, P. Bollmann-Sdorra and K. Obermayer (1999), Classification on pairwise proximity data, In M. I. Jordan, M. J. Kearns, and S. A. Solla (Eds.), NIPS, vol. 11, MIT Press, p. 438-444.

[5] T. Graepel and K. Obermayer (1999), A stochastic self-organizing map for proximity data, *Neural Computation* 11:139-155.

[6] B. Haasdonk and C. Bahlmann (2004), Learning with distance substitution kernels, in *Pattern Recognition - Proc. of the 26th DAGM Symposium*.

[7] B. Hammer, A. Hasenfuss (2007), Relational neural gas, to appear at *KI 2007*.

[8] B. Hammer, A. Hasenfuss, F.-M. Schleif, and T. Villmann (2006), Supervised batch neural gas, In *Proceedings of Conference Artificial Neural Networks in Pattern Recognition (ANNPR)*, F. Schwenker (ed.), Springer, pages 33-45.

[9] R. J. Hathaway and J. C. Bezdek. Nerf c-means: Non-euclidean relational fuzzy clustering. *Pattern Recognition* 27(3):429-437, 1994.

[10] R. J. Hathaway, J. W. Davenport, and J. C. Bezdek. Relational duals of the c-means algorithms. *Pattern Recognition* 22:205-212, 1989.

[11] T. Heskes (2001). Self-organizing maps, vector quantization, and mixture modeling. *IEEE Transactions on Neural Networks* 12:1299-1305.

[12] T. Hofmann and J. Buhmann Multidimensional Scaling and Data Clustering (1995), in *Advances in Neural Information Processing Systems* 7:459-466.

[13] A. Juan and E. Vidal (2000), On the use of normalized edit distances and an efficient k-NN search technique (k-AESA) for fast and accurate string classification, in *ICPR 2000*, vol.2, p. 680-683.

[14] S. Kaski, J. Nikkilä, E. Savia, and C. Roos (2005), Discriminative clustering of yeast stress response, In *Bioinformatics using Computational Intelligence Paradigms*, U. Seiffert, L. Jain, and P. Schweizer (eds.), pages 75-92, Springer.

[15] S. Kaski, J. Nikkilä, M. Oja, J. Venna, P. Törönen, and E. Castren (2003), Trustworthiness and metrics in visualizing similarity of gene expression, *BMC Bioinformatics*, 4:48.

[16] J. B. Kruskal (1964), Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika* 29:1-27.

[17] T. Kohonen (1982),Self-Organized formation of topologically correct feature maps, *Biological Cybernetics*, 43:59-69.

[18] T. Kohonen and P. Somervuo (2002), How to make large self-organizing maps for nonvectorial data, *Neural Networks* 15:945-952.

[19] C. Lundsteen, J. Phillip, and E. Granum (1980), Quantitative analysis of 6985 digitized trypsin G-banded human metaphase chromosomes, *Clinical Genetics* **18**:355-370.

[20] T. Martinetz, S.G. Berkovich, and K.J. Schulten (1993), 'Neural-gas' network for vector quantization and its application to time-series prediction, *IEEE Transactions on Neural Networks* 4:558-569.

[21] H. Mevissen and M. Vingron (1996), Quantifying the local reliability of a sequence alignment, *Protein Engineering* **9:**127-132.

[22] M. Neuhaus and H. Bunke (2006), Edit distance based kernel functions for structural pattern classification *Pattern Recognition* 39(10):1852-1863.

[23] A.K. Qin and P.N. Suganthan (2004), Kernel neural gas algorithms with application to cluster analysis, *ICPR 2004* vol.4, pp.617-620.

[24] H. Ritter (1999), Self-organizing Maps in non-euclidean Spaces, *Kohonen Maps*, 97-108, Eds.: E. Oja and S. Kaski.

[25] F. Rossi, A. Hasenfuss, B. Hammer, Accelerating relational clustering algorithms with sparse prototype representation, submitted to WSOM'07.

[26] A. Saalbach, T. Twellmann, A. Wismüller, J. Ontrup, H. Ritter, T. W. Nattkemper (2005), A Hyperbolic Topographic Mapping for Proximity Data, *Proc. of the IASTED International Conference on Artifical Intelligence and Applications*, 106-111, ACTA Press.

[27] J. W. Sammon Jr. (1969), A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers* 18:401-409.

[28] S. Seo and K. Obermayer (2004), Self-organizing maps and clustering methods for matrix data, *Neural Networks* 17:1211-1230.

[29] M. Strickert, U. Seiffert, N. Sreenivasulu, W. Weschke, T. Villmann, B. Hammer (2006), Generalized Relevance LVQ (GRLVQ) with Correlation Measures for Gene Expression Analysis, *Neurocomputing 69*: 651-659.

[30] P. Tino, A. Kaban, and Y. Sun (2004), A generative probabilistic approach to visualizing sets of symbolic sequences. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD-2004*, (eds) R. Kohavi, J. Gehrke, W. DuMouchel, J. Ghosh. pp. 701-706, ACM Press.

[31] J. Walter (2004), H-MDS: a new approach for interactive visualization with multidimensional scaling in the hyperbolic space. *Information Systems*, 29(4):273-292.

[32] H. Yin (2006), On the equivalence between kernel self-organising maps and self-organising mixture density network, *Neural Networks* 19(6):780-784.