TU Clausthal

Clausthal University of Technology

Topographic Mapping of Large Dissimilarity Data Sets

Barbara Hammer, Alexander Hasenfuss

IfI Technical Report Series IfI-10-01

Impressum

Publisher: Institut für Informatik, Technische Universität Clausthal
Julius-Albert Str. 4, 38678 Clausthal-Zellerfeld, Germany
Editor of the series: Jürgen Dix
Technical editor: Michael Köster
Contact: michael.koester@tu-clausthal.de
URL: http://www.in.tu-clausthal.de/forschung/technical-reports/
ISSN: 1860-8477

The IfI Review Board

Topographic Mapping of Large Dissimilarity Data Sets

Barbara Hammer, Alexander Hasenfuss

B. Hammer, A. Hasenfuss, Clausthal University of Technology, Germany, e-mail: {hammer|hasenfuss@in.tu-clausthal.de}

Abstract

Topographic maps such as the self organizing map (SOM) or neural gas (NG) constitute powerful data mining techniques which allow to simultaneously cluster data and infer its topological structure, such that additional features, e.g. browsing, become available. Both methods have been introduced for vectorial data sets, i.e. they require a classical feature encoding of information. Often, data are available in the form of pairwise distances only, such as e.g. arise from a kernel matrix, a graph, or some general dissimilarity measure. In such cases, NG and SOM cannot be applied directly. In this contribution, we introduce relational topographic maps as an extension of relational clustering algorithms which offer prototype-based representations of dissimilarity data, to incorporate neighborhood structure. These methods are equivalent to the standard (vectorial) techniques, if an Euclidean embedding exists, while preventing the necessity to explicitly compute such an embedding. Extending these techniques for the general case of non-Euclidean dissimilarities, an interpretation of relational clustering as clustering in pseudo-Euclidean space becomes possible. We compare the methods to well-known clustering methods for proximity data based on deterministic annealing and discuss in how far convergence can be guaranteed in the general case. Relational clustering is quadratic in the number of data points which makes the algorithms infeasible for huge data sets. We propose an approximate patch-version of relational clustering which runs in linear time. The effectivity of the methods is demonstrated in a number of examples.

Keywords:Clustering methods, self-organizing feature maps, neural gas, proximity data, large data sets

1 Introduction

Topographic maps such as the self-organizing map (SOM) constitute a valuable tool for robust data inspection and data visualization which has been

applied in diverse areas such as telecommunication, robotics, bioinformatics, business, etc. [37, 68]. A particular strength of SOM lies in the fact that it offers simultaneous data clustering, visualization, topological inference, and compression of data by means of prototypes such that diverse functionalities can be realized on top of SOM. Alternative methods such as neural gas (NG) [46] provide an efficient clustering and topographic mapping of data without fixing a prior lattice. This way, subsequent visualization such as multidimensional scaling [39] can readily be applied, whereby no prior restriction of a fixed lattice structure as for SOM is necessary and the risk of topographic errors is minimized. For NG, an optimum (nonregular) data topology is induced such that browsing in a neighborhood becomes directly possible [47]. A very elegant fundamental treatment of vector quantization and extensions such as SOM and NG has been presented in the work [32] based on information theoretic principles as introduced in [43]. In this framework, vector quantization is interpreted as encoding mechanism with limited resources, where SOM can be derived as robust model if channel noise is present, whereas NG accounts for the situation that certain channels are not available e.g. due to varying bandwidth. This also gives some hints in which situations the application of SOM or NG, respectively, is advisable from a theoretical model of the data, besides providing additional functionlity compared to simple clustering such as k-means due to the additional (fixed or data-optimum, respectively) neighborhood structure. Interestingly, as presented in [32], these approaches can be combined to yield models which are robust with respect to different types of noise.

Original SOM and NG, however, have been proposed for vectorial data such that their application is restricted to Euclidean space. In the last years, a variety of extensions of these methods has been proposed to deal with more general data structures. This accounts for the fact that more general metrics have to be used for complex data such as microarray data or DNA sequences. Further it might be the case that data are not embedded in a vector space at all, rather, pairwise similarities or dissimilarities are available.

Several extensions of classical SOM and NG to more general data have been proposed: a statistical interpretation of SOM as considered in [18, 29, 63, 64] allows to change the generative model to alternative general data models. The resulting approaches are very flexible but also computationally quite demanding, such that proper initialization and metaheuristics (e.g. deterministic annealing) become necessary when optimizing statistical models. For specific data structures such as time series or recursive structures, recursive models have been proposed as reviewed e.g. in the articles [4, 24]. However, these models are restricted to recursive data structures with Euclidean constituents. Online variants of SOM and NG have been extended to general kernels e.g. in the approaches presented in [57, 67]. However, these versions have been derived for (slow) online adaptation only.

The approaches [38] provide a fairly general method for large scale appli-

cation of SOM to nonvectorial data: it is assumed that pairwise similarities of data points are available. Then the batch optimization scheme of SOM can be generalized by means of the generalized median to a visualization tool for general similarity data. Thereby, prototype locations are restricted to data points. This method has been extended to NG in [13] together with a general proof of the convergence of median versions of clustering. Further developments concern the efficiency of the computation [12] and the integration of prior information if available to achieve meaningful visualization and clustering [20, 21, 65].

Median clustering has the benefit that it builds directly on the derivation of SOM and NG from a cost function. Thus, the resulting algorithms share the simplicity of batch NG and SOM, its mathematical background and convergence, as well as the flexibility to model additional information by means of an extension of the cost function. However, for median versions, prototype locations are restricted to the set of given training data which constitutes a severe restriction in particular for small data sets. Therefore, extensions which allow a smooth adaptation of prototypes have been proposed e.g. in [22]. In this approach, a weighting scheme is introduced for the points which represents virtual prototype in the space spanned by the training data. This model has the drawback that it is not an extension of the standard Euclidean version.

Here, we use an alternative way to extend NG to relational data given by pairwise dissimilarities, which is similar to the relational dual of fuzzy clustering as derived in [27, 28] and which directly builds on fundamental work in the context of relational clustering as introduced in [41, 55]. For a given dissimilarity matrix which stems from a (possibly high-dimensional and unknown) Euclidean space, it is possible to derive the relational dual of topographic map formation which expresses the relevant quantities in terms of the given matrix and which leads to a learning scheme similar to standard batch optimization. This scheme provides identical results as the standard Euclidean version if an embedding of the given data points is known. In particular, it possesses the same convergence properties as the standard variants, thereby restricting the computation to known quantities which do not rely on an explicit embedding. Since these relational variants rely on the same cost function, extensions to additional label information or magnification control [20, 21, 23] become readily available.

The methods can directly be applied to every possibly non-Euclidean dissimilarity matrix and, as we will see in a variety of experiments, they result in a good performance in practical applications. The theory behind the case of general dissimilarity data, however, is less clear. We will show that a simple shift of the dissimilarity matrix as proposed in [41] which makes data Euclidean and which does not affect the location of the optima of the dual cost function, can severely affect the underlying numeric. As an alternative, we will link the proposed algorithm to clustering in pseudo-Euclidean space,

such that an intuitive interpretation of the algorithm becomes possible also in the non-Euclidean setting. However, we show by counterexample that the algorithm need no longer converge to a fixed point of the dual cost function - albeit this behavior has not been observed by us in a single real application. We show that popular alternatives such as deterministic annealing for pairwise data clustering or SOM share this property, i.e. counterexamples which show possible divergence can also be found for these two well-known clustering algorithms. We argue that relational neural gas is in fact related to popular deterministic annealing variants in the sense that the latter can be derived as deterministic annealing in pseudo-Euclidean space. This provides a direct interpretation of these alternatives in terms of relational prototypes, i.e. inspection of the results becomes possible this way, and it explains why relational clustering shows remarkable results in practice which are comparable to results obtained by deterministic annealing, while consuming less training time.

Relational clustering as well as its deterministic annealing counterparts display squared complexity according to the size of the dissimilarity matrix. This makes the algorithms unsuitable for large data sets. Based on intuitive and powerful extensions of classical k-means and NG to large data sets by means of patch clustering [1, 15, 8], we propose an approximation of the algorithms which can work in constant memory and linear time, i.e. it is suited for large data sets. While we exemplarily test the results for patch relational NG clustering, the principled method can successfully be applied to every clustering scheme which relies on relational prototypes, i.e. a direct transfer of the method to relational SOM and deterministic annealing variants of relational clustering become possible.

Now, we first introduce batch learning algorithms for neural gas based on a cost function. Then we focus on a dissimilarity matrix which can be embedded in Euclidean space and we derive the respective relational dual resulting in a dual cost function and batch optimization schemes for the case of a given dissimilarity matrix of data. For the general non-Euclidean setting, we discuss the connection to an embedding in pseudo-Euclidean space. Based on this connection, a relation to well-established deterministic annealing schemes become possible. To make the algorithms suitable for large data sets, an approximation of prototypes is introduced which allows to process data subsequently in patches, using constant memory and linear time only. The efficiency of relational clustering is demonstrated in a couple of benchmark situations as well as an application for a text clustering task which involves almost 200.000 articles.

2 Neural Gas

Neural clustering and topographic maps constitute effective methods for data clustering, inspection, and preprocessing. Classical variants deal with vectorial data $\vec{x} \in \mathbb{R}^n$ which are distributed according to an underlying distribution P in the Euclidean space. The goal of prototype-based clustering algorithms is to distribute prototypes $\vec{w}^i \in \mathbb{R}^n$, $i = 1, \ldots, k$ among the data such that they represent the data as accurately as possible. A new data point $\vec{x}$ is assigned to the *winner* $I(\vec{x})$ which refers to the prototype with smallest distance $\|\vec{w}^{I(\vec{x})} - \vec{x}\|^2$. This separates 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 [37]. Assume the number of prototypes is fixed to k. Simple k-means directly optimizes the *quantization error*

$$E_{\mathrm{k-means}}(\vec{w}) = \frac{1}{2} \sum_{i=1}^{k} \int \delta_{i,I(\vec{x})} \cdot \|\vec{x} - \vec{w}^i\|^2 \, P(d\vec{x})$$

where $\delta_{i,I(\vec{x})}$ with Kronecker δ indicates the winner neuron for $\vec{x}$. Unlike k-means, neural gas (NG) [46] and the self organizing map (SOM) [37] incorporate the neighborhood of a neuron for adaptation. The cost function of NG is given by

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

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 and $h_\lambda(t) = \exp(-t/\lambda)$ scales the neighborhood cooperation with neighborhood range $\lambda > 0$.

The SOM itself does not possess a cost function, but a slight variation thereof as proposed e.g. by Heskes [29]:

$$E_{\mathrm{SOM}}(\vec{w}) = \frac{1}{2} \sum_{i=1}^{k} \int \delta_{i,I^*(\vec{x})} \cdot \sum_{l=1}^{k} h_\lambda(\mathrm{nd}(i,l)) \cdot \|\vec{x} - \vec{w}^l\|^2 \, P(d\vec{x})$$

where $I^*(\vec{x})$ denotes the neuron with smallest averaged distance $\sum_{l=1}^{k} h_\lambda(\mathrm{nd}(i,l)) \cdot \|\vec{x} - \vec{w}^l\|^2$ and $\mathrm{nd}(i,l)$ denotes a priorly chosen neighborhood structure of neurons, often induced by a low dimensional lattice structure.

The incorporation of a neighborhood structure into SOM and NG has several beneficial effects: additional functionality is achieved this way, since the topological structure of the data is respected by the neurons, and browsing and, in the case of SOM, visualization become possible. It has been shown

Algorithm 1: Batch NG

input

data $\{\vec{x}^1, \ldots, \vec{x}^m\} \subset \mathbb{R}^n$;

begin

init $\vec{w}^i$ randomly;

repeat

set $k_{ij} := |\{\vec{w}^l \mid \|\vec{x}^j - \vec{w}^l\|^2 < \|\vec{x}^j - \vec{w}^i\|^2\}|$;

set $\vec{w}^i := \sum_j h_\lambda(k_{ij})\vec{x}^j / \sum_j h_\lambda(k_{ij})$;

until convergence;

return $\vec{w}^i$;

end.

experimentally that, by neigbhorhood integration, a method which is widely insensitive to initialization can be achieved [45]. In the fundamental work [32, 43], a theoretical justification for this finding is given by linking NG and SOM, respectively, to information theoretical concepts in the context of encoding in the presence of channel noise. In the following, we will exemplarily consider NG, the argumentation for SOM and k-means being similar.

Often, the NG cost function is optimized by means of an online stochastic gradient descent. Alternatively, if data $\vec{x}^1, \ldots, \vec{x}^m$ are available priorly, batch optimization can be done. The corresponding discrete cost function is given by

$$E_{\mathrm{NG}}(\vec{w}, \vec{x}) = \frac{1}{2} \sum_{i=1}^{k} \sum_{j=1}^{m} h_\lambda(k_i(\vec{x}^j)) \cdot \|\vec{x}^j - \vec{w}^i\|^2 .$$

This is optimized by an iterative optimization of assignments and prototypes until convergence in Batch NG, see Algorithm 1. Thereby, the neighborhood cooperation is usually annealed to $\lambda \to 0$ during training such that the quantization error is optimized in the limit of small neighborhood size. It has been shown in [13] that this procedure converges after a finite number of steps to a local optimum of the NG cost function for fixed λ. In the following theoretical considerations, we will always assume fixed and usually small neighborhood parameter λ is chosen. This consideration approximately corresponds to final stages of training.

In the following, we will deal with the application of NG to settings where no Euclidean embedding of the data points is known. We will more generally deal with data points x^i which are characterized by pairwise proximities

$d(x^i, x^j)$ which are symmetric and 0 if $x^i = x^j$. For such general proximities, it can hold that no Euclidean embedding can be found. However, we will see that, also for non-Euclidean dissimilarities, a vector space with symmetric bilinear form $\langle \cdot, \cdot \rangle$ (which need not be positive definite) and embeddings $\vec{x}^i$ of the points x^i exist such that $d(x^i, x^j) = \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle$. We would like to stress that the cost function of NG can be formalized for every real vector space which possesses a symmetric bilinear form $\langle \cdot, \cdot \rangle$, by substituting the term $\|\vec{x}^i - \vec{w}^j\|^2$ in the cost function by $\langle \vec{x}^i - \vec{w}^j, \vec{x}^i - \vec{w}^j \rangle$. The corresponding cost function becomes

$$E_{\mathrm{NG}}(\vec{w}) = \frac{1}{2} \sum_{i=1}^{k} \sum_{j=1}^{m} h_\lambda(k_i(\vec{x}^j)) \cdot \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle$$

where the ranks $k_i(\vec{x}^j)$ are determined based on the bilinear form. In the same way, using this extended definition of the ranks, the procedure of Batch NG is well defined in every vector space equipped with a symmetric bilinear form. However, it is not guaranteed that this Batch NG procedure optimizes the NG cost function for the general setting, nor is the convergence of this procedure guaranteed, nor the fact that this procedure is useful at all. We will address these questions in more detail in the following.

3 Dissimilarity Data

Relational data x^i are not explicitly embedded in a Euclidean vector space, rather, pairwise similarities or dissimilarities are available. We assume that dissimilarities d_{ij} are given for every pair of data points $x^1, \ldots, x^m$. In the Euclidean setting, data can be represented as vectors $x^j = \vec{x}^j$ and the equality $d_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$ holds. In general, however, no such explicit embedding of data is available. Further, d_{ij} need not fulfill the requirements of a metric, i.e. the triangle inequality can be violated. Such data can stem from a general distance measure such as e.g. alignment distance for DNA sequences, Levenstein distance of words, general graph distances e.g. in a web graph, or even empirical results of questionnaires or experiments.

This situation is rather common in practice, and an overview about supervised classification techniques for similarity data has just recently been published [10]. Here we are interested in the unsupervised case. A variety of clustering algorithms for general dissimilarities has been proposed. This includes hierarchical clustering such as single or complete linkage, UPGMA, or neighbor joining [26, 71], which usually rely on heuristics, although guarantees of the correctness of the result can be derived in specific settings (e.g. for ultrametrics or additive metrics [34]). Alternatives rely on an appropriate cost function. Naturally, the standard kernel trick can be applied to many clustering algorithms or topographic mapping, such that a direct extension

of these approaches to similarity data which is induced by a kernel results. Proposals of this approach can be found in [12, 67, 57], for example. Clustering of dissimilarities which are represented by a graph can be formalized as a graph cut problem. Spectral clustering can be interpreted as an efficient approximation of the NP-hard optimization of the normalized or ratio graph cut [44]. Alternatively, in median or exemplar based clustering, the position of prototypes in the standard quantization error are restricted to locations in the set of given data points. This way, the quantization error can directly be transferred to the setting of general dissimilarities. Proposals which follow this line of argumentation include median clustering as proposed for SOM and NG [38, 13] and exemplar based clustering by formulating the problem in terms of a factor graph leading to affinity propagation [16]. By restricting the prototype locations, the set of potential solutions becomes restricted such that these proposals do not correspond to standard k-means or SOM in the Euclidean setting.

A few approaches try to directly generalize the standard Euclidean setting to general dissimilarity data. One proposal is to optimize a cost function for pairwise clustering which relies on the cluster assignments only, i.e. it can be formulated for pairwise dissimilarities, but which is equivalent to the standard quantization error in the Euclidean setting. This cost function can be optimized e.g. using some metaheuristic or deterministic annealing [30]. However, in the general setting, the notion of prototypes is lost this way and a clustering is represented in terms of assignments only. Similarly, the approach [18] uses deterministic annealing for a reliable training of SOM for proximity data. The transfer from vectorial data is achieved by an interpretation of SOM as an encoder-decoder framework. We will later see that these methods can be interpreted as deterministic annealing of standard Euclidean k-means clustering and SOM, respectively, which are transferred to proximity data using relational clustering, as will be proposed in this article.

A direct extension of k-means and fuzzy-k-means to dissimilarity data has been proposed in [27, 28]. Here the assumption is made that a Euclidean embedding of data exists, but it is unknown. In this case, k-means can be equivalently reformulated in terms of dissimilarities only. The resulting methods are termed relational clustering since they can deal with dissimilarities which result from relations rather than vectors. In this contribution, we will follow the last approach and transfer these ideas to topographic mapping. First, we consider the situation that a Euclidean embedding of data exist and derive the relational neural gas in this setting. Afterwards, we discuss the setting for non Euclidean situations.

3.1 Euclidean Data

We assume, as before, that training data $x^1, \ldots, x^m$ are given in terms of pairwise dissimilarities d_{ij}. One special setting is that data originate from a Euclidean distance measure, that means, we are able to find 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) $\Phi : x \mapsto \Phi(x^i) = \vec{x}^i$ with $d_{ij} = \|\Phi(x^i) - \Phi(x^j)\|^2$. Since the Euclidean distance is related to the Euclidean dot product, this setting is referred to as 'kernel trick for distances' in the literature. The approach [62], as an example, investigates sufficient conditions for a metric to fulfill this property. Since an explicit formula for the embedding Φ need not be known, we cannot directly apply batch NG in the embedding space. We want to derive a possibility to reformulate batch NG in such a way that an explicit embedding Φ is not needed, i.e. batch NG can directly be applied to this setting. The key observation is based on the fact that optimum prototype locations $\vec{w}^j$ of batch NG can be expressed as linear combination of the given data points. Therefore, the unknown values $\|\vec{x}^j - \vec{w}^i\|^2$ can be expressed in terms of known values d_{ij}. Correspondingly, the NG cost function can be substituted by a dual cost function in terms of ranks and cluster assignments only.

Relational Clustering Algorithm

In the following section, we lay out the details of this procedure. Thereby, we always make the assumption that pairwise dissimilarities d_{ij} are given, $i, j = 1 \ldots, m$, D denotes the corresponding matrix of dissimilarities. We assume that, for D, there exists a finite dimensional real vector space X together with a symmetric bilinear form $\langle \cdot, \cdot \rangle$, and there exist vectors $\vec{x}^j \in X$ with $d_{ij} = \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle$. We will see later, that this property is always guaranteed if D is symmetric with zero diagonal elements. For some theorems, we require X to be Euclidean space, i.e. we require positive definiteness of the bilinear form $\langle \cdot, \cdot \rangle$. This will be explicitly mentioned at corresponding places. Interestingly, the principled procedure as derived in this section as well as a couple of results hold for the more general case that X is an arbitrary (possibly non-Euclidean) vector space equipped with a symmetric bilinear form.

As already mentioned, prototypes will be restricted to convex combinations of given data points since optimum prototypes for NG have this form. For NG, we need to compute dissimilarities. It is possible to compute the dissimilarity of two such prototypes (or a prototype and a given data point) based on the given dissimilarity matrix only. More precisely, generalizing results as presented e.g. in [27]:

Theorem 1 *For every coefficient vectors $\alpha, \alpha' \in \mathbb{R}^m$ with $\sum_i \alpha_i = 1 = \sum_i \alpha'_i$ the*

following equality holds:

$$\left\langle \sum_i \alpha_i \vec{x}^i - \sum_i \alpha'_i \vec{x}^i, \sum_i \alpha_i \vec{x}^i - \sum_i \alpha'_i \vec{x}^i \right\rangle = (\alpha')^t D\alpha - \frac{1}{2} \cdot \alpha^t D\alpha - \frac{1}{2} \cdot (\alpha')^t D\alpha'$$

Proof: It holds

$$\begin{array}{l} \alpha^t D\alpha \\ = \sum_{ij} \alpha_i \alpha_j \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle \\ = \sum_i \alpha_i \langle \vec{x}^i, \vec{x}^j \rangle - 2 \sum_{ij} \alpha_i \alpha_j \langle \vec{x}^i, \vec{x}^j \rangle + \sum_j \alpha_j \langle \vec{x}^j, \vec{x}^j \rangle \\ = 2 \sum_i \alpha_i \langle \vec{x}^i, \vec{x}^i \rangle - 2 \sum_{ij} \alpha_i \alpha_j \langle \vec{x}^i, \vec{x}^j \rangle \end{array}$$

and

$$\begin{array}{l} (\alpha')^t D\alpha \\ = \sum_{ij} \alpha'_i \alpha_j \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle \\ = \sum_i \alpha'_i \langle \vec{x}^i, \vec{x}^i \rangle + \sum_j \alpha_j \langle \vec{x}^j, \vec{x}^j \rangle - 2 \sum_{ij} \alpha'_i \alpha_j \langle \vec{x}^i, \vec{x}^j \rangle \end{array}$$

Hence,

$$\begin{array}{rl} & (\alpha')^t D\alpha - 1/2 \cdot \alpha^t D\alpha - 1/2 \cdot (\alpha')^t D\alpha' \\ = & -2 \sum_{ij} \alpha'_i \alpha_j \langle \vec{x}^i, \vec{x}^j \rangle + \sum_{ij} \alpha_i \alpha_j \langle \vec{x}^i, \vec{x}^j \rangle \\ & + \sum_{ij} \alpha'_i \alpha'_j \langle \vec{x}^i, \vec{x}^j \rangle \\ = & \left\langle \sum_i \alpha_i \vec{x}^i - \sum_i \alpha'_i \vec{x}^i, \sum_i \alpha_i \vec{x}^i - \sum_i \alpha'_i \vec{x}^i \right\rangle \end{array}$$

□

Prototypes can be written as a convex combination $\vec{w}^i = \sum_j h_\lambda(k_{ij}) \vec{x}^j / \sum_j h_\lambda(k_{ij})$. Thus, because of Theorem 1, we can compute the dissimilarity between a prototype and a given data point based on the coefficients in this convex combination and the dissimilarity matrix D. Further, it is not necessary to explicitly store the prototype locations, rather, we can alternatively represent prototypes by means of the coefficients $\alpha_{ij} = h_\lambda(k_{ij}) / \sum_j h_\lambda(k_{ij})$. These observations lead to the following equivalent formulation of batch NG which does not rely on an explicit vectorial representation of the data but on the matrix D only, and which extends work as presented in [27] to neighborhood cooperation:

Theorem 2 *Relational NG as shown in Algorithm 2 constitutes an equivalent formulation of batch NG by means of the identity $\vec{w}^i = \sum_j \alpha_{ij} \vec{x}^j$. Thereby, α_i refers to the vector $(\alpha_{i1}, \ldots, \alpha_{im})^t$ and $[\cdot]_j$ refers to coordinate j. Equivalence means, if $\vec{w}^i = \sum_j \alpha_{ij} \vec{x}^j$ holds for the initialization of batch NG and relational NG, respectively, then the same identity holds for these values after every epoch of batch NG and relational NG, respectively.*

Proof: The equivalence holds because of Theorem 1 by setting α' to the jth unit vector because of $\vec{w}^i = \sum_j h_\lambda(k_{ij}) \vec{x}^j / \sum_j h_\lambda(k_{ij})$. Further, we can assume that prototypes are initialized in the convex hull of data points, i.e. also for

Algorithm 2: Relational NG

input

symmetric dissimilarity matrix with zero diagonal $D \in \mathbb{R}^{m \times m}$;

begin

init $\alpha_{ij} \geq 0$ such that $\sum_j \alpha_{ij} = 1$;

repeat

compute $\mathrm{dist}_{ij} := [D\alpha_i]_j - \frac{1}{2} \cdot \alpha_i^t D \alpha_i$;

set $k_{ij} := |\{l \mid \mathrm{dist}_{lj} < \mathrm{dist}_{ij}\}|$;

set $\alpha_{ij} := h_\lambda(k_{ij}) / \sum_j h_\lambda(k_{ij})$;

until convergence;

return α_{ij};

end.

the initialization equivalent coefficients α_{ij} can be found. □

Using the identity $\vec{w}^j = \sum_j \alpha_{ij} \vec{x}^j$, relational NG computes exactly the same prototype locations in every epoch as Batch NG does. In relational NG, however, prototype locations are computed only indirectly by means of the coefficients α_{ij}. It is not necessary to know the vectorial representation of the training vectors, rather, the knowledge of the pairwise dissimilarities d_{ij} is sufficient to compute the output of batch NG in terms of coefficients α_{ij}. Hence, NG can directly be performed based on a given dissimilarity matrix D only. For every prototype, m coefficients are stored, m denoting the number of training points. α_{ij} corresponds to the part which data points $\vec{x}^j$ takes in representing prototype $\vec{w}^i$. The larger α_{ij}, the more does $\vec{x}^j$ contribute to the prototype location. In the limit of zero neighborhood cooperation $\lambda \to 0$, the prototypes converge to the mean of the data points in their receptive fields. As for Batch NG, the neighborhood parameter λ is usually annealed to 0 during training. For all theoretical considerations, we assume a fixed and possibly small parameter λ corresponding to final stages of training.

The space complexity of relational clustering is linear w.r.t. the number of training data because of the requirement to store α_{ij} for $j = 1, \ldots, m$. The time complexity of one training epoch is quadratic w.r.t. the number of training points, since the computation of $[D\alpha_i]_j$ takes linear time for every j. The quantity $\alpha_i^t D \alpha_i$ does not depend on j and has to be computed only once for every i, thus it also contributes quadratic complexity. Since embedding data into a vector space based on the distance matrix D has cubic complexity

depending on the number of training points as we will see later, this procedure offers a more efficient alternative for NG if an embedding of data is not given priorly. Obviously, the algorithm provided by relational NG can be applied to every setting where pairwise dissimilarities are known. Euclideanity of an underlying vector space is not necessarily required to apply the algorithm. However, in such cases no guarantee on the convergence of relational NG is given or a connection to a cost function, just as no guarantee is given for the vectorial counterpart batch NG in non-Euclidean settings. We will discuss later, in how far the procedure is reasonable in such situations.

Dual Cost Function

For the Euclidean setting, i.e. $d_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$ for some embedding holds, convergence of relational NG towards a local optimum $\vec{w}^i = \sum_j \alpha_{ij}\vec{x}^j$ of the NG cost function is guaranteed, since relational NG is equivalent to standard batch NG; therefore, the same guarantees as for batch NG hold, provided the underlying space X is Euclidean [13]. For a non-Euclidean setting, convergence of the algorithm towards a local optimum of the NG cost function is not guaranteed. For both cases, however, prototypes are only indirectly obtained this way and the NG cost function can only be evaluated when the data $\vec{x}^j$ are known. Therefore, it is interesting to see whether there exists an alternative to evaluate the cost function of NG based on the quantities given by relational NG and the distance matrix D.

We introduce the following function which depends on the assignments k_{ij} and the dissimilarity matrix D only and which extends the dual cost function of k-means to neighborhood cooperation [27]:

$$E_{\mathrm{NG}}^{\vee}(k_{ij}, d_{ij}) = \sum_i \frac{1}{4\sum_j h_\lambda(k_{ij})} \cdot \sum_{jj'} h_\lambda(k_{ij}) h_\lambda(k_{ij'}) d_{jj'} \; .$$

The function $E_{\mathrm{NG}}^{\vee}(k_{ij}, d_{ij})$ has to be optimized with respect to the assignments k_{ij} under the constraint that $k_{1j}, \ldots, k_{kj}$ constitute a permutation of $\{0, \ldots, k-1\}$ for every fixed j. The cost function measures the intra cluster distances, averaged over the local neighborhood as induced by the data.

We will see that this cost function is in some sense dual to the standard NG cost function: For fixed points found by relational or batch NG, respectively, the values of the NG cost function and the above function coincide, as we will see in the following. Further, global optima of this cost function can be related to global optima of the standard NG cost function, as we will also show in the following. Therefore, we refer to this cost function as the dual NG cost function, which expresses the objective of NG in the dual variable k_{ij} instead of the prototypes $\vec{w}^i$.

This cost function is independent of an embedding of data in a vector space. Rather, it evaluates the quality of a clustering based on the assign-

ments k_{ij} which determine the rank of cluster i for a data point $\vec{x}^j$, and the pairwise distances $d_{jj'}$. This cost function constitutes a direct extension of the well-known cost function of pairwise data clustering, referred to as dual k-means cost function in the literature, to neighborhood cooperation [27, 30]:

$$E^{\vee}_{\mathrm{k-means}}(\delta_{ij}, d_{ij}) = \sum_i \frac{1}{4\sum_j \delta_{ij}} \cdot \sum_{jj'} \delta_{ij}\delta_{ij'} d_{jj'}$$

This cost function is the dual function of the standard quantization error, and it has to be optimized for assignments $\delta_{ij} \in \{0,1\}$ such that $\sum_i \delta_{ij} = 1$ for every j. It directly measures the intra cluster distances of a given clustering. It is well known that optimum prototypes of the vector quantization error correspond to optimum cluster assignments of the dual cost function in the Euclidean space, both values computed in the course of k-means clustering [27, 28]. The dual cost function $E^{\vee}_{\mathrm{k-means}}(\delta_{ij}, d_{ij})$ is more general in the sense that it offers a valid cost function for arbitrary dissimilarities d_{ij}, and alternative optimization methods such as deterministic annealing have been proposed for this setting [30].

Now we want to formally establish a relation of the NG cost function and its dual. For this purpose, we reformulate the dual cost function of NG for vectors $\vec{x}$ in a vector space X, which allows us to relate the dual function depending on k_{ij} only to a mixed function depending on k_{ij} and $\vec{w}^i$, and, finally, the standard NG cost function.

Theorem 3 *If $d_{ij} = \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle$, then the dual cost function $E^{\vee}_{\mathrm{NG}}(k_{ij}, \vec{x})$ of NG equals*

$$E^{\vee}_{\mathrm{help}}(k_{ij}, \vec{x}) := \frac{1}{2}\sum_{ij} h_\lambda(k_{ij}) \left\langle \vec{x}^j - \frac{\sum_l h_\lambda(k_{il})\vec{x}^l}{\sum_l h_\lambda(k_{il})}, \vec{x}^j - \frac{\sum_l h_\lambda(k_{il})\vec{x}^l}{\sum_l h_\lambda(k_{il})} \right\rangle .$$

Proof: We obtain

$$\begin{aligned}
& E^{\vee}_{\mathrm{NG}}(k_{ij}, \vec{x}) \\
= & \sum_i \frac{1}{4\sum_j h_\lambda(k_{ij})} \cdot \sum_{jj'} h_\lambda(k_{ij}) h_\lambda(k_{ij'}) \cdot (\langle \vec{x}^j, \vec{x}^j \rangle + \langle \vec{x}^{j'}, \vec{x}^{j'} \rangle - 2 \cdot \langle \vec{x}^j, \vec{x}^{j'} \rangle) \\
= & \sum_i \frac{1}{2\sum_j h_\lambda(k_{ij})} \cdot \sum_{jj'} h_\lambda(k_{ij}) h_\lambda(k_{ij'}) \cdot (\langle \vec{x}^j, \vec{x}^j \rangle - \langle \vec{x}^j, \vec{x}^{j'} \rangle) \\
= & \sum_{ij} \frac{h_\lambda(k_{ij})}{2} \langle \vec{x}^j, \vec{x}^j \rangle - \frac{1}{\sum_j h_\lambda(k_{ij})} h_\lambda(k_{ij}) h_\lambda(k_{ij'}) \langle \vec{x}^j, \vec{x}^{j'} \rangle \\
& + \sum_{ijl'} \frac{h_\lambda(k_{ij})}{2\sum_j h_\lambda(k_{ij})} h_\lambda(k_{il}) h_\lambda(k_{il'}) \langle \vec{x}^l, \vec{x}^{l'} \rangle \\
= & \frac{1}{2}\sum_{ij} h_\lambda(k_{ij}) \left\langle \vec{x}^j - \frac{\sum_l h_\lambda(k_{il})\vec{x}^l}{\sum_l h_\lambda(k_{il})}, \vec{x}^j - \frac{\sum_l h_\lambda(k_{il})\vec{x}^l}{\sum_l h_\lambda(k_{il})} \right\rangle \\
= & E^{\vee}_{\mathrm{help}}(k_{ij}, \vec{x})
\end{aligned}$$

□

This reformulation of the dual cost function allows us to link its value to the standard NG cost function for fixed points obtained by relational NG:

Theorem 4 *Assume relational NG converges towards a fixed point of the algorithm, i.e. coefficients α_{ij} and k_{ij} are found which remain constant after further adaptation steps. Define the vector $\vec{w}^i = \sum \alpha_{ij} \vec{x}^j$. Then $E_{\mathrm{NG}}(\vec{w}, \vec{x}) = E_{\mathrm{NG}}^{\vee}(k_{ij}, \vec{x})$.*

Proof: For a fixed point $\alpha_{ij} = h_\lambda(k_{ij}) / \sum_j h_\lambda(k_{ij})$ of relational NG, and thus a fixed point $\vec{w}^i = \sum_j h_\lambda(k_{ij}) \vec{x}^j / \sum_j h_\lambda(k_{ij})$ of batch NG, we find $E_{\mathrm{NG}}^{\vee}(k_{ij}, \vec{x}) = E_{\mathrm{help}}^{\vee}(k_{ij}, \vec{x}) = \frac{1}{2} \cdot \sum_{ij} h_\lambda(k_i(\vec{x}^j)) \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle = E_{\mathrm{NG}}(\vec{w}, \vec{x})$ □

This result allows us to evaluate the outcome of relational NG based on the dissimilarity matrix D and the coefficients α_{ij} only. The value of the dual cost function corresponds to the standard cost function value of standard NG if fixed points of the algorithms are considered, hence the quality of the solution can be judged independently of a concrete vectorial embedding of the data points. Obviously, this theorem also holds under the weaker condition, that k_{ij} coincides with the ranks of data point j for the prototypes $\vec{w}^i = \sum \alpha_{ij} \vec{x}^j$. As we will see in the experiments, this condition is often already approximately fulfilled in early stages of training such that both cost functions are approximately identical for relational NG during training.

It is interesting to further investigate whether the structure of the NG cost function and its dual are in some sense equivalent. This refers to the question whether local or global optima of these cost functions can be related to each other. We start looking at global optima of the cost functions. The following holds:

Theorem 5 *Define*

$$E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x}) := \frac{1}{2} \sum_{ij} h_\lambda(k_{ij}) \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle \,.$$

Then

$$E_{\mathrm{help}}^{\vee}(k_{ij}, \vec{x}) \geq \inf_{\vec{w}} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x}) \,.$$

If $\langle \cdot, \cdot \rangle$ is positive definite, equality holds. Further, it holds

$$E_{\mathrm{NG}}(\vec{w}, \vec{x}) = \min_{k_{ij} \in P_j} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$$

for every real vector space X with symmetric bilinear form where P_j denotes the space of permutations of $\{0, \ldots, k-1\}$ for every j.

Proof: The inequality $E_{\mathrm{help}}^{\vee}(k_{ij}, \vec{x}) \geq \inf_{\vec{w}} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$ is obvious because of the definition of the functions.

A positive definite bilinear form over a finite dimensional vector space can be expressed as $\langle \vec{x}, \vec{y} \rangle = \vec{x}^t A \vec{y}$ with a symmetric positive definite matrix A. We can compute the directional derivative of the cost function $\partial E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$ into the direction $\vec{\xi}$ as $\sum_i h_\lambda(k_{ij})(\vec{w}^i - \vec{x}^j)^t A \xi$. For local optima, this must be zero for every ξ, hence we find $\sum_i h_\lambda(k_{ij})(\vec{w}^i - \vec{x}^j) = 0$ and, hence, $\vec{w}^i = \sum_j h_\lambda(k_{ij}) \vec{x}^j / h_\lambda(k_{ij})$. The Hessian matrix of this cost function constitutes a matrix with diagonal blocks $\sum_j h_\lambda(k_{ij}) A$ and 0 entries otherwise. This is positive definite because A is positive definite and the factor $\sum_j h_\lambda(k_{ij})$ is positive, thus, this position constitutes a local optimum of the cost function. Since $E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$ is a quadratic form with respect to $\vec{w}$, this local optimum must be the global optimum. Therefore, equality holds in this case.

Assume $k_{ij} \in P_j$. Assume for two indices i and i' holds $k_{ij} < k_{i'j}$ and $\langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle > \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^{i'} \rangle$. Then, because of the monotonicity of h_λ, $h_\lambda(k_{ij}) \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle + h_\lambda(k_{ij'}) \langle \vec{x}^j - \vec{w}^{i'}, \vec{x}^j - \vec{w}^{i'} \rangle > h_\lambda(k_{ij'}) \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle + h_\lambda(k_{ij}) \langle \vec{x}^j - \vec{w}^{i'}, \vec{x}^j - \vec{w}^{i'} \rangle$, thus, we can decrease the cost function by substituting these two assignments. Therefore, optimum assignments k_{ij} are given by the ranks $k_{ij} = k_i(\vec{x}^j) = |\{\vec{w}^l \mid \langle \vec{x}^j - \vec{w}^l, \vec{x}^j - \vec{w}^l \rangle < \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \}|$, hence, the equality $E_{\mathrm{NG}}(\vec{w}, \vec{x}) = \min_{k_{ij} \in P_j} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$ follows. □

As a consequence of this theorem, we find equivalence of global optima of the NG cost function and its dual in the Euclidean setting:

Theorem 6 *The following inequality is valid*

$$\inf_{\vec{w}} E_{\mathrm{NG}}(\vec{w}, \vec{x}) \leq \min_{k_{ij} \in P_j} E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij}) \,.$$

If $\langle \cdot, \cdot, \rangle$ is positive definite, equality holds.

Proof: Because of Theorems (4,5), we find

$$\min_{k_{ij} \in P_j} E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij}) \geq \min_{k_{ij} \in P_j} \inf_{\vec{w}} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$$

with equality for positive definite bilinear form and

$$\inf_{\vec{w}} E_{\mathrm{NG}}(\vec{w}, \vec{x}) = \inf_{\vec{w}} \min_{k_{ij} \in P_j} E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x}) \,.$$

Because of the finiteness of P_j we can exchange the infimum and minimum, hence the theorem follows. □

Thus, in the Euclidean case, the NG cost function and its dual coincide in the sense that we have a correspondence of the values of global optima. Since global optima are fixed points of batch NG and relational NG in the Euclidean setting, Theorem 4 also gives a correspondence of the global optima itself.

The question occurs whether a more detailed correspondence of the functions according to their overall shape can be established. In particular, a connection of the values as obtained by batch and relational NG and their role

(e.g. local or global optimum) with respect to the cost functions would be interesting. Batch NG repeatedly optimizes the assignments k_{ij} and prototype locations $\vec{w}$ of the cost function $E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x})$ if a Euclidean embedding of data can be found. The convergence proof as presented in [13] relies on the fact that optimum values k_{ij} and $\vec{w}$, respectively, are determined in every step in the Euclidean setting, as also computed in the above proofs. Thus, the cost function decreases in successive steps until convergence can be observed for Euclidean, or, more generally, positive definite symmetric bilinear form. It is shown in [13], that the obtained fixed point constitutes a local optimum of the cost function $E_{\mathrm{NG}}(\vec{w}, \vec{x})$ under mild conditions on the setting. The same holds for RNG if the pairwise distances stem from a Euclidean space because RNG is just an equivalent formulation of batch NG using the identity $\vec{w}^i = \sum_j \alpha_{ij} \vec{x}^j$. However, in both cases, it is not guaranteed that a global optimum of the cost functions is reached.

We already know that, for fixed points of relational or batch NG, the values of the NG cost function and its dual coincide. Further, in the Euclidean setting, a local optimum $\vec{w}$ of the NG cost function is reached, and, conversely, every local optimum of the NG cost function constitutes a fixed point of NG. Now the question occurs in how far local optima of the NG cost function can be related to local optima of the dual cost function. For this purpose, we have to determine a neighborhood structure for the solution space of the dual cost function, since k_{ij} constitute discrete values. We use the following simple structure: We define a neighborhood structure on P_i such that k_{ij} and k'_{ij} are *neighbored* if and only if $k_{ij} = k'_{ij}$ for all but two indices (ij). An assignment k_{ij} is called a *local optimum* of $E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij})$ if $E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij}) \leq E^{\vee}_{\mathrm{NG}}(k_{ij'}, d_{ij})$ for all k'_{ij} in the neighborhood of k_{ij}. Using this definition, we obtain the following result:

Theorem 7 *Assume $\langle \cdot, \cdot \rangle$ is positive definite. Then local optima of $E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij})$ constitute fixed points of relational NG.*

Proof: Assume k_{ij} are given which do not constitute a fixed point of relational NG. Define $\vec{w}^i = \sum_j h_\sigma(k_{ij}) \vec{x}^j / h_\sigma(k_{ij})$. Then, k_{ij} does not coincide with the ranks $k_i(\vec{x}^j)$. Thus, we can argue as in the proof of Theorem 5 that substituting two assignments k_{ij} for which $k_{ij} < k_{ij'}$ holds, but the corresponding distances fulfill $\mathrm{dist}_{ij} > \mathrm{dist}_{ij'}$ leads to assignments k'_{ij} with $E_{\mathrm{help}}(\vec{w}, k_{ij}, \vec{x}) > E_{\mathrm{help}}(\vec{w}, k'_{ij}, \vec{x})$. Setting $(w^i)' = \sum_j h_\sigma(k'_{ij}) \vec{x}^j / h_\sigma(k'_{ij})$ we obtain as in the proof of Theorem 5 that $E_{\mathrm{help}}(\vec{w}, k'_{ij}, \vec{x}) > E_{\mathrm{help}}(\vec{w}', k'_{ij}, \vec{x})$. Because of Theorem 3, this means $E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij}) > E^{\vee}_{\mathrm{NG}}(k'_{ij}, d_{ij})$, thus, k_{ij} does not constitute a local optimum. □

The converse, however, is not true.

Theorem 8 *There exist fixed points of relational NG which do not constitute a*

local optimum of the dual cost function with respect to the neighborhood structure as defined above in the Euclidean setting.

Proof: We consider the limit case $\lambda \to 0$, i.e. crisp k-means. NG approximates this setting for small enough λ. Consider the Euclidean points in $\mathbb{R}^2$: $\vec{x}^1 = (0,0)$, $\vec{x}^2 = (0,1.1)$, $\vec{x}^3 = (2,1.1)$ and prototypes $\vec{w}^1 = (0,0)$ and $\vec{w}^2 = (1,1.1)$. The corresponding crisp assignments δ_{ij} (corresponding to $h_\lambda(k_{ij})$ for $\lambda \to 0$) are $\delta_{11} = 1$, $\delta_{12} = 0$, $\delta_{13} = 0$. This is a fixed point of batch NG and, hence, relational NG. The alternative $\delta_{11} = 1$, $\delta_{12} = 1$, $\delta_{13} = 0$, however, is in the neighborhood of this solution and leads to the better prototypes $\vec{w}^1 = (0, 0.55)$, $\vec{w}^2 = (2, 1.1)$. □

Thus, it is guaranteed that local optima of the NG cost function or its dual, respectively, lead to fixed points of Batch NG or relational NG in the Euclidean setting. Conversely, every fixed point constitutes a local optimum of the NG cost function under mild conditions while the converse is not true, there exist settings where a fixed point of RNG can be further improved within the neighborhood structure as defined above for the dual NG cost function. Since a one-one connection of fixed points of Batch NG and RNG exists, this leads to the consequence that the dual NG cost function possesses less local optima than the original NG cost function with respect to the neighborhood as defined above; in particular, there is no exact correspondence of the overall structure of the NG cost function and its dual.

Out of Sample Extensions

We will conclude this section by introducing two further issues which are of relevance in clustering algorithms: is it possible to extend a given clustering to new data points which are not contained in the training set (so called out-of-sample extensions)? Is it possible to integrate prior knowledge e.g. given by a partial labeling of the data into the algorithms? Both questions can be answered very satisfactorily in the context of prototype-based clustering as follows:

Out of sample extensions of NG to a point $\vec{x} \neq \vec{x}^j$ can be defined based on the standard winner assignment, i.e. $\vec{x}$ is mapped to the class represented by the prototype $\vec{w}^j$ with smallest dissimilarity $d(\vec{w}^j, \vec{x}^i)$ to $\vec{x}$. This very simple scheme is one of the benefits of prototype based clustering as opposed to alternatives such as assignment based clustering. This scheme can directly be transferred to relational clustering because of its connection to batch NG in a vector space. The question occurs whether this procedure can be transferred to a scheme which refers to pairwise dissimilarities only without an explicit knowledge of the embedding of data or prototypes. The key ingredient is a scheme which allows to compute the dissimilarity of a new data point and a prototype based on α_{ij} and pairwise dissimilarities only:

Theorem 9 *Assume a data point $\vec{x}$ is contained in X and the dissimilarities to points $\vec{x}^j$ are given as $d_j = \langle \vec{x}^j - \vec{x}, \vec{x}^j - \vec{x} \rangle$, the corresponding vector is denoted by $D(\vec{x})$. Assume $\vec{w} = \sum_j \alpha_j \vec{x}^j$ with $\sum_j \alpha_j = 1$. α denotes the corresponding vector. Then*

$$\langle \vec{x} - \vec{w}, \vec{x} - \vec{w} \rangle = (D(\vec{x})^t \cdot \alpha) - 1/2 \cdot \alpha^t \cdot D \cdot \alpha \,.$$

Proof: This follows directly from Theorem 1, whereby we consider the linear combinations of $\vec{x}, \vec{x}^1, \ldots, \vec{x}^m$ with coefficients $(1, 0 \ldots, 0)$ and $(0, \alpha)$, respectively. □

This allows to compute out-of-sample extensions of the clustering based on the dissimilarities and prototype coefficients only.

Supervision

Clustering constitutes an ill-posed problem since the objective of clustering is not clear a priori, see e.g. [7]. If pairwise distances of data are given, the dual cost function of k-means constitutes one possible objective of clustering. Alternatively, when restricting to prototype-based algorithms, the standard quantization error can serve as an objective function. We have seen that these cost functions are equivalent for metric data and neural gas and relational neural gas optimize a relaxation thereof. However, it is not clear a priori whether the outcome meets the intended result in practical problems. 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 [35, 36].

Here we consider the situation that additional label information is available which should be accounted for by clustering or visualization. Thereby, labels are embedded in $\mathbb{R}^d$ and can be fuzzy. 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. The basic idea consists in a substitution of the standard dissimilarities $\langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle$ by a mixture

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

which takes the similarity of label assignments into account and where $\beta \in [0, 1]$ controls the influence of the label values. This procedure has been proposed in [20, 21, 65] for Euclidean and median clustering and online neural gas, respectively. One can use the same principles to extend relational clustering. The cost function of NG becomes

$$E_{\mathrm{NG}}(\vec{w}, \vec{Y}, \vec{x}) = \sum_{ij} h_\lambda(k_i(\vec{x}^j)) \cdot \left((1 - \beta) \cdot \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle + \beta \cdot \|\vec{y}^j - \vec{Y}^i\|^2 \right)$$

Algorithm 3: Supervised Relational NG

input

symmetric dissimilarity matrix $D \in \mathbb{R}^{m \times m}$ with zero diagonal;

label information $\{\vec{y}^1, \ldots, \vec{y}^m\} \subset \mathbb{R}^d$;

begin

init α_{ij} with $\sum_j \alpha_{ij} = 1$;

repeat

compute $\mathrm{dist}_{ij} := (1-\beta) \cdot ([D \cdot \alpha_i]_j - 1/2 \cdot \alpha_i^t D \alpha_i) + \beta \cdot \|Y^i - y^j\|^2$;

set $k_{ij} := |\{l \mid \mathrm{dist}_{lj} < \mathrm{dist}_{ij}\}|$;

set $\alpha_{ij} := h_\lambda(k_{ij}) / \sum_j h_\lambda(k_{ij})$;

set $\vec{Y}^i := \sum_j \alpha_{ij} \vec{y}^j$;

until convergence;

return α_{ij}, $\vec{Y}^i$;

end.

where $k_i(\vec{x}^j)$ denotes the rank of neuron i measured according to the dissimilarities $(1-\beta) \cdot \langle \vec{x}^j - \vec{w}^i, \vec{x}^j - \vec{w}^i \rangle + \beta \cdot \|\vec{y}^j - \vec{Y}^i\|^2$. Batch NG can be directly derived thereof. The ranks are determined based on this extended cost term, this 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.

Relational learning becomes possible in the same way as beforehand. Assume pairwise dissimilarities of data d_{ij} are given instead of explicit data locations. Then, supervised relational neural gas results as displayed in Algorithm 3. Assume that a vector space and a symmetric bilinear form exists which induces the dissimilarities, i.e. $d_{ij} = \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle$. In that case, supervised NG can be interpreted as an extension of standard NG applied to the data with coefficients $\sqrt{1-\beta} \cdot \vec{x}^j$ in the first dimensions and $\sqrt{\beta} \cdot \vec{y}^j$ in the remaining ones. In particular, the theorems as proved in this section also hold for the supervised case.

3.2 Non Euclidean Data

It often holds that data cannot be embedded in Euclidean space. As an example, discrete data such as DNA sequences or strings can be considered and pairwise dissimilarities stem from an alignment of data. This yields a metric, but not necessarily the Euclidean one. For data which stem from exper-

imental measurements it can even be the case that metric properties such as the triangle equality are violated. In these cases, parts of the argumentation of the previous section do no longer hold. Here we want to investigate the question whether relational NG can still be applied in this more general setting, whether it can be connected to a batch NG scheme in an appropriate vector space, whether a connection to the NG cost function and its dual can be made, and we want to investigate properties such as convergence of the algorithm based on these findings. Further, we will put relational clustering as introduced in the last section into a wider context and relate it to two well known and very powerful clustering schemes for proximity data which are based on deterministic annealing, the framework of pairwise data clustering as proposed by Hofmann/Buhmann [30] and extended to neural gas schemes in [31], and the SOM for proximity data as introduced by Graepel/Obermayer [18] and later on extended to hyperbolic SOM [60]. We will argue that relational clustering constitutes a simple way which allows to derive deterministic annealing variants as proposed in [30, 18] directly from the corresponding deterministic annealing schemes for standard k-means and SOM, respectively, in the Euclidean setting. For general dissimilarities, relational clustering as well as deterministic annealing can be related to clustering algorithms in a vector space, in particular an interpretation in terms of prototypes is possible for all algorithms. However, it can be shown that none of the algorithms guarantees convergence to a local optimum of the dual cost function or related, nor convergence of the algorithm at all.

Relational Clustering and Batch Clustering in Pseudo-Euclidean Space

We assume in the following that data are represented by pairwise dissimilarities d_{ij}. D denotes the corresponding dissimilarity matrix. This setting has been considered e.g. in [30, 56, 18]. There, the dual cost function of k-means or related costs are considered, which are well defined for general dissimilarities. The approaches [30, 56, 18] propose an optimization of this discrete function using methods of statistical physics, resulting in deterministic annealing schedules for clustering. Note that approximations have to be used because optimization of this cost function constitutes an NP hard problem [9]. In this section, we discuss in how far relational NG can be seen as an alternative solution and we discuss in which cases optimization by relational NG as well as deterministic annealing fails. Note that relational NG as defined above can be applied as an algorithm to every setting where a dissimilarity matrix D is given. However, it is not clear in how far this procedure leads to meaningful results.

In the following, we always make the reasonable assumption that the diagonal is zero, i.e. $d_{ii} = 0$. Further, we assume symmetry of D, i.e. $d_{ij} = d_{ji}$. The latter does not constitute a restriction because it does not affect the cost function. More precisely, it has been shown e.g. in [41] that the dual cost

function of k-means is invariant with respect to symmetric transformations. The same holds for the dual cost function of NG:

Theorem 10 *Assume pairwise dissimilarities* $d_{ij} \geq 0$ *are given. If we set* $d'_{ij} := (d_{ij} + d_{ji})/2$ *then* $E^{\vee}_{\mathrm{NG}}(k_{ij}, d_{ij}) = E^{\vee}_{\mathrm{NG}}(k_{ij}, d'_{ij})$ *is not affected by this transform.*

Proof: This equality is obvious because of the identity

$$\sum_{jj'} h_\lambda(k_{ij}) h_{\lambda'}(k_{ij'}) d_{jj'} = \sum_{jj'} h_\lambda(k_{ij}) h_{\lambda'}(k_{ij'}) d_{j'j} \,.$$

□

All finite data sets which are characterized by a symmetric dissimilarity matrix with zero diagonal can be embedded into a vector space which possesses a symmetric bilinear form $\langle \cdot, \cdot, \rangle$ as follows (see e.g. [53, 17]): define

$$J := I - 1/m \vec{1}\vec{1}^t$$

with identity matrix I and the vector $\vec{1} = (1, \ldots, 1) \in \mathbb{R}^m$. Define

$$G := -\frac{1}{2} \cdot JDJ \,.$$

Obviously, this matrix is symmetric and, thus, it can uniquely be decomposed into the form

$$G = Q \Lambda Q^t$$

with orthonormal matrix Q and diagonal matrix of eigenvalues Λ with p positive and q negative entries. Taking the square root of Λ allows the alternative representation in the form

$$G = X I_{pq} X^t = Q |\Lambda|^{1/2} \begin{pmatrix} I_{pq} & 0 \\ 0 & 0 \end{pmatrix} |\Lambda|^{1/2} Q^t$$

where I_{pq} constitutes a diagonal matrix with p entries 1 and q entries -1, i.e. $X = Q_{p+q} |\Lambda_{p+q}|^{1/2}$ where only $p + q$ nonzero eigenvalues of Λ are taken into account. We can define the symmetric bilinear form in $\mathbb{R}^{p+q}$

$$\langle \vec{x}, \vec{y} \rangle_{pq} := \sum_{i=1}^{p} x_i y_i - \sum_{i=p+1}^{p+q} x_i y_i \,.$$

Then, the columns of X constitute vectors $\vec{x}^i$ with pairwise dissimilarities $d_{ij} = \langle \vec{x}^i - \vec{x}^j, \vec{x}^i - \vec{x}^j \rangle_{pq}$. Hence we have found a vector space together with a symmetric bilinear form and an embedding of the points $x^i \mapsto \vec{x}^i$ which yields these dissimilarities under the bilinear form. This embedding

is referred to as *pseudo-Euclidean embedding* of the data points. The values $(p, q, m - p - q)$ are referred to as *signature* of the pseudo-Euclidean space.

Because of this fact, we can always assume that data points are given as vectors $x^j = \vec{x}^j$. However, this embedding need not correspond to standard Euclidean space. The dissimilarity matrix D stems from Euclidean points if and only if $q = 0$, i.e. the matrix which describes the bilinear form does only contain positive diagonal entries (dropping the parts with entry 0 since they obviously do not carry any contribution to the dissimilarity). Otherwise, Euclideanity is violated.

As already mentioned, the code of NG can be executed in every vector space which possesses a symmetric bilinear form such as pseudo-Euclidean space. Further, relational NG can be executed based on a dissimilarity matrix D only. Because of Theorem (2), these two algorithms correspond to each other, i.e. relational NG applied to a symmetric dissimilarity matrix with 0 diagonal is the same algorithm as standard batch NG in pseudo-Euclidean space using the correspondence $\vec{w}^i = \sum_j \alpha_{ij} \vec{x}^j$ for the embedding $\vec{x}^j$ of data as given above. Thereby, relational NG does not rely on explicit coordinates of the data points. The above embedding depends on matrix diagonalization, thus it has cubic complexity. Hence relational NG provides a more efficient scheme than standard NG for the Euclidean as well as non-Euclidean setting. Obviously, out of sample extensions of the found assignments for new data are easily possible as seen in Theorem 9, further the extension to supervised settings as proposed in the last section is immediate.

Connection to a Cost Function

The question now occurs whether a relation of this procedure to the NG cost function and its dual can be made, and whether convergence of the algorithm is guaranteed. If D corresponds to a Euclidean setting, i.e. $q = 0$, the guarantees as given in the last section hold. In particular, RNG converges to a local optimum of the NG cost function and evaluation of these costs are possible based on the dual.

Unfortunately, these guarantees do not hold in general in pseudo-Euclidean space. This property is due to the fact that consecutive steps of batch optimization do not necessarily find optima in the respective step. While assignments of k_{ij} based on the ranks are still optimum in the non-Euclidean setting, assignments $\vec{w}^i = \sum_j h_\lambda(k_{ij}) \vec{x}^j / \sum_j h_\lambda(k_{ij})$ are not. These values can constitute a saddle point or (in case of only negative entries of the bilinear form) even a local maximum of the corresponding part of the cost function. Because of this fact the value of the NG cost function does not necessarily decrease in consecutive steps and convergence is not guaranteed.

A proof that convergence can in general not be guaranteed is given by the following theorem.

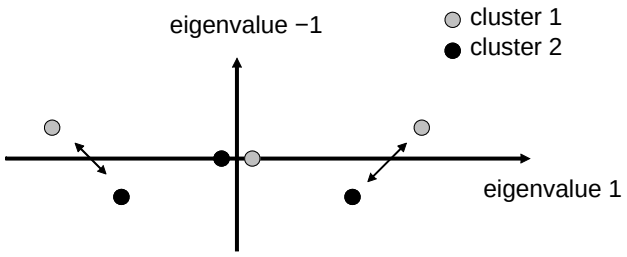

Figure 1: Example of points in pseudo-Euclidean space for which relational clustering does not converge to a fixed point. It is indicated by arrows which points cyclically change their cluster assignments.

Theorem 11 *Assume d_{ij} constitutes a symmetric matrix with diagonal elements 0. Then relational neural gas does not necessarily converge towards a fixed point.*

Proof: Consider the two dimensional pseudo-Euclidean space with signature $(1, 1, 0)$. Consider the points $\vec{x}^1 = (6.1, 1)$, $\vec{x}^2 = (-6.1, 1)$, $\vec{x}^3 = (0.1, 0)$, $\vec{x}^4 = (-0.1, 0)$, $\vec{x}^5 = (4, -1)$, $\vec{x}^5 = (-4, -1)$. The dissimilarity measure $d(\vec{x}, \vec{y}) = (x_1 - y_1)^2 - (x_2 - y_2)^2$ yields the dissimilarity matrix

$$\begin{pmatrix} 0 & 148.84 & 35 & 37.44 & 0.41 & 98.01 \\ 148.84 & 0 & 37.44 & 35 & 98.01 & 0.41 \\ 35 & 37.44 & 0 & 0.04 & 14.21 & 15.81 \\ 37.44 & 35 & 0.04 & 0 & 15.81 & 14.21 \\ 0.41 & 98.01 & 14.21 & 15.81 & 0 & 64 \\ 98.01 & 0.41 & 15.81 & 14.21 & 64 & 0 \end{pmatrix}$$

which is obviously symmetric with diagonal 0 and positive off diagonal elements. We consider two classes i.e. two prototypes of relational NG only. We assume that the neighborhood λ is chosen small enough (e.g. $\lambda < 1$) and we start with the initialization $\alpha_1 = (1/3, 1/3, 1/3, 0, 0, 0)$, $\alpha_2 = (0, 0, 0, 1/3, 1/3, 1/3)$. Then we obtain a cyclic behavior which switches between the two cluster assignments $(1, 1, 1, 2, 2, 2)$ and $(2, 2, 1, 2, 1, 1)$ of points $\vec{x}^1, \ldots, \vec{x}^6$ in subsequent steps. See Fig. 1. □

This example demonstrates the fact that a large contribution of coefficients in the negative axes of the pseudo-Euclidean space can prevent convergence. In this case, the dissimilarity of a data point and a prototype can even become negative although pairwise dissimilarities of data are positive. Further, even if a fixed point is found by relational NG, it need not correspond to a local optimum of the cost function, rather, it can correspond to a saddle point. One example of this behavior is the following situation: consider the three

points $\vec{x}^1 = (0,0)$, $\vec{x}^2 = (1.5,1)$, $\vec{x}^3 = (3,0)$ in pseudo-Euclidean space with signature $(1,1,0)$. The corresponding dissimilarity matrix given by the bilinear form $d(\vec{x},\vec{y}) = (x_1 - y_1)^2 - (x_2 - y_2)^2$ is the matrix

$$\begin{pmatrix} 0 & 1.25 & 9 \\ 1.25 & 0 & 1.25 \\ 9 & 1.25 & 0 \end{pmatrix} .$$

If we choose only one prototype $\vec{w}$, NG converges towards the mean vector $\vec{w} = (3/2, 1/3)$. The quantization error for this solution yields the value 1.9167. If we choose $\vec{w} = \vec{x}^2 = (1.5,1)$, we obtain the better value 1.25. Obviously, in this case, the quantization error is given by a quadratic form which possesses at most one local optimum, which then coincides with the global optimum. Thus, the found solution must be a saddle point. Note, that the assignments are the same in this case, although the prototype is not located at an optimum position.

We will see in our experiments, that, although convergence is not always guaranteed, it can nevertheless be observed for most practical cases; in our experiments, convergence was always given. For fixed points of relational NG, the value of the dual cost function for the found solution corresponds to a value of the standard NG cost function because of Theorem 4. Further, the values which are obtained in these experiments seem reasonably good and the cost function of NG is decreased in most epochs, such that relational NG can be seen as a reasonable heuristic to arrive at a good solution in these cases. Note that non-convex quadratic programming is NP hard as shown e.g. in [61, 52], such that an efficient algorithm which finds optimum prototypes in every epoch instead of the simple mean cannot easily be derived. We can restrict the search space of prototypes to positions which are given by convex combinations $\sum_j h_\lambda(k_{ij})\vec{x}^j$ with permutations k_{ij} if we are interested in an optimization of the dual cost function, because of Theorem (3). However, these values k_{ij} need not coincide with rank assignments for optimum choices. Nevertheless, in practice, this compromise often constitutes a reasonable and efficiently computable tradeoff.

Connection to Deterministic Annealing Approaches

We would like to stress that the fact that convergence of relational NG is not guaranteed is shared by very popular and successful alternatives which have been proposed in the literature in the context of clustering proximity data, namely deterministic annealing for pairwise data clustering and SOM, respectively, as proposed in [30, 18]. Actually, relational neural gas can be understood as a direct simple derivation of the corresponding crisp clustering algorithm which results from the approaches as proposed in [30, 18] in the limit of zero temperature. Thus, because the deterministic annealing

schemes [30, 18] constitute powerful and effective techniques for grouping and clustering proximities, also relational approaches can be seen as an effective compromise to arrive at reasonable solutions of the NP hard optimization problem for general dissimilarity data. We will explain this link and give examples of the divergence of the deterministic annealing schemes in the following section.

Deterministic annealing for pairwise data clustering (DA) is introduced in [30] based on a normalization of the dual k-means cost function:

$$E^{\vee}_{\mathrm{k-means}} \sim \frac{1}{2}\sum_i \sum_k \frac{d_{ik}}{m}\left(\sum_l \frac{M_{li}M_{lk}}{p_l} - 1\right)$$

where m denotes the number of data points, $M_{ij} = \delta_{ij}$ refers to the assignment of point x^j to cluster i, and $p_i = \sum_l M_{il}/m$ denotes the probability of cluster i. The additional summand $\sum_{ik} d_{ik}/(2m)$ yields a constant term which emphasizes the independence of the clustering cost function of the absolute dissimilarity scale. Obviously, up to constant factors and summands, the standard dual k-means cost function is obtained. In the approach [30], a clustering scheme is derived from this cost function by substituting crisp cluster assignments by expectation values for the assignments under a specified certainty level which is parameterized by a temperature T. Using methods of statistical physics which have been pioneered in the context of clustering by Rose et al. under the frame of deterministic annealing [58], the following algorithm is derived, which consists in a subsequent computation of expected assignments $\langle M_{ij}\rangle$ and potentials $\mathcal{E}_{ij}$ as shown in Algorithm 4. $\eta < 1$ determines the decrease of the temperature $T \to 0$.

The formulas as displayed in Algorithm 4 have been derived under the assumption

$$\frac{M_{ik}}{mp_i} = \frac{M_{ik}}{\sum_{k'\neq k} M_{ik'} + 1}$$

as shown in Equation (38) in [30]. If the equation

$$\frac{M_{ik}}{mp_i} = \frac{M_{ik}}{\sum_{k'} M_{ik'}}$$

is used instead which directly results from the definition of p_i, the update formula for $\mathcal{E}_{ij}$ is slightly altered to

$$\mathcal{E}_{ij} = \frac{1}{\sum_{j'}\langle M_{ij'}\rangle}\sum_k \langle M_{ik}\rangle\left(d_{jk} - \frac{1}{2\sum_{j'}\langle M_{ij'}\rangle}\sum_l \langle M_{il}\rangle d_{lk}\right)$$

In the limit of many points $m \to \infty$ the differences between these two update formulas vanish. These latter updates correspond to the update rules of relational k-means in the limit of zero temperature $T \to 0$ where the averages

Algorithm 4: Deterministic Annealing for Pairwise Data Clustering

input

symmetric dissimilarity matrix $D \in \mathbb{R}^{m \times m}$ with zero diagonal;

begin

init $\mathcal{E}_{ij}$, $\langle M_{ij} \rangle$ randomly;

init temperature $T = T_0$;

repeat

repeat

$$\mathcal{E}_{ij} := \frac{1}{\sum_{j' \neq j} \langle M_{ij'} \rangle + 1} \sum_k \langle M_{ik} \rangle \cdot \left(d_{jk} - \frac{1}{2 \sum_{j' \neq j} \langle M_{ij'} \rangle} \sum_l \langle M_{il} \rangle d_{lk} \right);$$

$$\langle M_{ij} \rangle := \frac{\exp(-\mathcal{E}_{ij}/T)}{\sum_i \exp(-\mathcal{E}_{ij}/T)};$$

until convergence

$T := \eta T$;

until $T \leq T_{\text{final}}$;

return assignments $j \mapsto \text{argmax}_i \{\langle M_{ij} \rangle\}$

end.

$\langle M_{ij} \rangle$ become crisp assignments δ_{ij}, and the potentials $\mathcal{E}_{ij}$ correspond to the dissimilarity dist_{ij} of data point x^j and prototype w^i:

$$\text{dist}_{ij} = \sum_k \frac{\delta_{ik}}{\sum_k \delta_{ik}} d_{jk} - \frac{1}{2} \sum_{lk} \frac{\delta_{ik}}{\sum_k \delta_{ik}} \frac{\delta_{il}}{\sum_l \delta_{il}} d_{lk}$$

$$\delta_{ij} = \begin{cases} 1 & \text{if } \text{dist}_{ij} \text{ is minimum} \\ 0 & \text{otherwise} \end{cases}$$

Thus, in the limit, DA indirectly performs k-means clustering in pseudo-Euclidean space using the possibility to compute dissimilarities only indirectly by means of the formula provided in Theorem 1. Prototypes are given indirectly by the coefficients $\delta_{ij} / \sum_j \delta_{ij}$ and can be recovered as $\vec{w}^i = \sum_j \delta_{ij} \vec{x}^j / \sum_j \delta_{ij}$ for an embedding of data in pseudo-Euclidean space $\vec{x}^i$. Instead of a derivation from the dual cost function for pairwise data, DA could alternatively directly be derived from the formulas of deterministic annealing in pseudo-Euclidean space by using Theorem 1 and the ideas of the derivation of relational NG from batch NG. Thus, DA for pairwise proximities is an application of deterministic annealing to relational k-means, to obtain better results

by substituting crisp assignments by their expected values.

Similarly, DA for proximity SOM (DASOM) as proposed in [18] optimizes the dual cost function of the standard SOM objective as proposed by Heskes [29]:

$$E_{\mathrm{SOM}}^{\vee} = \frac{1}{2} \sum_{j,l} \sum_{i,s,u} \frac{h_\lambda(\mathrm{nd}(s,i)) M_{sj} \cdot h_\lambda(\mathrm{nd}(u,i)) M_{uj}}{\sum_{j,i'} h_\lambda(\mathrm{nd}(i',i)) M_{i'j}} \, d_{jl}$$

The resulting algorithm repeats the following assignments in the inner loop:

$$\langle M_{ij} \rangle = \frac{exp(-\mathcal{E}_{ij}/T)}{\sum_i exp(-\mathcal{E}_{ij}/T)}$$

for the expected assignments of data to clusters and

$$\mathcal{E}_{ij} = \sum_{i'} h_\lambda(\mathrm{nd}(i',i)) \sum_k \alpha_{i'k} \left(d_{jk} - \frac{1}{2} \sum_l \alpha_{i'l} d_{lk} \right)$$

where

$$\alpha_{ij} = \frac{\sum_{i'} h_\lambda(\mathrm{nd}(i,i')) \langle M_{i'j} \rangle}{\sum_{i',j'} h_\lambda(\mathrm{nd}(i,i')) \langle M_{i'j'} \rangle}$$

denotes the coefficients of prototypes in pseudo-Euclidean space. The term $\mathcal{E}_{ij}$ corresponds to the distance of data points $\vec{x}^j$ from prototype $\vec{w}^i$ averaged over the local neighborhood provided by SOM. As before, these update formulas can be interpreted as relational versions of deterministic annealing of batch SOM in pseudo-Euclidean space, or alternatively, as deterministic annealing of relational SOM.

The deterministic annealing schemes are derived from their respective cost function using methods of statistical physics: cluster assignments are characterized by a Gibbs distribution depending on the cost terms. A mean field approximation is taken to approximate the Gibbs distribution in factorial form. Partial assignment costs $\mathcal{E}_{ij}$ can be derived thereof by optimizing the Kullback-Leibler divergence between the Gibbs distribution and the factorial approximation. In both cases, optimal potentials $\mathcal{E}_{ij}$ (which relate to prototype positions in pseudo-Euclidean space) are determined based on the derivatives, i.e. only the necessary condition for a local optimum is guaranteed in both cases. In the same way as for relational NG, the sufficient condition on a local optimum is not necessarily fulfilled if a non-Euclidean dissimilarity matrix D is considered since the Hessian is not globally definite in this case. In consequence, there exist situations where deterministic annealing does not converge in the same way as relational clustering.

As an example, we consider the same situation as provided in Theorem 11. Since the update rules of DA (in changed form) and DASOM yield the standard relational k-means update rules, the same behavior as for relational k-means can be observed if initialized appropriately. More precisely, for a temperature at most 0.001 and initialization as $\langle M \rangle = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$,

a cyclic change of this state to the setting $\langle M \rangle \approx \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$ is observed. For the original DA rule, due to the slightly different update, a different behavior holds for this setting. We obtain the same cyclic changes if we start with 4 identical copies of every point, since a large enough number of data m causes essentially the same updates for both DA versions.

The fact that cycles can occur has already been pointed out in [30]. As a consequence, in [30], it is recommended to update the quantities $\mathcal{E}_{ij}$ and $\langle M_{ij} \rangle$ sequentially by picking a random coefficient j (corresponding to a data point x^j) and updating $\mathcal{E}_{ij}$ and $\langle M_{ij} \rangle$ for this j only in every inner loop. As a consequence, cycles which constitute only local traps of the update dynamics can partially be avoided, such as the example as introduced above which does not lead to cyclic behavior for online updates. However, still, cyclic behavior can be present for online updates. As an example, we consider the points $\vec{x}^1 = (1.5, 0)$, $\vec{x}^2 = (1, 2.5)$, $\vec{x}^3 = (-1.5, 0)$, $\vec{x}^4 = (-1, -2.5)$ in two dimensional pseudo-Euclidean space with signature $(1, 1, 0)$. The corresponding dissimilarity matrix is given as

$$\begin{pmatrix} 0 & -6 & 9 & 0 \\ -6 & 0 & 0 & -21 \\ 9 & 0 & 0 & -6 \\ 0 & -21 & -6 & 0 \end{pmatrix}$$

When started in $\langle m \rangle = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$ and corresponding $\mathcal{E}$ with temperature T at most 0.001, the points $\vec{x}^2$ and $\vec{x}^4$ change their expected assignments between $\langle M_{ij} \rangle \approx 0$ and $\langle M_{ij} \rangle \approx 1$ while the other assignments remain constant for DA with altered update rule. Again, the original DA rule behaves slightly differently, but we obtain qualitatively the same cyclic behavior when using four identical copies of every point, as before. Thus, also sequential update can only partially avoid cyclic behavior.

As a consequence of this argumentation, we can interpret relational clustering as the crisp limit case of very popular deterministic annealing variants for pairwise data clustering. Because of its avoidance of the inner loop, it is considerably faster than DA schemes, but the price of probably worse optima is paid. In principle, however, the theoretical guarantees of both methods with respect to convergence are the same since they show limit cycles in comparable situations (which turn out to be rare in practical applications – we did not observe this behavior in a single practical experiment).

The connection of relational clustering to DA schemes allows to interpret DA schemes in the same way as relational variants: DA schemes constitute standard clustering methods in pseudo-Euclidean space. Thus, the methods can be interpreted as prototype-based schemes where prototypes are represented indirectly by means of coefficients. In particular, interpretation of

the methods in terms of data points closest to the prototypes as well as fast extensions to very large data sets in terms of patch clustering, which we introduce later in this article, become possible this way also for the latter algorithms.

Spread Transformation

As an alternative to a direct application of clustering in pseudo-Euclidean space, one can first change the dissimilarity matrix to make it Euclidean, such that convergence of the algorithm is guaranteed. This procedure has been proposed in the context of k-means e.g. in the approaches [41, 55]. In these approaches, it is demonstrated that a theoretical base of this technique is given by the fact that the dual k-means cost function is invariant under additive shifts of the off-diagonal elements of the dissimilarity matrix. The approach [55] shows that this also holds for a few further popular cost functions e.g. connected to muldidimensional scaling or graph cut.

This fact is preserved by the dual cost function of NG for small neighborhood range. More precisely, we find the following result:

Theorem 12 *Assume pairwise distances d_{ij} are given such that the corresponding matrix D is symmetric with zero diagonal. Assume $\epsilon > 0$. Consider the distances with shifted off-diagonal terms $\tilde{d}_{ij} = d_{ij} + d_0(1 - \delta_{ij})$ for $d_0 \geq 0$ where δ_{ij} denotes Kronecker delta. Assume the neighborhood range is chosen as $\lambda \leq 1/\ln(md_0k/(2\epsilon))$. Then the distance of the dual cost function of neural gas for d_{ij} and $\tilde{d}_{ij}$ can be estimated as*

$$|E_{\mathrm{NG}}^{\vee}(k_{ij}, d_{ij}) - E_{\mathrm{NG}}^{\vee}(k_{ij}, \tilde{d}_{ij}) + \frac{1}{4}(Cm - k)d_0| \leq \epsilon$$

where $C = \sum_{i=0}^{k-1} h_\lambda(i)$.

Proof: We find

$$\begin{aligned} & E_{\mathrm{NG}}^{\vee}(k_{ij}, \tilde{d}_{ij}) \\ = & \sum_i \frac{1}{4\sum_j h_\lambda(k_{ij})} \cdot \sum_{jj'} h_\lambda(k_{ij}) h_\lambda(k_{ij'})(d_{jj'} + d_0) \\ & - \sum_i \frac{1}{4\sum_j h_\lambda(k_{ij})} \cdot \sum_j h_\lambda(k_{ij}) h_\lambda(k_{ij}) d_0 \\ = & E_{\mathrm{NG}}^{\vee}(k_{ij}, d_{ij}) + \frac{1}{4} \cdot Cmd_0 \\ & - \sum_i \frac{1}{4\sum_j h_\lambda(k_{ij})} \cdot \sum_j h_\lambda(k_{ij}) h_\lambda(k_{ij}) d_0 \end{aligned}$$

The latter term can be decomposed into the sum over all prototypes for which at least one $k_{ij} = 0$ exists, and the remaining prototypes, which are not winner for a data point. We obtain for the first term

$$\begin{aligned}
&\frac{d_0}{4} \sum_{i:\exists k_{ij}=0} \frac{\sum_j h_\lambda(k_{ij})^2}{\sum_j h_\lambda(k_{ij})} \geq \\
&\frac{d_0}{4} \sum_{i:\exists k_{ij}=0} \frac{\sum_{j:k_{ij}=0} h_\lambda(k_{ij})^2}{\sum_{j:k_{ij}=0} h_\lambda(k_{ij}) + \sum_{j:k_{ij}\neq 0} h_\lambda(k_{ij})} \geq \\
&\frac{d_0}{4} \cdot \frac{k}{1 + m \cdot \exp(-1/\lambda)} \geq \frac{d_0}{4} \cdot k - \frac{\epsilon}{2}
\end{aligned}$$

for $\lambda \leq 1/\ln(md_0k/(2\epsilon))$, and for the second term

$$\begin{aligned}
&\frac{d_0}{4} \sum_{i:\nexists k_{ij}=0} \frac{\sum_j h_\lambda(k_{ij})^2}{\sum_j h_\lambda(k_{ij})} \\
\leq \quad &\frac{d_0}{4} \sum_{i:\nexists k_{ij}=0} \exp(-1/\lambda) \frac{\sum_j h_\lambda(k_{ij})}{\sum_j h_\lambda(k_{ij})} \leq \frac{\epsilon}{2}
\end{aligned}$$

for $\lambda \leq 1/\ln(d_0k/(2\epsilon))$. Therefore, substituting these terms by $d_0k/4$ changes the result by at most ϵ. □

Thus, a transformation of the dissimilarity matrix does not change the shape of the dual cost function and the location of local and global optima for small λ. One can see that, for large enough d_0 a dissimilarity matrix results which stems from the squared Euclidean metric. This procedure has been introduced in the literature under the notion *spread transformation* e.g. in connection to relational fuzzy clustering [27] or *constant shift embedding* in the connection of k-means clustering [55]. The following result is well known in the literature and follows immediately from the embedding of data in pseudo-Euclidean space (e.g. [53]):

Theorem 13 *Assume D is a symmetric dissimilarity matrix with zero diagonal. Then*

$$\tilde{D} = D - 2\Lambda_m(G)$$

is squared Euclidean, i.e. there exist Euclidean points $\vec{x}^i$ with $\tilde{d}_{ij} = \|\vec{x}^i - \vec{x}^j\|^2$ for all i, j, where $G = -1/2(I - 1/m\vec{1}\vec{1}^t)D(I - 1/m\vec{1}\vec{1}^t)$ denotes the Gram matrix used for the pseudo-Euclidean embedding of the points, and $\Lambda_m(G)$ denotes the smallest eigenvalue of G. This shift denotes the smallest possible value to achieve this property.

The result is obvious because this shift corresponds to a shift of the Gram matrix of the form $G - \Lambda_m(G)$. This procedure has been proposed e.g. in [41, 55]. Since only the smallest eigenvalue of G is needed, this procedure is

much more efficient than an explicit embedding and correction of the non-Euclideanity, which would require cubic effort. Further, unlike alternatives to move a general dissimilarity matrix in squared Euclidean form as proposed e.g. in [10], the spread transform does not affect the cost function. In fuzzy clustering, it has been proposed to use spread-transformation if and only if distances of data points and prototypes become negative [27].

Unfortunately, it turns out that this procedure is partially of theoretical interest for some practical problems. The reason lies in the fact that this transformation can make the classification problem harder for the standard methods such as relational NG and deterministic annealing of pairwise proximities. While the transform does not affect the location of local optima, it changes the numeric of the cost function and the relative differences between good and bad local optima of the cost function such that these methods can no longer easily find good local optima. Hence, the direct application of the relational models to non-Euclidean data can give better results (although convergence is not guaranteed in theory, but can usually be observed in practice) than the application of the corresponding method in the shifted Euclidean space (although convergence is guaranteed in the latter case).

We demonstrate this effect in a standard benchmark data set. The cat cortex data originates from anatomic studies of cats' brains. The dissimilarity matrix displays the connection strength between 65 cortical areas [18]. For our purposes, a preprocessed version as presented in [25] was used. The matrix is symmetric with zero diagonal, but the triangle inequality does not hold. The signature of the related pseudo-Euclidean space is $(41, 23, 1)$, i.e. about one third of the directions are associated to negative eigenvalues. The corresponding eigenspectrum is depicted in Fig. 2. We applied the spread transform to this data set according to Theorem 13. We trained (batch) deterministic annealing and relational neural gas on these data sets, using 5 prototypes in each case and default parameters (for deterministic annealing: number of epochs of the outer loop = 300, number of epochs of the inner loop = 50, start temperature = 100, noise level added to distances to avoid identical prototypes = 10^{-6}, for relational neural gas: number of epochs = 100, start neighborhood range = 2.5). Fig. 3 shows the quality measured by the dual quantization error of the found optima when repeatedly initializing the algorithms with small random values for the data assignments obtained over 1000 runs. A curve with a well expressed maximum results representing the most prominent optimum. As expected, this optimum is a bit better for deterministic annealing than for relational neural gas due to the soft assignments of the data points, which result in a slightly better optimum at the cost of a slower convergence speed and the necessity of an additional inner loop. When considering the spread transformed data which correspond to an Euclidean representation, the optima are shifted to the right in both cases. Thereby, we evaluate the result based on the found assignments only, taking the original dissimilarities, i.e. we evaluate the quality of the

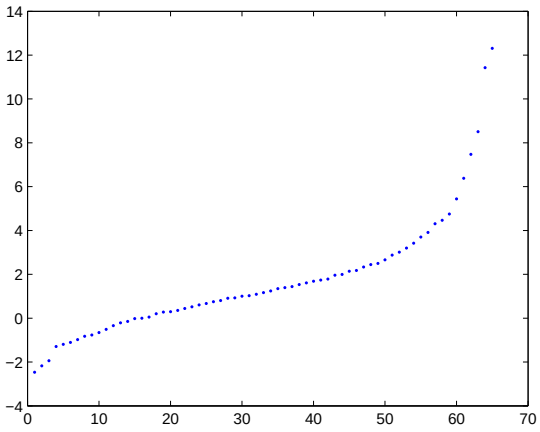

Figure 2: Eigenspectrum of the cat cortex data set when embedded into pseudo-Euclidean space

found clustering for the original clustering problem in both cases. Hence, although the algorithms deal with a Euclidean setting instead of an only pseudo-Euclidean one in these cases, the optima found by the algorithms are worse than the original ones. This behavior is quite typical and it can be observed also for other non-Euclidean data sets. Hence, a direct application of relational neural gas to the dissimilarity matrix instead of a prior spread transformation of the data can be advisable.

4 Patch Clustering

In recent years, the problem of mining large data sets has become one of the central issues of data mining. Roughly, the amount of electronically available data doubles every 20 months reaching almost every area of daily life and science, such that people have to cope with massive data sets which cannot be scanned manually. Clustering and visualization offers one of the fundamental techniques to adequately compress and preprocess such enormous data sets. However, in these cases, data do no longer fit into main memory such that batch processing methods which rely on all data at once become infeasible. Further, at most one scan through the data is still affordable, which also makes online alternatives such as online neural gas unsuitable. The situation is still worse, since clustering methods for dissimilarity data rely on the quadratic dissimilarity matrix, i.e. they display at least

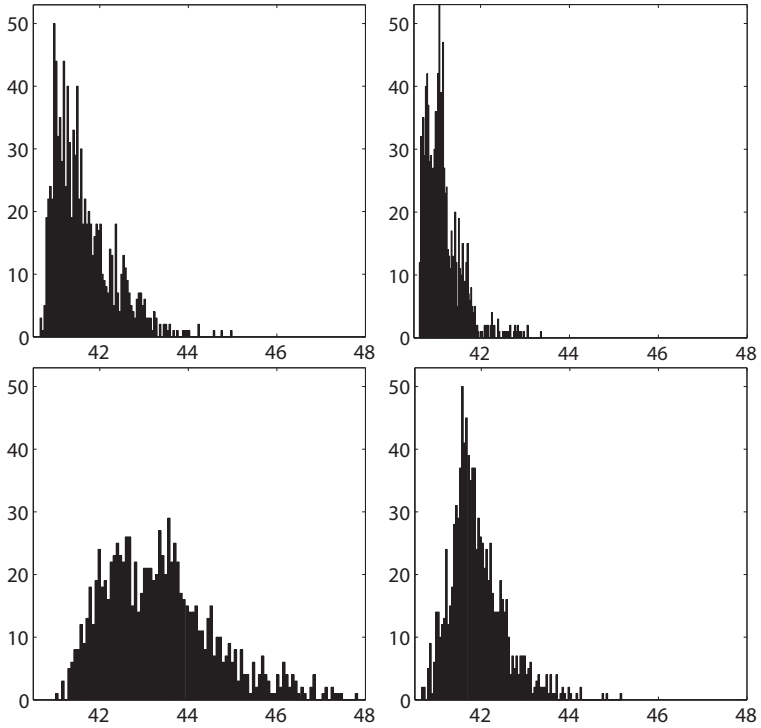

Figure 3: Local optima of the dual k-means cost function reached by repeated runs of relational neural gas (left column) and deterministic annealing (right column) on the cat cortex dataset for the original cat cortex data set (top) and its spread transformation (bottom), respectively.

quadratic complexity and, as is the case of relational clustering and deterministic annealing, linear space complexity for the classifier. Both issues make the methods slow for settings which reach ten thousand data points, and entirely unsuitable for common desktop computers available today if more than a hundred thousand data points are involved.

Due to these problems a variety of methods which introduce clustering algorithms for streaming data have been proposed in the literature, which ideally work in linear time and constant space. Thereby, most of the approaches have been proposed for Euclidean data. This includes extensions of classical k-means clustering or k-median clustering, partially incorporating approximation guarantees such as e.g. [3, 40] which reach linear time, but for which space requirements depend on the number of points, or heuristics which partially rely on sampling and according statistical guarantees or

grid based methods, such as e.g. popular algorithms dedicated to very large data sets as CURE, STING, and BIRCH [14, 19, 66, 70], or iterative compression approaches which process only a fixed subset of the given data at a time [8, 15, 1].

A few proposals for large sets of general non-Euclidean dissimilarity data exist. Hierarchical clustering have typically a squared complexity, which can partially be optimized. However superliner complexity is still kept and these methods are typically not very stable with respect to noise [49]. The method [38] extends the standard SOM to dissimilarity data by means of the generalized median and it tackles large data sets by simple subsampling of the data. Affinity propagation [16] also relies on median clustering by restricting prototype locations to data points, and it can be used for large data sets if the connection matrix is sparse since only existent connections are effectively used in this setting. However, both methods are restricted due to restricted prototype locations, and they require several sweeps through the whole data set. Two approaches which require only one sweep are given in [6, 69]. [6] relies on relational variants of fuzzy k-means clustering and extends this to large data sets by subsampling a characteristic part of the dissimilarity matrix, clustering this subpart, and extending it to all data. This way, however, the full matrix has to be available in advance or, alternatively, data must be i.i.d. to obtain reliable extensions to new parts. Similarly, the Nyström approximation constitutes a popular vehicle to extend the results of a part of a dissimilarity or kernel matrix to the full data set, where approximation bounds can be derived explicitely. The work [5] constitutes one proposal where this approach has been used in the context of graph clustering for general dissimilarity graphs. However, a representative sample has to be available which allows the extension of the clustering to the full data set. In contrast, the approach [69] proposes to process only parts of the given data on parallel processors by a direct optimization of pairwise clustering and to subsequently reach valid assignments of all data this way. However, the presented method does no longer represent the solution in form of interpretable prototypes.

Here we rely on patch clustering as introduced in [1, 15, 8] for Euclidean data sets to extend prototype based clustering methods for dissimilarity data to large or streaming data sets. The basic idea is to iteratively process only a small part of the data using standard k-means or neural gas, and to store these data in compressed form in terms of the prototypes and their multiplicity. This serves as sufficient statistics for further runs. Subsequent runs cluster the compressed data points which were already seen in form of the prototypes counted with multiplicities in addition to the next patch.

Patch Definition

To transfer this method to dissimilarity data, we assume the following setting: A (possibly large) set of points x_i indexed $i = 1, 2, \ldots, m$ is given such that, for all i and j, the dissimilarity d_{ij} between these points can be computed directly. D denotes the corresponding dissimilarity matrix where we assume symmetry $d_{ij} = d_{ji}$ and zero diagonal $d_{ii} = 0$ as before. A typical example of this setting is a data base of strings for which pairwise comparisons are given by alignment. For large m, it is in general infeasible to compute or store the full dissimilarity matrix in main memory due to the squared complexity. Patch processing relies on the principle to process data in n_p patches of priorly fixed size $p = m/n_p$. Thereby, we assume divisibility of m by n_p for simplicity. In practice, the last patch is of smaller size. For dissimilarity data, a patch P_t is then represented by the corresponding portion of the dissimilarity matrix D:

$$P_t = (d_{sl})_{s,l=(t-1)\cdot p+1,\ldots,t\cdot p} \in \mathbb{R}^{p\times p}$$

which represents the dissimilarities of points $(t-1)\cdot p+1, \ldots, t\cdot p$.

K-Approximation

The idea of original patch clustering is to add prototypes from the processing of the former patch P_{t-1} counted with multiplicities according to the size of their receptive field as additional data points to the current patch P_t. These points play the role of a compressed representation of all already seen data points, i.e. they provide a sufficient statistics of the information processed so far. This way, all data are processed without loss of essential information since the previous information is represented by the sufficient statistics.

A naive transfer of this method from the Euclidean case to relational clustering, however, is not possible due to two reasons: unlike patch processing for Euclidean data, prototypes correspond to a weighting of all points involved in the clustering. Thus, the dimensionality of the coefficient vectors is determined by the number of number of data points which have to be processed. This results in an infeasible linear space complexity for huge data sets. In addition, for further processing of the data, the dissimilarities in between all prototypes and all data from a new patch have to be computed. Since prototypes are represented indirectly by the contribution of all data points seen so far, the distance of prototypes and a new patch relies on the distance of the new patch and all data seen so far. By induction, one can see that this processing is therefore only possible if the full dissimilarity matrix D is available. Hence, this approach results in infeasible quadratic time and space complexity.

Because of this fact, an approximation scheme is introduced which substitutes the full vector of coefficients which characterize the prototypes by only the K most prominent ones, K being a fixed number. Every patch is

clustered until convergence, i.e. a small neighborhood range $\lambda \to 0$ is used in the resulting prototype representations. Obviously, for the coefficients α_{ij} of the prototypes, the following is valid:

$$\alpha_{ij} = h_\lambda(k_{ij}) / \sum_j h_\lambda(k_{ij}) \to \begin{cases} 1/|R_i| & \text{if } k_{ij} = 0 \\ 0 & \text{otherwise} \end{cases}$$

where R_i denotes the receptive field of prototype i. Thus, the coefficient vector α_{ij} yields $1/|R_i|$ for all data points $\vec{x}_j$ for which the winner is the point i. Thereby, for simplicity, we assume that $R_i \neq \emptyset$, which is usually the case for NG schemes. If $R_i = \emptyset$, then α_{ij} is nonvanishing if prototype i is the second closest neuron of $\vec{x}_j$. If this set is also empty, the third closest points determine the nonzero entries of α_{ij} and so on.

We approximate the prototype $\vec{w}^i = \sum_j \alpha_{ij} \vec{x}^j$ by the closest K points in R_i, i.e. the points $\vec{x}^j$ where $k_{ij} = 0$ and $\vec{x}^j$ is among the K points with smallest dissimilarity $d(\vec{w}^i, \vec{x}^j)$ as computed in relational neural gas. Note that more complex alternatives could be possible which choose K points and coefficients such that the corresponding prototype location changes as little as possible as described in [59]. However, the above simple approximation scheme will already lead to good results as we will see in experiments.

These considerations give rise to the definition of a K-approximation of a relational prototype. Assume a prototype $\vec{w}^i = \sum \alpha_{ij} \vec{x}^j$ is given with $\sum_j \alpha_{ij} = 1$. A *K-approximation* refers to the K indices $j_1, \ldots, j_K$ corresponding to points $\vec{x}^{j_1}, \ldots, \vec{x}^{j_K}$ with smallest dissimilarity to $\vec{w}^i$. For optimum prototypes as computed by relational NG in the limit phase, these points are the K closest points in the receptive field of $\vec{w}^i$. Obviously, for these points, the coefficient α_{ij} is maximum. The k-approximation can be computed easily in relational NG since the dissimilarities of data points and neurons are readily available.

Note that, by always restricting to the K closest data points, K being a priorly fixed number, prototypes can always be approximated in constant space while processing all data. Further, only a fixed portion of the global dissimilarity matrix D is necessary to compute dissimilarities in between prototypes and further patches.

Extended Patches

More precisely, we describe the parts of the dissimilarity matrix which are needed for further processing of patches and K-approximated prototypes. Assume the current patch P_t is considered. Assume N_{t-1} refers to the index set of the K-approximation of all prototypes obtained in the previous step. When considering k prototypes, the size of this set is restricted by $|N_{t-1}| \leq k \cdot K$, under the assumption that at least K points lie in every receptive field,

equality $|N_{t-1}| = k \cdot K$ holds. For the next round of patch processing, dissimilarity clustering is applied to the points corresponding to the indices in N_t and the data from the current patch, i.e. we need the following part of the dissimilarity matrix

$$P_i^* = \begin{pmatrix} d(N_{t-1}) & d(N_{t-1}, P_t) \\ \\ d(N_{t-1}, P_t)^t & P_t \end{pmatrix}$$

where $d(N_{t-1}) = (d_{uv})_{u,v \in N_{t-1}}$ denotes the inter-dissimilarities of points from the K-approximation, and $d(N_{t-1}, P_t) = (D_{uv})_{u \in N_{t-1}, v=(t-1)\cdot p+1,\ldots,t\cdot p}$ denotes the dissimilarities of points in the K-approximation and the current patch. We refer to P_i^* as *extended patches*.

Patch Relational Neural Gas

Based on these data handling techniques, patch relational neural gas can be defined as iterative processing of patches enriched by the K-approximation of prototypes from the previous patch. The prototypes contribute to the new clustering task according to the sizes of their receptive fields, i.e. a prototype $\vec{w}^i$ is counted with multiplicity $|R_i|$. Correspondingly, every point $\vec{x}^j$ in N_{t-1} contributes according to the fraction $|R_i|/K$ if it lies in the receptive field of $\vec{w}^i$, i.e. $k_{ij} = 0$. Hence, we set the multiplicity $m_j = |R_i|/K$ where $\vec{x}^i$ lies in the receptive field of $\vec{w}^j$. It is straightforward to extend relational neural gas to deal with multiplicities m_j of point $\vec{x}^j$ corresponding to the underlying cost function $\sum_{ij} h_\lambda(k_i(\vec{x}^j)) \cdot m_j \cdot d(\vec{x}^j, \vec{w}^i)$. The only change concerns the update of the coefficients α_{ij}, see Algorithm 5. This algorithm can be used as internal loop for patch processing for dissimilarity data as shown in Algorithm 6. Thereby, prototypes W are always represented by an index vector corresponding to the data points which contribute to this prototype, and a coefficient vector which specifies the strength of the contributions. Unlike the coefficients α_{ij} in full relational NG, the coefficient vector of patch NG is sparse and it can be represented in constant space. Note that we assume that the relevant parts of the dissimilarity matrix can be computed on demand such that the full dissimilarity matrix need not be computed nor stored at the beginning of the algorithm. This fact constitutes a further advantage of patch processing, since it is sufficient to compute only a linear part of the dissimilarity matrix. Since, depending on the application scenario, the dissimilarity computation can be quite demanding (e.g. alignment of sequences in ioinformatics, or the normalized compression distance for text processing), this can result in drastic computational savings.

After processing, a set of prototypes together with a reasonable K-approximation thereof is obtained which compresses the full data set. As before, an inspection of prototypes is easily possible by looking at the points which are closest to these prototypes.

Algorithm 5: Relational NG with Multiplicities

input

symmetric dissimilarity matrix $D \in \mathbb{R}^{m \times m}$

multiplicities $\{m_1, \ldots, m_m\} \in \mathbb{N}$

begin

init α_{ij} with $\sum_j \alpha_{ij} = 1$;

repeat

compute $\mathrm{dist}_{ij} := [D\alpha_i]_j - \frac{1}{2} \cdot \alpha_i^t D \alpha_i$;

set $k_{ij} := |\{l \mid \mathrm{dist}_{lj} < \mathrm{dist}_{ij}\}|$;

set $\alpha_{ij} := m_j \cdot h_\lambda(k_{ij}) / m_j \cdot \sum_j h_\lambda(k_{ij})$;

until convergence;

return α_{ij};

end.

Note that the algorithm runs in constant space if the size p of the patches is chosen independently of the data set size m. Similarly, under this assumption, the fraction of the distance matrix which has to be computed for the procedure is of linear size $\mathcal{O}(m/p \cdot p) = \mathcal{O}(m)$ and the overall time complexity of patch clustering is of size $\mathcal{O}(m/p \cdot p^2) = \mathcal{O}(mp) = \mathcal{O}(m)$, assuming constant p. Hence, a linear time and constant space algorithm for general dissimilarity data results which is suited for large data sets, if constant patch size is taken. In the experiments, we will demonstrate an application to a data set of size almost 200,000, which corresponds to a full dissimilarity matrix for which the storage would require almost 251 GB, assuming double precision.

This way, a linear time and constant space clustering algorithm is obtained which can deal with general dissimilarity data. Since it relies on only a linear part of the full dissimilarty matrix, the complexity of data preprocessing, i.e. the computation of probably complicated pairwise dissimilarities such as alignment distances, is also greatly reduced. Further, the algorithm provides an explanation of the clustering in terms of prototypes which can be reprented by a finite number of representative data points, hence the result can be directly inspected by human experts.

Algorithm 6: Patch Relational NG

begin

cut the first patch P_1;

apply relational NG to $P_1 \rightarrow$ prototypes W_1;

compute the K-approximation N_1 of W_1;

update multiplicities m_i of N_1;

set $t = 2$;

repeat

cut the next patch P_t;

construct extended patch P_t^* using P_t and N_{t-1};

set multiplicities of points in P_t to $m_i = 1$;

apply relational NG with multiplicities to $P_t^* \rightarrow$ prototypes W_t;

compute K-approximation N_t of W_t;

update multiplicities m_i of N_t;

$t := t + 1$;

until $t = n_P$

return prototypes W_{n_P};

end.

5 Experiments

We demonstrate the behavior of relational NG in a variety of experiments whereby we mainly focus on benchmark data sets which cannot be embedded in Euclidean space. Since convergence of relational NG or the monotonicity of the cost function during training is theoretically not guaranteed in such situations, we have a look at the development of the cost function values in a typical setting, first. Afterwards, we evaluate relational NG in comparison to deterministic annealing on a variety of benchmark data sets, showing competitiveness of the algorithm. For patch NG, we first demonstrate the robustness of the algorithm with respect to the patch size, the quality of the K-approximation, and the order of the presentation of patterns in comparison to full batch clustering. Afterwards, we exemplarily show an application to a huge text corpus, demonstrating the efficiency of the method for large data sets.

Convergence

For all data sets considered in the experiments, convergence of relational NG was observed. We exemplarily depict the behavior of relational NG for the cat cortex data set as introduced above using 5 neurons and 100 epochs. Fig. 4 (top) displays the value of the NG cost function and its dual based on the rankings k_{ij} and the prototypes $\sum_j \alpha_{ij}\vec{x}^j$, respectively, as computed by NG. Obviously, the two cost functions are strictly monotonic and convergent, and, apart from the first steps, they coincide for the computed values. Similarly, the vector quantization cost function and its dual computed for the assignments and prototypes as given by relational neural gas are strictly monotonic and convergent as can be seen in Fig. 4 (bottom). Due to the fact that the quantization error is computed for the (in terms of k-means) suboptimum prototypes which incorporate neighborhood smoothing, while the dual costs are determined on the assignments only (implicitly assuming optimum positions of the prototypes in terms of k-means), the quantization error is worse compared to the value of the dual cost function for early stages of training which display a large neighborhood cooperation.

Experiments on Benchmark Data Sets

In addition to the cat cortex data set as described above, we consider the following benchmark data sets, which were symmetrized prior to training and linearly transformed from similarities to dissimilarities, if necessary:

- **Protein data:** The protein data set as described in [48] consists of 226 globin proteins which are compared based on their evolutionary distance. The samples originate from different protein families: hemoglobin-α, hemoglobin-β, myoglobin, etc. Here we distinguish five classes as proposed in [25]: HA, HB, MY, GG/GP, and others. Unlike the other data sets considered here, the protein data set has a highly unbalanced class structure, with class distribution HA (31.86%), HB (31.86%), MY (17.26%), GG/GP (13.27%), and others (5.75%).

- **Copenhagen chromosomes data:** The Copenhagen chromosomes data set is a benchmark from cytogenetics [42]. 4200 human chromosomes from 22 classes (the autosomal chromosomes) are represented by the gray levels of their images. These images are transferred to strings based on their thickness. These strings can be compared using edit distance which constitutes a typical dissimilarity measure for strings [33]. The substitution costs are thereby given by the difference of the entries and insertion/deletion costs are set to 4.5 [50].

- **Aural sonar data:** The aural sonar data set as described in [10] consists of 100 returns from a broadband active sonar system, which are

	Signature	minimum value spread transform
cat cortex	(41,23,1)	4.93
chromosomes	(1951,2206,43)	851.79
protein data	(218,4,4)	$2.6 \cdot 10^{-14}$
aural sonar	(54,45,1)	2.1
caltech	(8664,12,1)	$1.67 \cdot 10^{-14}$
face recognition	(311,310,324)	$7.9 \cdot 10^{-4}$
patrol	(173,67,1)	4.33
voting	(105,235,95)	$3.2 \cdot 10^{-4}$

Table 1: Signature of the benchmark data set and minimum absolute value of the spread transform to obtain squared Euclidean distances

labeled in two classes, target-of-interest versus clutter. The dissimilarity is scored by two independent human subjects each resulting in a dissimilarity score in $\{0, 0.1, \ldots, 1\}$.

- **Caltech data:** The caltech data set consists of 8677 images from 101 object categories which are compared using the pyramid match kernel on SIFT features, see [10].
- **Face recognition data:** The face recognition data set consists of faces of 139 people. Dissimilarities are computed by means of cosine similarity between integral invariant signatures based on surface curves of the 3D-faces, see [10].
- **Patrol data:** The patrol data set describes 241 members of seven patrol units and one class corresponding to people not in any unit. Dissimilarities are computed based on every person in the patrol units naming five other persons in their unit, whereby the responses were partially inaccurate. Every mentioning yields an entry of the dissimilarity matrix, see [10]. Data are sparse in the sense that most entries of the matrix correspond to the maximum dissimilarity which we set to 3.
- **Voting data:** The voting data set describes a two-class classification problem incorporating 435 samples which are given by 16 categorical features with 3 different possible values each. The dissimilarity is determined based on the value difference metric, see [10].

These data sets are non-Euclidean with signature as given in Tab. 1. Obviously, the protein data and the caltech data set are almost Euclidean, while at least one third of the eigenvalues is negative for the other data sets.

We performed a repeated cross-validation for all data sets, using ten repeats. We report the results of relational neural gas (RNG) and supervised

	number of neurons	number of folds
cat cortex	12	2
chromosomes	60	2
protein data	20	10
aural sonar	10	2
caltech	103	2
face recognition	139	10
patrol	24	10
voting	20	10

Table 2: Number of neurons and number of folds used for the runs.

relational neural gas (SRNG) for these data sets. For relational neural gas, 100 epochs where used; for the supervised version, the supervision parameter equals $\beta = 0.5$. For comparison, we report the results of deterministic annealing (DA); here, 300 epochs where used for training. The number of prototypes used for every data set and the number of folds are reported in Tab. 2. Results are reported on the training and test set. The runs are evaluated by the classification accuracy obtained by posterior labeling on the training set. In addition, the quantization error and the value of the dual k-means cost function are reported. Thereby, relational prototypes allow an out-of-sample extension for both, relational neural gas and deterministic annealing as discussed previously. Since we can interpret deterministic annealing as annealing of clustering in pseudo-Euclidean space, out of sample extensions for deterministic annealing can be obtained in the same way.

The results of the runs are reported in Tab.3. Interestingly, the value of the dual cost function, the quantization error, and the classification accuracy of RNG is always competitive to the value obtained by DA, although the latter method requires more training time due to an additional inner loop. As expected, the values of SRNG for the unsupervised cost functions are a bit worse, since this method alters the objective to better take label information into account. This corresponds to an improvement of the classification accuracy on the training data for all but one cases.

Since supervised label information is available for these datasets, it is possible to compare the results to the behavior of supervised training algorithms such as SVM or k-nearest neighbor for these data sets. The last six datasets have recently been considered in [10] as benchmarks where different supervised techniques to deal with dissimilarity data including SVM with various preprocessing and kernels and k-nearest neighbor have been compared. Interestingly, errors of 13-17.75% are reported for the aural data set using the different methods, and errors of 4.89-5.8% are reported for the voting data set, placing SRNG as a method with best classification accuracy for both sit-

uations. For the other datasets, errors of 1.86-30.35% (protein data), 29.9-41.99% (caltech), 3.92-4.55% (face recognition), and 11.56-42.19% (patrol) are reported. Thus, SRNG is competitive for the protein data set, too. For caltech, and patrol, it achieves a considerable accuracy on the training set, but not generalizing properly to the test set, while clustering seems not suited for the supervised classification task specified in the face recognition data. Nevertheless, the aim of clustering as measured by the quantization error and the dual cost function, gives reasonable results in all cases, clearly demonstrating the competitiveness of RNG methods to deterministic annealing schemes.

Demonstration of Patch Clustering

Patch clustering extends RNG towards huge data sets at the costs of decreased accuracy due to the compression of data in early patches in form of prototypes, and due to the approximation of relational prototypes by only the most important coefficients. Thereby, both, the patch size and the number of coefficients used for the approximation are parameters which can be chosen to balance the accuracy of the results (this is high for large patch size) and the required space and speed of computation (which is small for small patch size and small K for the approximation). The effect of these choices is reported in Fig. 5. The Kopenhagen chromosomes data is taken trained with patch RNG and 60 neurons for 100 epochs. Thereby, the number of patches was varied from 1 to 10, and the K used for the K-approximation was varied from 1 to 5. The reported result is the classification accuracy obtained on a test set for a repeated 10-fold cross-validation with 10 repetitions. Using 10 patches (corresponding to a speed-up 10 of the computation) and using a K approximation with $K = 2$ leads to a reduction of the hit rate of about 3.3%, using a K approximation with $K = 1$ leads to a reduction of the hit rate of about 4.5%. Hence, patch approximation leads to an only slight decrease of the quality of the found solution.

For huge data sets, data can often be accessed only sequentially, such that data are not i.i.d. with respect to the underlying distribution. It is interesting to investigate whether this fact has consequences on the accuracy of patch clustering. As already demonstrated in [1] for Euclidean settings, this is not the case since prototypes accurately represent already seen data, following trends accurately, if necessary. In [1], a data set with a strong trend was created and presented to patch clustering following the trend. The result of the overall clustering which was achieved after iterative patch processing was virtually indistinguishable from the result of NG when applied to all data at once or patch NG with i.i.d. data due to the ability of patch NG to use all previous information in terms of the sufficient statistics.

For the non-euclidean setting, it is not as obvious how to create data with a strong trend to test the abilities of patch clustering to deal with such settings. Therefore, we rely on auxiliary information as given by the data labels. We

compare patch clustering with randomly presented samples to patch clustering where samples are sorted according to the label information such that, in early patches, a highly unbalanced data set is presented. The results as achieved for the chromosomes data set are reported in Tab. 4. Thereby, a 2-fold crossvalidation was repeated 10 times, using 10 patches (corresponding to about 420 data points per patch), 60 neurons, 100 epochs per patch, and an approximation of relational prototypes by 3 coefficients. For comparison, the result of full batch RNG using the same splits for the repeated crossvalidation are reported. Obviously, the differences are negligible, i.e. the order of the presentation does not severely influence the output in this case.

Processing Large Text Data

To demonstrate the ability of patch clustering to deal with large dissimilarity data, a data set was generated in the same way as the 20 newsgroup data set from the UCI repository [2]. 183,546 articles from 13 different newsgroups were taken, where the distribution of the articles is displayed in Tab. 5. The texts were preprocessed by removing stop words and applying word stemming [54]. Comparisons of two texts took place using the normalized compression distance (NCD) as proposed in the approach [11]. The NCD is an approximation of a universal distance for strings based on Kolmogorov complexity. It is defined as

$$\mathrm{NCD}(x, y) = \frac{C(xy) - \min\{C(x), C(y)\}}{\max\{C(x), C(y)\}}$$

where x and y are the document strings, xy its concatenation, and $C(x)$ denotes the size of a compression of the string x. For our experiments, the bzip2 compression method was used.

We would like to demonstrate the ability of our method to project large data sets into the Euclidean plane in reasonable time. Therefore, we report the result of a supervised relational hyperbolic SOM (HSOM), which offers particularly powerful visualization facilities of possibly complex data sets due to the flexibility of the hyperbolic space [51]. Obviously, batch processing for HSOM can be transferred to a relational version in the same way as batch NG. The goal is to map the full data set to the hyperbolic lattice structure by means of supervised HSOM. Patch processing was applied using supervised relational HSOM with 85 neurons and a 3 approximation of the prototypes. Since the full dissimilarity matrix would occupy approx. 250 GB space, it is no option to process the data at once or precalculate the full dissimilarity at once. Instead, 183 patches of around 1000 texts were taken and the distances of these patches were precalculated, resulting in only around 12 MB space. In addition, since patch processing requires the dissimilarities of the extended patches, around $1000 \cdot 85 \cdot 3$ dissimilarities had to be computed on demand

for every patch. This way, the whole computation could be performed on a common desktop computer in reasonable time.

The outcome is depicted in Fig. 6. Thereby, the hyperbolic lattice structure is mapped onto Euclidean plane for visualization and data inspection. For interpretability, neurons were posteriorly labeled according to a majority vote. Clearly, the data arrange on the lattice according to their semantic meaning as mirrored by the corresponding newsgroup, such that a visualization and browsing becomes possible.

6 Discussion

The focus of this article was put on clustering algorithms for general dissimilarity data which can be derived from the standard vector quantization cost function and extensions thereof. For this purpose, we have introduced relational clustering as an equivalent formulation of batch clustering for neural gas or self organizing maps if Euclidean data are given in terms of pairwise dissimilarities only. While convergence can be guaranteed in Euclidean settings only, applicability of the algorithms is granted for every possible symmetric dissimilarity measure. Further, a variety of properties is still valid such as easy out-of-sample extensions and interpretability in terms of the dual cost function for fixed points of the algorithms. This way, we arrived at relational neural gas and self-organizing maps for arbitrary dissimilarity data. We have seen that the algorithms can be interpreted as clustering in pseudo-Euclidean space whereby a reasonable heuristic is taken to arrive at good prototype locations. The algorithms yield competitive results to alternatives such as deterministic annealing and they often show better behavior than theoretical alternatives with guaranteed convergence such as spread transformation of the data. Thus, relational neural gas and alternatives can be taken as simple and efficient prototype-based models for the clustering and topographic mapping of general dissimilarity matrices with wide applicability to domains involving non-Euclidean data such as text processing, biological sequence analysis, image processing, etc.

One major problem of clustering algorithms for dissimilarity data lies in the fact that the dissimilarity matrix scales quadratically with the number of data points. This leads to at least quadratic time and space complexity which is infeasible already for medium sized problems. Therefore, an approximation scheme was introduced which processes data based on patches in linear time and constant space, provided the patch size is chosen fixed depending on the available resources. We demonstrated that approximate patch clustering leads to an only minor decrease of the accuracy and that it is robust with respect to the data ordering. In particular, it can deal with settings where data are not identically distributed, and it leads to competitive results as if all data were available prior to training, as demonstrated in

experiments. This way, a very simple and fast, but powerful and flexible linear time and constant space clustering scheme for large dissimilarity data results. This patch scheme was proposed based on relational neural gas or relational self-organizing maps, however, alternative prototype-based schemes such as affinity propagation or median clustering could be easily extended accordingly. We demonstrated the applicability of the approach for a large text corpus. The evaluation and application in further large scale scenarios is subject of ongoing research.

References

[1] N.Alex, A.Hasenfuss, and B.Hammer (2009), Patch clustering for massive data sets. *Neurocomputing* 72(7-9):1455-1469.

[2] A.Asuncion and D.J.Newman (2007). *UCI Machine Learning Repository* [http://www.ics.uci.edu/∼mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

[3] M.Badoiu, S.Har-Peled, and P.Indyk (2002), Approximate clustering via core-sets, In: *Proceedings of the 34th ASM Symposium on Theory of Computing (STOC)*, pp.250-257.

[4] G.De A.Barreto, A.F.R.Araujo, S.C.Kremer (2003). A Taxonomy for Spatiotemporal Connectionist Networks Revisited: The Unsupervised Case. *Neural Computation* 15(6): 1255-1320.

[5] S. Belongie, C.Fowlkes, F.Chung, and J.Malik (2002). Spectral partitioning with indefinite kernels using the Nyström extension. In: A. Heyden (ed.), *ECCV 2002*, LNCS 2352, pp.531-542.

[6] J.C.Bezdek, R.J.Hathaway, J.M.Huband, C.Leckie, R.Kotagiri (2006), Approximate data mining in very large relational data, G.Dobbie, J.Bailey (Eds.): *Database Technologies 2006, Proceedings of the 17th Australasian Database Conference*, pp.3-13.

[7] M.Biehl, B.Hammer, S.Hochreiter, S.C.Kremer, and T. Villmann (2009), *Similarity-based learning on structures*, Dagstuhl Seminar Proceedings 09081, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

[8] P.S.Bradley, U.Fayyad, C.Reina (1998), Scaling clustering algorithms to large data sets, in: *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pp.9-15, AAAI Press.

[9] P. Brucker (1978). On the complexity of clustering problems. In: M. Beckman, H.P. Kunzi (eds.), *Optimization and Operations Research: Lecture Notes in Economics and Mathematical Systems*, pp.45-54, Springer.

[10] Y.Chen, E.K.Garcia, M.R.Gupta, A.Rahimi, and L.Cazzani (2009). Similarity-based classification: concepts and algorithms, *Journal of Machine Learning Research* 10: 747-776.

[11] R.Cilibrasi and M.B.Vitanyi (2005), Clustering by compression, *IEEE Transactions on Information Theory* 51(4):1523-1545.

[12] B. Conan-Guez, F. Rossi, and A. El Golli (2005), A fast algorithm for the self-organizing map on dissimilarity data, in *Workshop on Self-Organizing Maps*, 561-568.

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

[14] P. Domingos, G. Hulten (2001), A General Method for Scaling Up Machine Learning Algorithms and its Application to Clustering, In: *Proc.ICML*, 106-113.

[15] F.Farnstrom, J.Lewis, C.Elkan (2000), Scalability for clustering algorithms revisited, *SIGKDD Explorations* 2(1):51-57.

[16] B.J. Frey and D. Dueck (2007). Clustering by passing messages between data points. *Science* 315: 972-976, 2007.

[17] L.Goldfarb (1984), A unified approach to pattern recognition. *Pattern Recognition* 17(5):575-582.

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

[19] S. Guha, R. Rastogi, K. Shim (1998), CURE: an efficient clustering algorithm for large datasets. In: *Proceedings of ACM SIGMOD International Conference on Management of Data*, 73-84.

[20] B. Hammer, A. Hasenfuss, F.-M. Schleif, and T. Villmann (2006), Supervised median neural gas, In Dagli, C., Buczak, A., Enke, D., Embrechts, A., and Ersoy, O. (Eds.), *Intelligent Engineering Systems Through Artificial Neural Networks* 16, Smart Engineering System Design, pp.623-633, ASME Press.

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

[22] A. Hasenfuss, B. Hammer, F.-M. Schleif, and T. Villmann (2007), Neural gas clustering for dissimilarity data with continuous prototypes, in: F. Sandoval, A. Prieto, J. Cabestany, and M. Grana (eds.), *Computational*

and Ambient Intelligence – Proceedings of the 9th Work-conference on Artificial Neural Networks (IWANN), San Sebastian (Spain), pp. 539-546, LNCS 4507, Springer.

[23] B. Hammer, A. Hasenfuss, and T. Villmann (2007), Magnification control for batch neural gas, *Neurocomputing* 70:1225-1234.

[24] B. Hammer, A. Micheli, A. Sperduti, and M. Strickert (2004), Recursive self-organizing network models, *Neural Networks* 17(8-9):1061-1086.

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

[26] J.A. Hartigan (1975), *Clustering algorithms*. Wiley.

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

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

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

[30] T. Hofmann and J. Buhmann (1997). Pairwise data clustering by deterministic annealing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19(1):1-14.

[31] T. Hofmann and J.M. Buhmann (1996), An annealed 'neural gas' network for robust vector quantization, in: *Artificial Neural Networks ICANN'96*, pp. 151-156, Springer.

[32] T. Hofmann and J.M. Buhmann (1998), Competitive learning algorithms for robust vector quantization, *IEEE Transactions on Signal Processing*, 46(6):1665-1675.

[33] 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.

[34] I. Kaplansky (1977). *Set Theory and Metric Spaces*. AMS Chelsea Publishing.

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

[36] 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.

[37] T. Kohonen (1995), *Self-Organizing Maps*, Springer.

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

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

[40] A.Kumar, Y.Sabharwal, and S.Sen (2004), A simple linear time (1+epsilon)- approximation algorithm for k-means clustering in any dimensions. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp.454-462.

[41] J. Laub, V. Roth, J.M. Buhmann, K.-R. Müller (2006). On the information and representation of non-Euclidean pairwise data. *Pattern Recognition* 39:1815-1826.

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

[43] S.P. Luttrell (1989), Self-organisation: a derivation from first principles of a class of learning algorithms, In: *Proceedings of 3rd International Joint Conference on Neural Networks*, vol. 2, pp. 495-498.

[44] U. von Luxburg (2007). A tutorial on spectral clustering. *Statistics and Computing* 17(4):395-416.

[45] T. Martinetz (1992). Selbstorganisierende neuronale Netzwerkmodelle zur Bewegungssteuerung. Sankt Augustin. PhD thesis.

[46] 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.

[47] T. Martinetz and K. Schulten (1994), Topology representing networks. *Neural Networks* 7:507-522.

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

[49] F. Murtagh (2002), Clustering in massive data sets, in: *Handbook of Massive Data Sets*, Kluwer.

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

[51] J.Ontrup and H.Ritter (2001), Hyperbolic slef-organizing maos for semantic navigation, in T.Dietterich, S.Becker, and Z.Ghahramani (eds.), *Advances in Neural Information Processing Systems* 14, pp.1417-1424, MIT Press.

[52] P.M. Pardalos and S.A. Vavasis (1991). Quadratic programming with one negative eigenvalue is NP hard. *Journal of Global Optimization* 1:15-22.

[53] E. Pekalska and R.P.W. Duin (2005). *The Dissimilarity Representation for Pattern Recognition – Foundations and Applications*. World scientific.

[54] M.F.Porter (1980). An algorithm for suffix stripping. *program* 14(3):130-137.

[55] V. Roth, J. Laub, M. Kawanabe, J.M. Buhmann (2003). Optimal cluster preserving embedding of nonmetric proximity data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(12):1540-1551.

[56] J. Puzicha, T. Hofmann, J. Buhmann (1999). A theory of proximity based clustering: structure detection by optimziation. *Pattern Recognition Letters* 33(4):617-634.

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

[58] K.Rose (1998), Deterministic annealing for clustering, compression, classification, regression, and related optimization problems, *Proceedings of the IEEE*: 2210-223.

[59] F.Rossi, A.Hasenfuss, and B.Hammer (2007), Accelerating relational clustering algorithms with sparse prototype representations, *Proceedings of the 6th International Workshop on Self-Organizing Maps (WSOM'07)*.

[60] A.Saalbach, T.Twellmann, T.W.Nattkemper, A.Wismüller, J.Ontrup, H.Ritter (2005). A Hyperbolic Topographic Mapping for Proximity Data. *Artificial Intelligence and Applications 2005*: 106-111.

[61] S. Sahni (1974). Computationally related problems. *SIAM Journal on Computing* 3(4):262-279.

[62] B. Schölkopf (2000), *The kernel trick for distances*, Microsoft TR 2000-51.

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

[64] 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.

[65] T. Villmann, B. Hammer, F. Schleif, T. Geweniger, and W. Herrmann (2006), Fuzzy classification by fuzzy labeled neural gas, *Neural Networks*, 19:772-779.

[66] W. Wang, J. Yang, R.R. Muntz (1997). STING: a statistical information grid approach to spatial data mining. In *Proceedings of the 23rd VLDB Conference*, pp. 186-195.

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

[68] H.Yin (2008), Self-organising maps: Background, theories, extensions and applications, *Computational Intelligence: A Compendium*, Springer, 715-762.

[69] E. Yom-Tov, N. Slonim (2009), Parallel Pairwise Clustering, *Proceedings of the SIAM International Conference on Data Mining, SDM 2009*, pp. 745-755.

[70] T. Zhang, R. Ramakrishnan, M. Livny (1996). BIRCH: an efficient data clustering method for very large databases. In *Proceedings of the 15th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Databas Systems*, 103-114.

[71] S. Zhong, J. Ghosh (2003). A unified framework for model-based clustering. *Journal of Machine Learning Research* 4:1001–1037.

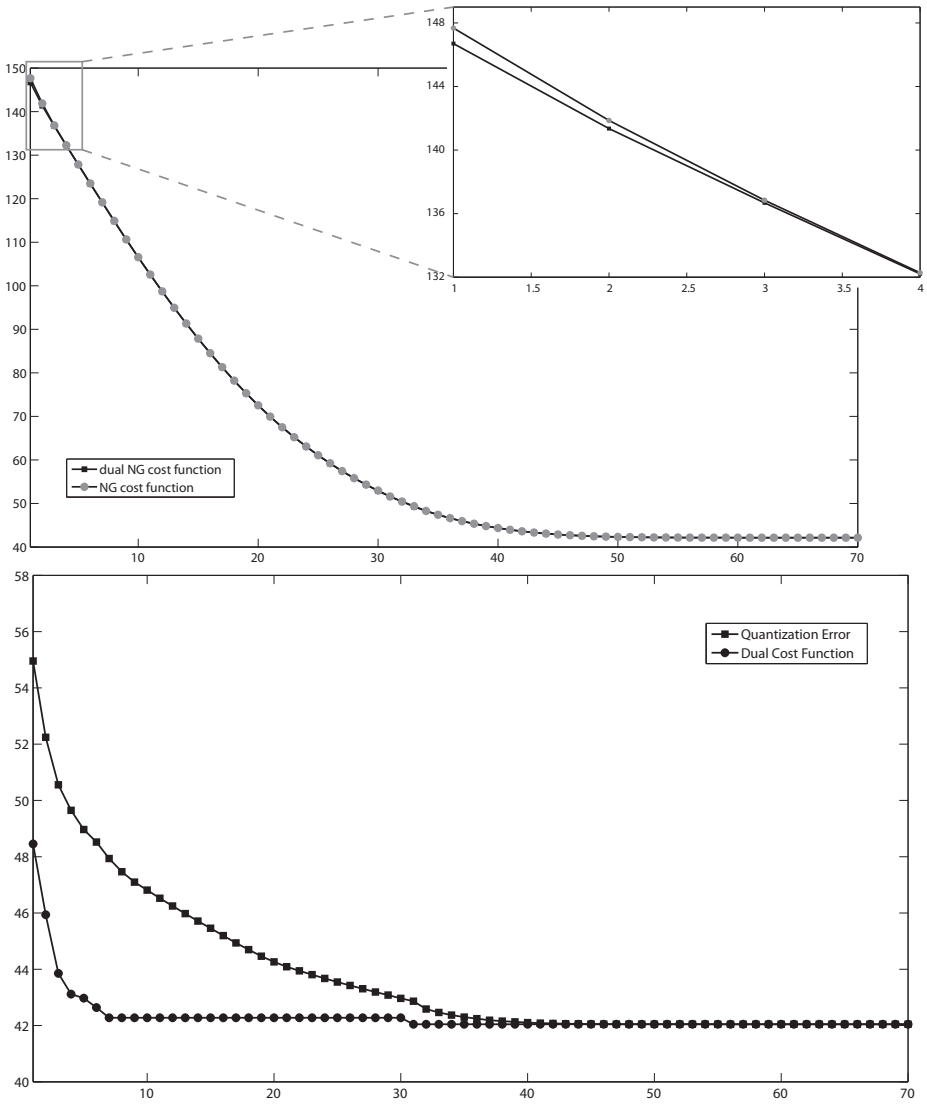

Figure 4: Cost function of NG and its dual (top) and standard quantization error and its dual (bottom) for the parameters as determined by relational NG

	dual cost function		quantization error		classification accuracy	
	training	test	training	test	training	test
cat cortex						
RNG	11.494 (0.359)	15.473 (0.569)	11.494 (0.359)	24.921 (0.327)	0.928 (0.028)	0.698 (0.076)
SRNG	11.965 (0.563)	15.163 (0.390)	12.027 (0.616)	24.292 (0.479)	0.994 (0.008)	0.724 (0.062)
DA	11.129 (2.587)	14.931 (2.074)	11.129 (2.587)	25.850 (0.643)	0.890 (0.105)	0.803 (0.083)
chromosomes						
RNG	22479.360 (54.428)	22864.645 (45.324)	22479.360 (54.428)	24275.995 (39.635)	0.893 (0.004)	0.897 (0.004)
SRNG	22470.299 (65.974)	22848.012 (35.060)	22475.191 (65.791)	24241.312 (30.111)	0.912 (0.003)	0.907 (0.004)
DA	22608.490 (43.417)	23090.597 (51.963)	22608.490 (43.417)	24374.898 (38.229)	0.910 (0.004)	0.908 (0.003)
protein data						
RNG	293.658 (0.730)	17.173 (0.540)	293.658 (0.730)	40.450 (0.938)	0.942 (0.002)	0.919 (0.016)
SRNG	291.283 (1.136)	17.109 (0.575)	291.996 (1.226)	40.081 (1.012)	0.980 (0.005)	0.944 (0.013)
DA	282.440 (0.603)	15.350 (0.820)	282.440 (0.603)	39.312 (0.835)	0.931 (0.003)	0.907 (0.008)
aural sonar						
RNG	4.174 (0.076)	5.048 (0.101)	4.174 (0.076)	6.908 (0.079)	0.892 (0.017)	0.834 (0.014)
SRNG	4.334 (0.095)	5.121 (0.115)	4.360 (0.109)	6.927 (0.142)	0.993 (0.006)	0.870 (0.032)
DA	3.993 (0.073)	4.995 (0.125)	3.993 (0.073)	6.993 (0.136)	0.897 (0.013)	0.856 (0.026)
caltech						
RNG	2435.724 (1.587)	2467.042 (2.522)	2435.724 (1.587)	2647.701 (2.349)	0.451 (0.002)	0.407 (0.006)
SRNG	2466.918 (1.854)	2508.075 (3.782)	2489.320 (2.100)	2640.619 (2.042)	0.791 (0.009)	0.364 (0.025)
DA	2440.096 (1.880)	2471.818 (2.351)	2440.096 (1.880)	2659.730 (1.812)	0.452 (0.003)	0.401 (0.004)
face recognition						
RNG	0.071 (0.003)	0.005 (0.001)	0.071 (0.003)	0.013 (0.001)	0.887 (0.003)	0.865 (0.002)
SRNG	0.250 (0.013)	0.021 (0.002)	0.472 (0.020)	0.058 (0.002)	0.830 (0.005)	0.811 (0.007)
DA	0.116 (0.007)	0.008 (0.001)	0.116 (0.007)	0.017 (0.001)	0.900 (0.002)	0.873 (0.004)
patrol						
RNG	120.779 (0.194)	7.885 (0.297)	120.779 (0.194)	17.343 (0.256)	0.835 (0.005)	0.665 (0.024)
SRNG	123.766 (0.193)	7.935 (0.434)	124.266 (0.226)	16.995 (0.144)	0.989 (0.001)	0.657 (0.022)
DA	127.340 (5.399)	10.393 (0.936)	127.340 (5.399)	17.134 (0.199)	0.713 (0.077)	0.521 (0.051)
voting						
RNG	8.861 (0.047)	0.615 (0.024)	8.861 (0.047)	1.186 (0.028)	0.953 (0.001)	0.950 (0.004)
SRNG	8.933 (0.038)	0.651 (0.033)	9.084 (0.027)	1.188 (0.030)	0.972 (0.001)	0.953 (0.004)
DA	8.849 (0.069)	0.626 (0.021)	8.849 (0.069)	1.156 (0.034)	0.955 (0.001)	0.951 (0.005)

Table 3: Averaged results of a repeated cross-validation on the data sets, the standard deviation is given in parenthesis.

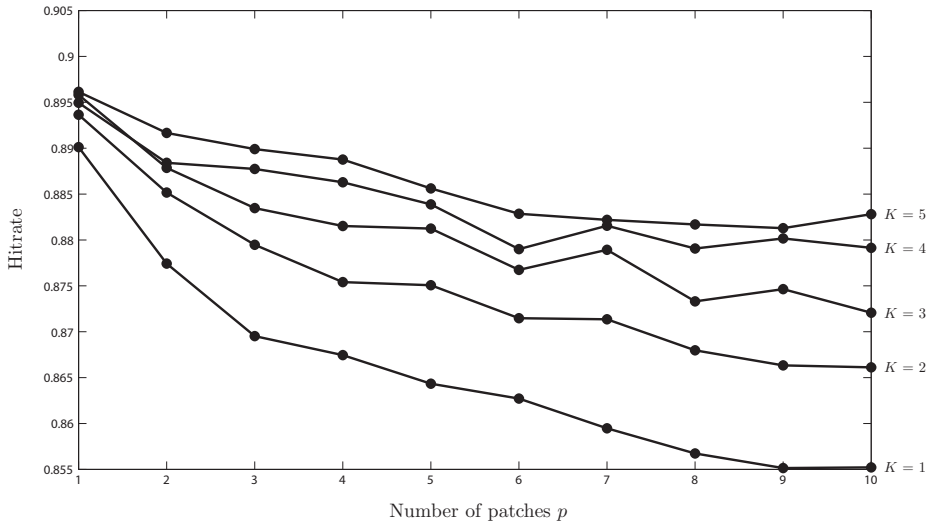

Figure 5: Accuracy of patch training depending on the size of the patches and the accuracy of the approximation

	dual cost function		**quantization error**		**classification accuracy**	
	training	**test**	**training**	**test**	**training**	**test**
random order						
patch RNG	23589.613 (68.993)	23898.934 (85.568)	30157.041 (83.365)	31623.014 (133.924)	0.873 (0.006)	0.868 (0.007)
RNG	22307.283 (51.475)	22777.673 (85.027)	22309.766 (51.048)	24255.182 (95.339)	0.898 (0.005)	0.900 (0.005)
ordered according to the labeling						
patch RNG	23495.345 (80.502)	23831.059 (89.282)	30016.631 (131.794)	31534.463 (124.280)	0.875 (0.007)	0.870 (0.009)
RNG	22326.892 (39.646)	22757.129 (35.638)	22329.489 (39.927)	24250.462 (31.639)	0.900 (0.007)	0.901 (0.007)

Table 4: Results of patch clustering when data sets are represented in different order, i.i.d. resp. non i.i.d. with respect to the labeling, in comparison to full batch RNG.

name	count	percentage
alt.atheism	72068	39.26
comp.os.ms-windows.misc	178	0.10
comp.windows.x	182	0.10
rec.motorcycles	15841	8.63
rec.sport.baseball	462	0.25
rec.sport.hockey	510	0.28
sci.crypt	3203	1.75
sci.med	2912	1.59
soc.religion.christian	341	0.19
talk.politics.guns	23209	12.64
talk.politics.mideast	9337	5.09
talk.politics.misc	54683	29.79
talk.religion.misc	620	0.34

Table 5: Newsgroup present in the big newsgroup data set

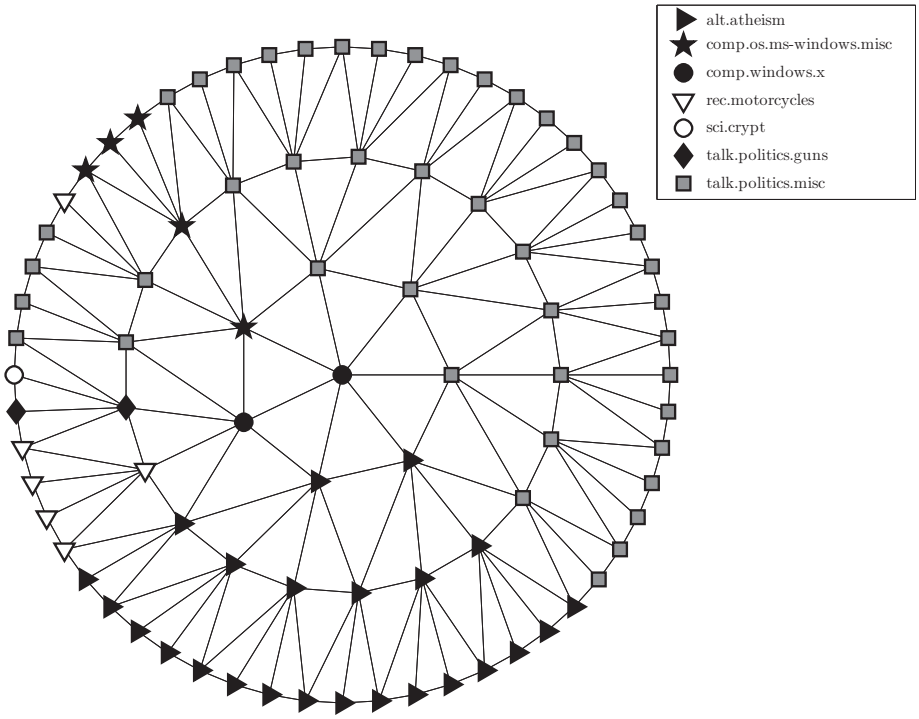

Figure 6: Visualization of 183,546 newsgroup articles using patch relational HSOM. Labeling of the neurons is based on majority vote, displaying the topographic arrangement of the biggest newsgroups on the map. In this case, patch processing reduces the required space of the full dissimilarity matrix from approx. 251 GB to only around 13 MB.