PRACE NAUKOWE UNIWERSYTETU EKONOMICZNEGO WE WROCŁAWIU
Nr 7 (1207) 2008
TAKSONOMIA 15
Klasyfikacja i analiza danych – teoria i zastosowania

Reinhold Decker
Bielefeld University, Germany

GROWING NEURAL GAS-LIKE NETWORKS AND THEIR APPLICATION TO DATA ANALYSIS AND CLUSTERING IN MARKETING

1. Introduction

Today, the Self-Organizing feature Map (SOM) suggested by T. Kohonen is a standard approach for pattern recognition and clustering [Kohonen 2001]. Although the neural gas network approach by Martinetz and Schulten [1991] also gains increasing popularity, most of its applications are reported in technical fields like image analysis, so far. Applications in economics and particular in marketing are still rare.

The methodological basis of the Neural Gas Network (NGN) algorithm is vector quantization. By means of this unsupervised process a vector space can be partitioned such that all data vectors are represented by a predefined number of prototypes, the nodes of the respective network. An important difference between SOM and NGN results from the structure of the network. A SOM represents the input data by means of a predefined grid, whereas NGN consists of nodes that can freely "move" in the data space, like gas molecules in the air. Accordingly, SOMs are able to obtain good vector quantization results if the network topology matches the topology of the data manifold. However, the latter is often unknown in advance, for instance in empirical market segmentation. So, NGN can outperform SOM when quantizing topologically arbitrarily structured data manifolds [Estévez and Figueroa 2006].

According to Questier et al. [2002] NGNs have equal visualization power as the Kohonen SOMs and provide similar clustering results as *k*-means. Cottrell et al. [2006] even call it a simple and highly effective algorithm for data clustering. On the other hand, NGN requires the specification of several control parameters, which can degenerate to a time-consuming task, particularly if high-dimensional

data has to be analyzed. Above all, the a priori fixing of the number of nodes, or clusters, has initiated further developments. The most popular one may be the Growing Neural Gas Network (GNGN) suggested by Fritzke in 1995. Further developments of this concept will be presented in the following.

The basic idea of recent growing neural gas-like networks is to create a new node if the input data suggests an enlargement of the neural network in favor of the data representation as a whole. In return, nodes with low "utility" with respect to the goodness of data representation are removed during the network adaptation process. The available algorithms differ, among other things, in the way of adding new nodes. The GNGN algorithm, for instance, repeatedly inserts a new node after a pre-specified number of adaptation cycles. Others, e.g., the so-called Grow When Required Network by Marsland et al. [2002, 2005] and the Self-controlled Growing Neural Network (SGNN) by Decker [2005], add a new node if the fit of the node, which best matches the current input data, is lower than a dynamically adapted threshold. We refer to this by the term "activity" in the following. Since the number of parameters that have to be preset by the user in advance is less with SGNN, it features comparatively high autonomy regarding the dynamic controlling of the adaptation process.

2. The SGNN approach and its modifications

Before sketching the SGNN algorithm, some notation has to be introduced. The input data is represented by K-dimensional vectors $\mathbf{x}_j = (x_{j1}, \ldots, x_{jk}, \ldots, x_{jK})$, where index j ($j = 1, \ldots, J$) refers to an individual data point or input signal. The neural network comprises H nodes or prototypes which are represented by weight vectors $\boldsymbol{\eta}_h = (\eta_{h1}, \ldots, \eta_{hk}, \ldots, \eta_{hK})$. By means of vector quantization the $J \times K$ input space can be partitioned into an $H \times K$ output space. The set of nodes is denoted by B and the set of connections in the neural network by C.

The SGNN approach starts from a neural network, which consists of only two nodes, u_1 and u_2. Both of them are operationalized by K-dimensional weight vectors $\boldsymbol{\eta}_1$ and $\boldsymbol{\eta}_2$. In the beginning, the adaptation cycle counter l is set to zero and the set of connections between different nodes C is empty. For controlling the network growth and adaptation process, two internal variables are needed, namely a firing counter y_h (initialized with 0) and a variable w_h (initialized with 1), which refers to the training requirement a node features during the adaptation process. Before starting the process, the user has to specify parameters ε_{Best} and ε_{Second}. Both are needed to control the growth and weight adapting process externally. The same applies to the age variable a_{Max}. The maximum number of adaptation cycles L should be selected significantly greater than the number of input vectors J. Starting from this initialization, a network adaptation cycle looks like this:

1. Generate input vector $\mathbf{x}_j$ and compute Euclidean distances

$$dist(\mathbf{x}_j, \boldsymbol{\eta}_h) = \sqrt{\sum_{k=1}^{K} (x_{jk} - \eta_{hk})^2} \quad \forall u_h \in B.$$

2. Determine the best ($u_{h_{Best}}$) and the second best matching node ($u_{h_{Second}}$), such that

$$h_{Best} = \arg\min_{h \in \{1,\ldots,H\}} dist(\mathbf{x}_j, \boldsymbol{\eta}_h) \quad \text{and} \quad h_{Second} = \arg\min_{h \in \{1,\ldots,H\} \setminus h_{Best}} dist(\mathbf{x}_j, \boldsymbol{\eta}_h).$$

3. If the nodes $u_{h_{Best}}$ and $u_{h_{Second}}$ are not connected, then

$$C = C \cup \{(h_{Best}, h_{Second})\} \text{ and } a_{h_{Best}h_{Second}} = 0, \text{ else } a_{h_{Best}h_{Second}} = 0.$$

4. Compute the activity of the best matching node according to:

$$v_{h_{Best}} = \exp(-dist(\mathbf{x}_j, \boldsymbol{\eta}_{h_{Best}})).$$

5. Compute thresholds for both the activity (i.e. v_{Thres}) and the training requirement (i.e. w_{Thres}) of $u_{h_{Best}}$:

$$\text{If } v_{h_{Best}} < v_{Thres} \text{ then } v_{Thres} = \frac{l v_{Thres} + v_{h_{Best}}}{l+1}, \quad w_{Thres} = (1 - \varepsilon_{Best})^{\sqrt{|B|}} \cdot \frac{\ln(l+1)}{|B|}.$$

6. If $v_{h_{Best}} \geq v_{Thres}$ or $w_{h_{Best}} \geq w_{Thres}$ then go to step 7, else create a new node $u_{h_{New}}$ with $\boldsymbol{\eta}_{h_{New}} = \frac{1}{2}(\boldsymbol{\eta}_{h_{Best}} + \mathbf{x}_j)$, update the topological structure of the network regarding B and C, set $y_{h_{New}} = 0$, $w_{h_{New}} = 1$, and go to step 8.

7. Adapt the best matching node and all the nodes to which it is connected:

$$\boldsymbol{\eta}_{h_{Best}} = \boldsymbol{\eta}_{h_{Best}} + \Delta\boldsymbol{\eta}_{h_{Best}} \text{ with } \Delta\boldsymbol{\eta}_{h_{Best}} = \varepsilon_{Best} w_{h_{Best}} (\mathbf{x}_j - \boldsymbol{\eta}_{h_{Best}}),$$

$$\boldsymbol{\eta}_{h_i} = \boldsymbol{\eta}_{h_i} + \Delta\boldsymbol{\eta}_{h_i} \text{ with } \Delta\boldsymbol{\eta}_{h_i} = \varepsilon_{Second} w_{h_i} (\mathbf{x}_j - \boldsymbol{\eta}_{h_i}) \quad \forall h_i \text{ with } (h_{Best}, h_i) \in C.$$

8. Update the control variables:

$$y_{h_{Best}} = y_{h_{Best}} + 1, \quad w_{h_{Best}} = \frac{1}{y_{h_{Best}} + 1}, \quad a_{h_{Best}h_i} = a_{h_{Best}h_i} + 1 \ \forall h_i \text{ with } (h_{Best}, h_i) \in C.$$

9. Delete all connections $(h_i, h_{i'}) \in C$ with $a_{h_i h_{i'}} > a_{Max}$ and all nodes $u_{h_i} \in B$ which are not connected to any other node $u_{h_{i'}}$ $(i \neq i')$ and satisfy $y_{h_i} < |B|$.

10. Increase the adaptation cycle counter l by 1,
11. If $l \leq L$ then go to step 1, else stop and create connectivity matrix

$$\mathbf{C} = (c_{h_1 h_2})_{h_1, h_2 = 1, \ldots, H = |B|} \text{ with } c_{h_1 h_2} = \begin{cases} 1, \text{ if nodes } u_{h_1} \text{ and } u_{h_2} \text{ are connected} \\ 0, \text{ otherwise} \end{cases}.$$

Hence, matrix $\mathbf{C}$ describes the final topological structure of the neural network. Each node – or more precisely – the associated weight vectors can be interpreted as prototypes, which represent a certain pattern in the data set considered. Similar prototypes are connected either directly or by a small number of edges in the implicit connectivity graph. In the neural network terminology the clusters are represented by Voronoi regions and the weight vectors equal the centroids.

The flexible structure of the SGNN algorithm enables modifications serving different objectives. The first one, called Growing Neural Network with Autonomous Parameter Specification (GNNAPS), was driven by the intention of simplifying the parameterization and increasing the average adaptation speed. The only external parameter to be set by the user – besides the maximum number of adaptation cycles L – is the compression level ϕ. Choosing a ϕ value close to 1 implies an intensive compression of the data, whereas the opposite happens if ϕ approaches 0. Moreover, in order to internally compute appropriate values for the learning parameters ε_{Best} and ε_{Second}, a variable S_{Max} is introduced, which refers to the "diameter" of the data base to be analyzed. We define

$$S_{Max} = \max\{d_{j_1}, \ldots, d_{j_n}\}, \text{ with } n \leq J \text{ and}$$

$$d_{j_i} = \sqrt{\sum_{k=1}^{K} (x_{j_i k} - x_{j_{i'} k})^2} \quad \forall j_i, j_{i'} \in \{1, \ldots, J\}, j_i \neq j_{i'}.$$

We use the distance between the first input vector and the origin as a default. The relation between ε_{Best} and ε_{Second} on the one hand and ϕ, S_{Max} as well as L on the other hand can be modeled using a logistic function. The maximum age of a connection is defined as $a_{Max} = |B| - 1$. Further details are given in Decker [2006].

The basic motivation underlying the second modification, called Growing Neural Network with integrated testing for Cluster-wise Normality (GNNCN), was to find Gaussian clusters. In the current version of GNNCN, we apply the well-known Mardia test to verify the Gaussian assumption [Mardia 1970; Schwager 1985]. The suggested principle bears resemblance to the G-means approach by Hamerly and Elkan [2003].

A Gaussian structure of the clusters is useful in market segmentation and consumer typology, for instance, since it eases the interpretation of the cluster centers as consumer profiles. Insofar, the extent to which the objects belonging to the indi-

vidual clusters are normally distributed can be used as an indicator for the overall quality of the obtained segmentation.

In GNNCN we use the Gaussian distribution of the Voronoi regions as a new stopping criterion for the growth and adaptation process. Therefore, we modified the Mardia formulas of the kurtosis d_h^{kurt} and the skewness d_h^{skew} by integrating the weight vectors of the current adaptation cycle:

$$d_h^{kurt} = \frac{1}{|N_h|} \sum_{n \in N_h} ((\mathbf{x}_n - \boldsymbol{\eta}_h)\mathbf{S}_h^{-1}(\mathbf{x}_n - \boldsymbol{\eta}_h)')^2 \text{ and}$$

$$d_h^{skew} = \frac{1}{|N_h|^2} \sum_{n \in N_h} \sum_{n' \in N_h} ((\mathbf{x}_n - \boldsymbol{\eta}_h)\mathbf{S}_h^{-1}(\mathbf{x}_{n'} - \boldsymbol{\eta}_h)')^3 ,$$

where $\mathbf{S}_h$ denotes the cluster-specific sample covariance matrix and N_h is the set of input vectors assigned to node u_h. The corresponding p-values are:

$$p_h^{kurt} = 2(1 - P_N(|\, b_h^{kurt}\,|)) \text{ with } b_h^{kurt} = \frac{d_h^{kurt} - K(K+2)}{\sqrt{8K(K+2)/|\,N_h\,|}} \quad \forall h$$

$$p_h^{skew} = 1 - P_{\chi^2(K(K+1)(K+2)/6)}(b_h^{skew}) \text{ with } b_h^{skew} = |\,N_h\,|\, d_h^{skew}/6 \quad \forall h.$$

The test statistic for the sample kurtosis is asymptotically standard normally distributed, whereas the one for the sample skewness is asymptotically chi-square distributed.

The hypothesis of multivariate normality can be rejected if d_h^{skew} is very large or if d_h^{kurt} is either very large or very small. Moreover, the term $1 - \varepsilon_{Best}$ is replaced by the significance level α of the Mardia test. Further details are given in Decker et al. [2006].

The different approaches are compared in the following by using the well-known quantization error [Kohonen 2001]

$$PM_1 = \sum_{j=1}^{J} dist(\mathbf{x}_j, \boldsymbol{\eta}_{m_j}) \text{ with } m_j = \arg\min_h dist(\mathbf{x}_j, \boldsymbol{\eta}_h)$$

and the compactness [Marsland et al. 2002]

$$PM_2 = \sum_{h_1=1}^{H} \sum_{h_2=h_1+1}^{H} c_{h_1 h_2} dist(\boldsymbol{\eta}_{h_1}, \boldsymbol{\eta}_{h_2}).$$

The first measure shows how the neural network minimizes the distances between the centroids and the cluster elements, whereas the second one penalizes the neural network for neighboring connections between nodes that are placed far apart

on the connectivity graph. In general, minimizing both measures leads to an acceptable mapping of the data.

3. Some evidence of network performance

We make use of two synthetic 2D data sets to show strengths and weaknesses of the relevant algorithms in pattern recognition and clustering. The first data set contains more than 19 000 data points which define seven different graphical objects (see the gray background images in Figure 1). Especially the two single lines are worth to be considered closely due to their significantly differing numbers of data points. The horizontal line represents 7260 data points, whereas the vertical one consists of only 100 points. But the crucial thing is the varying density of the horizontal line. The more one goes to the right on this line, the more points are underlying the respective section of the line. Thus, we have high numbers of more or less identical data points here. In contrast to this, each of the five black rectangles in the chessboard pattern, as well as the dotted rectangle on the left hand side, have been generated with 1600 data points. The second data set comprises seven data clouds, each of them consisting of 1000 approximately bivariate normally distributed data points.

Applying the GNGN algorithm by Fritzke [1995] to the first data set provides the result given on the left hand side of Figure 1. The external parameters are in line with the author's suggestions. On the right hand side, the GNNAPS output for compression level $\phi = 0.15$ can be seen.

At a first glance, both representations look rather similar. However, the performance measures PM_1 and PM_2 indicate significant differences, particularly when considering the quantization error, which is remarkably lower for GNGN. The reason for this becomes obvious when examining the horizontal line. Since this object or region represents almost 40 percent of the data, it attracts a comparatively large number of nodes (see [Hammer et al. 2007] for a recent discussion of magnification in connection with the NGN approach). This, at least partly, happens at the expense of the representation of other objects, for instance, the wavy line at the top and the large oval in the lower right corner. In contrast to this, the results provided by GNNAPS are more balanced, particularly when taking into account that the size of the nodes indicates the relative quantity of data points they represent.

From Table 1, it can be seen how the performance measures of GNGN and GNNAPS successively improve, the more adaptation cycles are passed. But the interesting point concerns the node generation process. Already after $L = 100\ 000$ cycles, GNNAPS converges to the final number of nodes, which seems to be around 182. However, whereas GNNAPS determines the size of the network more or less autonomously, the growth process of GNGN has to be controlled by implicitly defining the frequency of adding new nodes in advance.

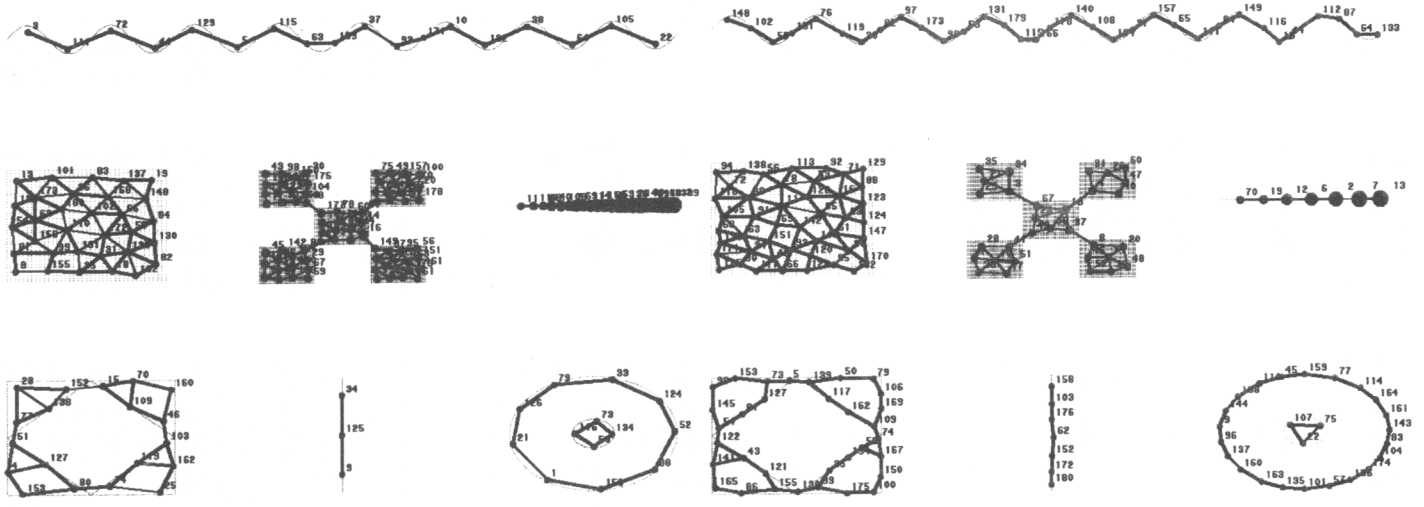


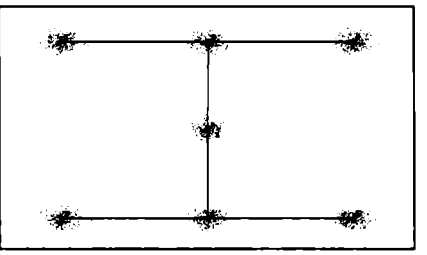
Fig. 1. Pattern representation with GNGN (left: $PM_1 = 12\ 422.5$, $PM_2 = 870.4$) and GNNAPS (right: $PM_1 = 17\ 343.5$, $PM_2 = 801.6$)

Table 1. Performance of GNGN (each first row) and GNNAPS (each second row) depending on the number of adaptation cycles

L	H	PM_1	PM_2
1 000	73	57 254.6	809.2
	71	29 606.0	970.2
10 000	174	17 607.2	1250.6
	172	19 386.0	1441.5
100 000	186	12 958.1	896.9
	187	18 394.5	1139.3
1 000 000	179	12 691.0	866.5
	178	18 155.8	822.5
5 000 000	180	12 427.3	846.4
	179	18 762.8	803.5
10 000 000	182	12 422.5	870.4
	181	17 343.5	801.6

Table 2. Pattern representation with GNNCN (for $\alpha = 0.01$)

h	p_h^{kurt}	p_h^{skew}
1	0.63	0.86
2	0.38	0.02
3	0.63	0.83
4	0.77	0.30
5	0.23	0.62
6	0.02	0.38
7	0.02	0.49



In order to demonstrate the performance of the GNNCN algorithm we make use of the second 2D data set. The p-values for both the kurtosis and the skewness regarding the seven clusters are depicted in Table 2, together with the resulting data representation. Obviously, the Gaussian assumption can not be rejected at the one percent level. The associated performance values are $PM_1 = 17\,375.7$ and $PM_2 = 146.8$. For GNGN (with $L = 10^6$), we get $PM_1 = 17\,582.0$ and $PM_2 = 149.4$, which points to a slightly poorer performance. By the way, applying k-means (with $h = 7$) to this data set provides very similar results.

In the recent past, the clustering on streaming data has been increasingly discussed in the relevant literature (see, e.g., [Orlowska et al. 2006]). Data streams can occur in retailing (in terms of point of sale scanner data), travel business (in terms of online booking profiles), e-commerce (in terms of customer click streams), and many other fields. The key issue in these applications is the necessity of getting along with one pass. In the present case this means that each new data point initiates exactly one adaptation cycle. Static clustering algorithms, such as k-means, are not suited for this kind of data, at least in their basic implementation.

Using the second 2D data set, it can be shown that the algorithms considered here are largely insensitive to the ordering of the input data. For demonstration purposes the GNNAPS algorithm is used in the following. The left diagram in Figure 2 shows the results we get when 100 randomly drawn data points are successively arriving and initiating one adaptation cycle each. By repeating this experiment 20 times, the illustrated performance profiles result. Less astonishing, the shortness of the data streams leads to varying values regarding PM_1 (bar profile) and PM_2 (line profile). However, if the data stream consists of 1000 input signals the results of the different runs are already converging, as to be seen from the right diagram. If the one-pass adaptation includes all 7000 data points the performance measures approach the above-mentioned GNNCN results.

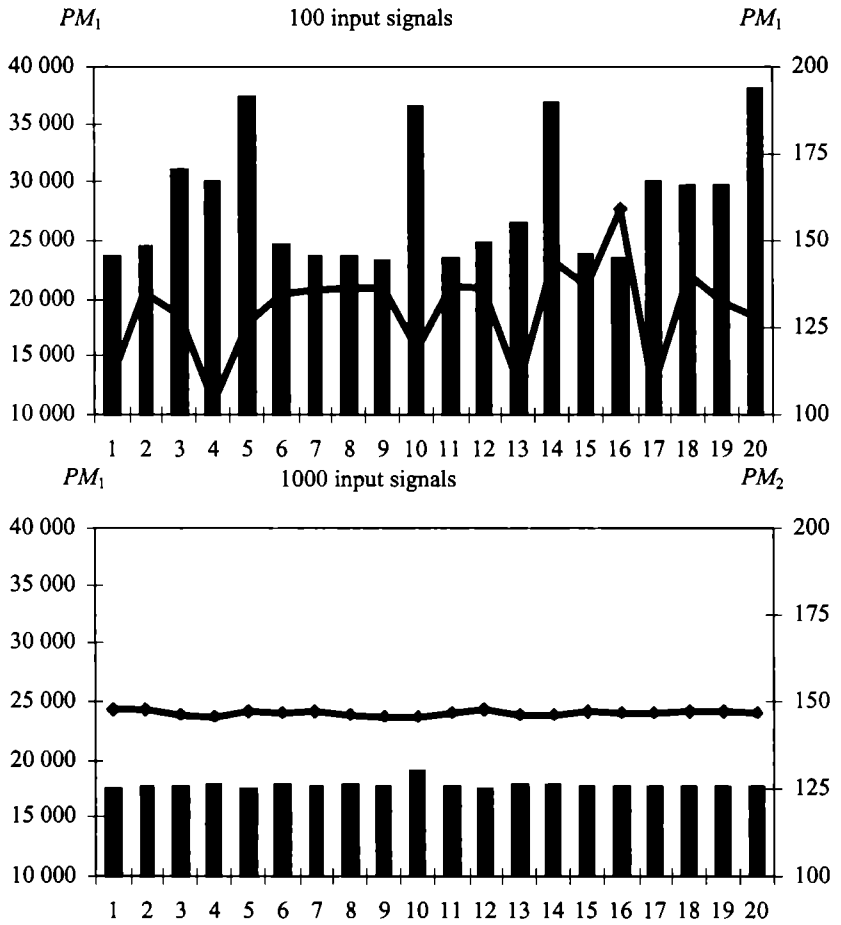


Fig. 2. Performance when clustering on streaming data with GNNAPS

4. Applications in empirical marketing research

4.1. Consumer or household typology

The first example stems from lifestyle analysis, a typical domain for applying clustering algorithms. Following Brassington and Pettitt [2005] lifestyle segmentation methods can open the door to a better-tailored, more subtle offering to the customer on all aspects of the marketing mix. In such applications the a priori fixing of the number of clusters or segments is extremely difficult, if not even impossible.

For demonstration purposes a data set is used which was provided by the German ZUMA Institute [Papastefanou et al. 2001], and which comprises 61 Likert-scaled statements (5-point scale) characterizing the individual lifestyles of 1000 German households as well as four demographic variables (different quasi-metric scales). A considerable number of statements are concerned with nutrition behavior. A typical statement reads as follows: "Multivitamin juices are an important supplement to daily nutrition". Rating this statement with 1 means "I strongly disagree", whereas 5 means "I strongly agree". The individual ratings are represented by the input vectors $\mathbf{x}_j$, where, e.g., $\mathbf{x}_{10} = (4,3,2, \ldots)$ indicates that household 10 responded "4" to statement 1, "3" to statement 2, and so on.

Applying the GNNCN algorithm to this data (with $\alpha = 0.01$) leads to an 11-cluster solution which satisfies with respect to the relevant performance measures and the interpretability of the resulting cluster centroids and weight vectors respectively. Figure 3 illustrates the weight profile of cluster 10 (i.e. $\boldsymbol{\eta}_{10} = (\eta_{10,1}, \ldots, \eta_{10,65})$) which obviously represents rather conservative consumers. The low value for statement 5, for example, indicates that the members of this cluster typically do not like to try new products. But seemingly, according to statement 10 and 27, the respective respondents like to stay at home and prefer hearty plain fare, whereas vegetarian food is strictly rejected according to statement 41. The average net income of these households (see item 63) is rather low compared to other segments.

4.2. Market basket analysis

A second field of application is market basket analysis based on point of sale scanner data. According to Russell and Petersen [2000], market basket analysis focuses on the decision process by which consumers select items from a given set of product categories on the same shopping trip. More technically speaking, it aims at uncovering interrelations between choices of different products purchased in a specific retail store such as a supermarket [Giudici and Passerone 2002]. Investigations of this kind can be useful for the category management and the micromarketing in retail stores [Mild and Reutterer 2003].

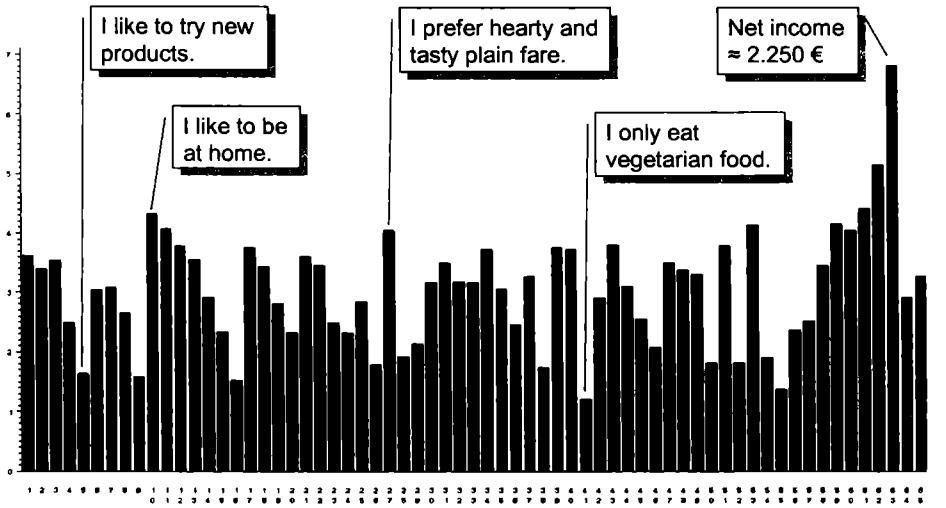


Fig. 3. Profile of lifestyle cluster 10

The data for this study has been provided by a large German supermarket chain. In the following, the basic SGNN approach is used to study the cross-category purchase behavior concerning 200 product categories by analyzing almost 89 700 market baskets. Each market basket is characterized by the occurrence or non-occurrence of the individual product categories. In the case of binary coded scanner data, an input vector $\mathbf{x}_{10} = (1,0,1, \ldots)$ would indicate that market basket 10 contains at least one item of product category 1, no item of category 2, and so on.

As to be expected, the neural network uncovers well-known and easy to explain cross-category dependences, for instance concerning the product categories "noodles" and "ready-to-serve sauces". The interdependency graph depicted in Figure 4 has to be read as follows: Two product categories are connected by an edge, if both have weights larger than a user-defined display level, say 0.25, for at least one node (prototype) in the underlying neural network. By varying the display level the user can determine the degree of differentiation in the graph. If the level is set to a low value the weak relations are displayed as well. The higher the display level, the less relations (edges) are shown. Further details about this representation technique are provided by Decker [2005].

But the SGNN approach is also able to uncover more complex and hardly foreseeable cross-category dependencies as illustrated by the oval graph in Figure 4. The cross-category purchase effects coming from the dairy assortment depicted on the right side may provide useful insights regarding possible starting points of promotional activities. The large number of edges emanating from product category 59 marks fruit yoghurt as a top selling product, which is predestinated for sales promotions aiming at the initiation of cross-category purchases. By this

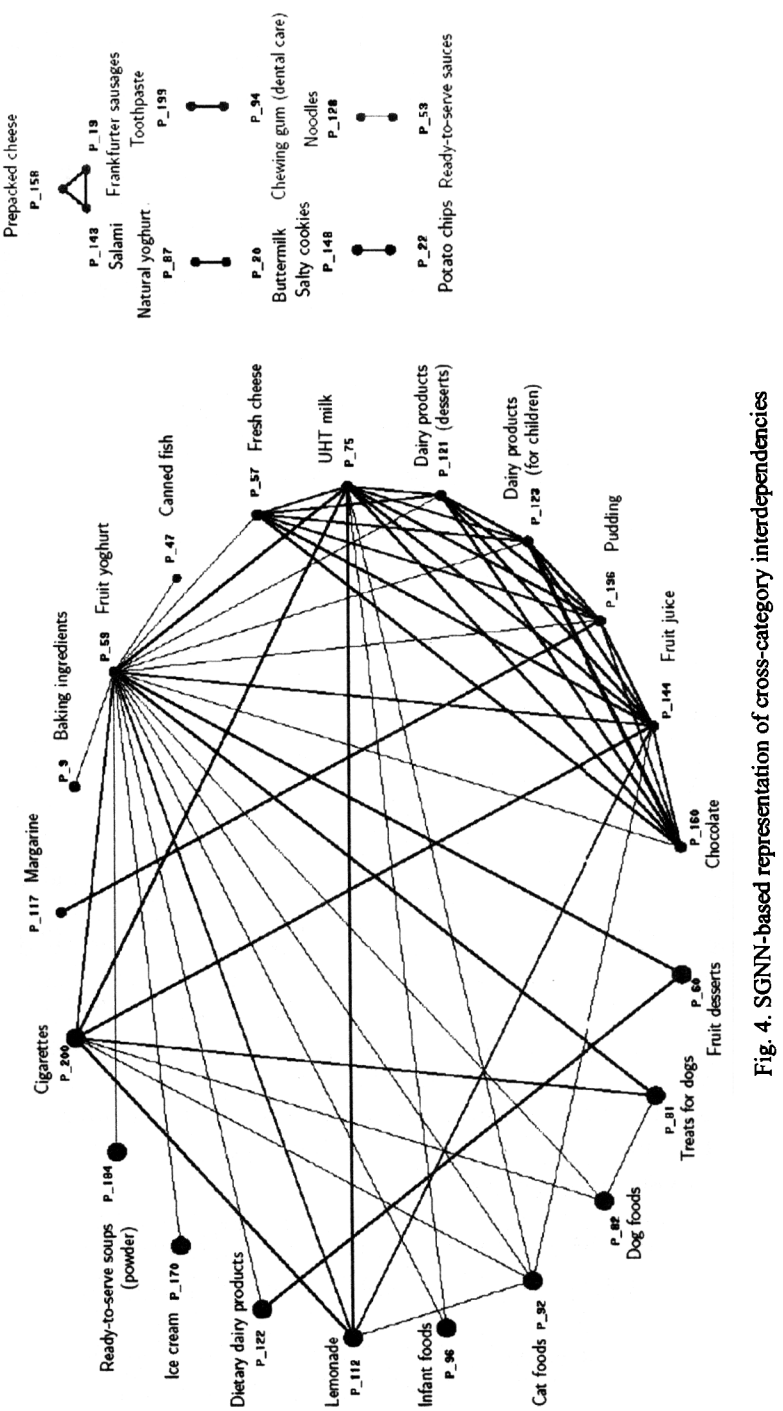


Fig. 4. SGNN-based representation of cross-category interdependencies

means, SGNN-based market basket analysis can also help to establish product listing decisions of the category management on a broader basis.

The available weights η_{hk} can be used to analyze existing asymmetries in purchasing behavior [Decker and Monien 2003]. Therefore, the probability of observing product category PC_k in a market basket that corresponds to pattern h can be defined as follows:

$$P(PC_k \mid u_h) = \frac{\eta_{hk}}{\sum_{h'=1}^{H} \eta_{h'k}} \quad \forall h,k.$$

If we further define the probability of realizing a market basket pattern h as $P(u_h) = |N_h|/J$, with J sufficiently large, then the a posteriori probability $P(u_h \mid PC_k)$ of realizing a market basket pattern h, provided product category PC_k is already included in the basket, can be estimated by applying the Bayes theorem:

$$P(u_h \mid PC_k) = \frac{P(PC_k \mid u_h) \cdot P(u_h)}{\sum_{u_i \in B} P(PC_k \mid u_i) \cdot P(u_i)} \quad \forall h,k.$$

4.3. Missing value imputation

Many databases resulting from surveys (e.g. in market research) or experiments (e.g. in the social or natural sciences) are characterized by the problem of incompleteness in terms of missing values. While several data analysis methods can deal with incomplete data matrices, a considerable number of methods still require complete data matrices. So, imputation is often the only feasible option.

In the last decade multiple imputation methods have been intensively discussed in the literature [Allison 2001; Rubin 2004] and are now available in popular statistical packages such as SAS® and S-Plus®. Multiple imputation is a well-established technique for analyzing data sets where some units have incomplete observations [Carpenter et al. 2006]. The method is valid provided the data are missing at random (MAR) or completely at random (MCAR).

One of the most popular multiple imputation techniques is the Markov Chain Monte Carlo (MCMC) method, particularly when the missingness is more complicated [Horton and Lipsitz 2001]. The MCMC method requires the Gaussian assumption [SAS 2003], but there is some evidence [Schafer 1997] that the inferences tend to be robust to minor departures from normality. A major advantage of MCMC is its ability to handle arbitrary patterns of missing data. Therefore, MCMC is used as a benchmark for a SGNN-based imputation approach.

Two different data sets are considered to provide a more comprehensive picture of how the suggested approach works. The first one comprises the average prices (in cents per pound) of $K = 5$ consumer goods in $J = 23$ cities in the United States [Sharma 1996]. For this 23×5 data set the Gaussian assumption cannot be rejected at the 10 percent level according to Mardia's test of multivariate normality [Mardia 1970]. The second one was generated from the well-known Iris data set by Fisher [1936] with $K = 4$ flower attributes and $J = 150$ observations. Both data sets are converted into incomplete data matrices by randomly deleting a certain number of values (according to the MCAR definition).

Before applying the SGNN approach to the problem at hand, a slight modification of the formula for computing the distances between the input vectors $\mathbf{x}_j$ and the individual weight vectors $\boldsymbol{\eta}_h$ is appropriate:

$$dist(\mathbf{x}_j, \boldsymbol{\eta}_h) = \frac{K}{\sum_{k=1}^{K} z_{jk}} \sqrt{\sum_{k=1}^{K} (z_{jk}(x_{jk} - \eta_{hk}))^2} \quad \forall h, j, \text{ with } z_{jk} = \begin{cases} 0, & \text{if } x_{jk} \text{ is missing} \\ 1, & \text{else} \end{cases}$$
$$\forall j, k.$$

In doing so, we take into account that the distances between the input vectors and the weight vectors directly depend on the number of observed and missing values respectively. Additionally weighting these distances with the reciprocal share of observed values improves the comparability with those distances that result from complete input vectors. The whole procedure consists of four steps:

1. Read the whole, but incomplete $J \times K$ data matrix $\mathbf{X}$.
2. Delete all incomplete input vectors.
3. Adapt SGNN to the reduced data matrix $\mathbf{X}'$
4. Complete $\mathbf{X}$ by imputing values from that weight vectors which have a minimal distance to the respective incomplete input vectors.

Table 3 shows the imputations for six (i.e. 5 %) randomly missing values. The true values are given in the first row, followed by the imputations we get when applying the modified SGNN approach. In case of MCMC the multiple imputation tool provided by the SAS® procedure MI was applied. The confidence level was set to 0.05 and the number of imputations was limited to 5. Both values equal the SAS® defaults. In the last row of the table, the means of the above MCMC imputations are listed.

Table 3. Missing value imputation for the price data

True values	100.80	25.30	63.30	60.20	87.10	41.50	*MPD*
SGNN	99.50	26.10	65.40	60.80	85.07	39.41	2.689
MCMC 1	101.32	30.69	72.96	69.68	83.79	37.00	11.245
MCMC 2	100.45	33.13	78.91	96.31	76.28	51.86	25.554
MCMC 3	103.25	30.60	74.41	67.64	87.26	40.41	9.350
MCMC 4	98.02	29.91	81.30	80.75	100.81	54.03	21.581
MCMC 5	104.59	32.56	65.44	62.93	96.73	54.69	13.868
MCMC Ø	101.53	31.38	74.60	75.46	88.97	47.60	14.133

In order to compare the goodness of imputation, the Mean Percentage Deviation

$$MPD = \frac{100}{no.\ of\ m.v.} \sum_{m.v.} \frac{|true\ value - imputed\ value|}{|true\ value|},$$

is given in the last column (m. v. = "missing values"). Both the computed imputation values and the *MPD* values indicate that the considered methods are able to generate acceptable results. However, the SGNN approach seems to slightly outperform the benchmark, also when considering the averaged MCMC values given in the last row of the table.

For further analyzing the performance of both approaches, the number of missing values in the data set has been varied systematically. The *MPD* values for different numbers of missing values are displayed in Table 4. The results confirm the impression we already got from Table 3. From the *MPD* values, which result when using the means of the five MCMC values (see last row), the conclusion can be drawn that the SGNN approach mostly keeps pace with the best MCMC imputation.

Table 4. *MPD* values for different numbers of missing values (price data)

No. of m. v.	1	2	3	4	5	10	15	20	25
SGNN	4.56	6.64	4.82	7.31	10.98	8.73	10.53	9.65	9.46
MCMC 1	4.43	4.53	6.89	17.01	13.00	13.11	11.73	13.40	11.59
MCMC 2	3.05	4.06	4.70	11.43	7.38	11.04	13.44	10.25	11.48
MCMC 3	4.17	10.17	10.62	7.05	14.85	14.94	12.41	12.37	10.55
MCMC 4	13.68	6.91	3.89	10.72	11.66	14.81	14.71	14.84	12.37
MCMC 5	1.70	5.13	5.59	16.86	7.27	14.93	9.44	12.50	10.75
MCMC Ø	5.41	4.46	4.47	9.31	7.87	8.63	10.04	11.06	9.15

Table 5. *MPD* values for different percentages of missing values (Iris data)

No. of m. v.	68	61	58	58	68	54	77	64	66	64	Mean
SGNN	9.39	10.97	11.19	8.95	11.33	10.58	10.84	8.12	11.48	12.68	10.55
MCMC 1	20.08	14.30	20.51	12.41	24.68	15.96	15.30	12.24	13.80	5.54	
MCMC 2	13.41	14.52	17.02	12.93	20.40	10.64	16.82	14.64	17.93	17.20	
MCMC 3	19.26	20.42	20.62	16.76	26.02	20.77	22.30	14.10	13.10	17.70	17.33
MCMC 4	16.71	15.41	18.38	15.48	19.85	21.24	17.57	12.13	16.14	20.69	
MCMC 5	17.96	15.19	21.64	15.72	19.48	32.01	15.23	10.44	21.30	22.49	
MCMC Ø	13.66	9.74	11.98	11.20	15.77	14.33	11.72	9.84	13.23	16.57	12.80
No. of m. v.	116	129	114	126	113	113	109	117	120	105	Mean
SGNN	10.19	11.38	16.65	10.53	11.33	8.71	10.96	21.44	18.75	12.38	13.23
MCMC 1	21.51	16.03	23.53	20.54	20.76	21.36	22.38	22.52	19.86	23.64	
MCMC 2	14.44	16.29	24.11	16.22	20.47	21.14	21.72	30.21	22.11	23.13	
MCMC 3	16.61	17.91	25.94	16.91	22.50	18.37	21.17	25.84	24.21	27.84	20.89
MCMC 4	16.51	16.81	22.43	17.56	16.55	16.74	17.11	23.11	23.55	20.14	
MCMC 5	19.10	18.65	24.97	17.91	21.23	16.52	23.50	33.25	18.55	21.14	
MCMC Ø	13.80	12.43	15.92	12.61	15.74	12.28	15.76	21.40	15.67	17.95	15.36

But what happens if the data do not or not completely satisfy the Gaussian assumption? To answer this question the Iris data set was used. Table 5 shows the *MPD* values for approximately 10 percent (upper block) and 20 percent (lower block) of missing values respectively. In both cases the results show a certain superiority of the SGNN approach, particularly when considering the non-weighted averages of the *MPD* 's in the last column.

Further experiments with the above data sets, but also with a 1400×10 synthetic data set, largely support the estimation that growing neural gas-like networks, such as SGNN, constitute a promising alternative for missing value imputation if the data are Gaussian distributed, as well as in cases where this assumption does not hold. However, before being able to draw more general implications from this, additional simulations using different data sets and including further imputation methods are necessary.

5. Conclusion and outlook

In the present paper basic concepts of different growing neural-gas like networks have been outlined, and it was demonstrated that the available algorithms can be easily adapted to varying clustering tasks. The selected examples have shown that the presented approaches work quite well, even if the data sets are large or if the data are generated dynamically. At the same time the application of the algorithms promoted in this paper requires only little knowledge about the structure of the data to be analyzed.

But there are still a couple of white spots which have to be filled in the future. This, for instance, concerns the systematic investigation of the topology preservation capability and the additional inclusion of sophisticated criteria for determining the optimal number of nodes, such as the minimum description length (see [Qin and Suganthan 2004]) for a GNGN based approach). Analyzing the performance of the algorithms on high-dimensional data streams is also up to future research. Finally, magnification control is a topic that gains importance if the data to be analyzed is characterized by different density regions (see [Hammer et al. 2007]) for a recent discussion of this problem in the NGN context). Altogether, the development of NGN based algorithms which are not only capable of reliably determining the "natural" number of clusters but are also insensitive to parameter initialization, the presence of outliers and the ordering of the input data is a challenging task for future research.

References

Allison P.D. (2001), *Missing Data*, Thousand Oaks, Sage University Papers Series.
Brassington F., Pettitt S. (2005), *Essentials of Marketing*, Harlow, Prentice Hall.

Carpenter J.R., Kenward M.G., Vansteelandt S. (2006), *A Comparison of Multiple Imputation and Inverse Probability Weighting for Analyses with Missing Data*, „Journal of the Royal Statistical Society", Series A, vol. 169, no. 3, p. 571-584.

Cottrell M., Hammer B., Hasenfuß A., Villmann T. (2006), *Batch and Median Neural Gas*, „Neural Networks", vol. 19, p. 762-771.

Decker R. (2005), *Market Basket Analysis by Means of a Growing Neural Network*, „International Review of Retail, Distribution and Consumer Research", vol. 15, no. 2, p. 151-169.

Decker R. (2006), *A Growing Self-organizing Neural Network for Lifestyle Segmentation*, „Journal of Data Science", vol. 4, no. 2, p. 147-168.

Decker R., Holsing C., Lerke S. (2006), *Generating Normally Distributed Clusters by Means of a Self-organizing Growing Neural Network – An Application to Market Segmentation*, „International Journal of Computer Science", vol. 1, no. 2, p. 138-144.

Decker R., Monien K. (2003), *Market Basket Analysis with Neural Gas Networks and Self-organizing Maps*, „Journal of Targeting, Measurement and Analysis for Marketing", vol. 11, no. 4, p. 373--386.

Estévez P.A., Figueroa C.J. (2006), *Online Data Visualization Using the Neural Gas Network*, „Neural Networks", vol. 19, p. 923-934.

Fisher R.A. (1936), *The Use of Multiple Measurements in Taxonomic Problems*, „Annals of Eugenics", vol. 7, p. 179-188.

Fritzke B. (1995), *A Growing Neural Gas Network Learns Topologies*, [in:] G. Tesauro, D.S. Touretzky, T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, MIT Press, p. 625-632.

Giudici P., Passerone G. (2002), *Data Mining of Association Structures to Model Consumer Behaviour*, „Computational Statistics & Data Analysis", vol. 38, p. 533-541.

Hamerly G., Elkan C. (2003), *Learning the k in k-means*, Proceedings of the 17th Annual Conference on Neural Information Processing Systems – NIPS, p. 281-288.

Hammer B., Hasenfuss A., Villmann, T. (2007), *Magnification Control for Batch Neural Gas*, „Neurocomputing", vol. 70, p. 1225-1234.

Horton N.J., Lipsitz S.R. (2001), *Multiple Imputation in Practice: Comparison of Software Packages for Regression Models with Missing Values*, „The American Statistician", vol. 55, no. 3, p. 244-254.

Kohonen T. (2001), *Self-Organizing Maps*, 3rd ed., Berlin, Springer.

Mardia K.V. (1970), *Measures of Multivariate Skewness and Kurtosis with Applications*, „Biometrika", vol. 57, p. 519-530.

Marsland S., Nehmzow U., Shapiro J. (2005), *On-line Novelty Detection for Autonomous Mobile Robots*, „Robotics and Autonomous Systems", vol. 51, p. 191-206.

Marsland S., Shapiro J., Nehmzow U. (2002), *A Self-Organising Network that Grows when Required*, Neural Networks", vol. 15, no. 8-9, p. 1041-1058.

Martinetz T., Schulten K. (1991), *A 'Neural Gas' Network Learns Topologies*, [in:] T. Kohonen, K. Maekisara, O. Simula, J. Kangas (eds.), *Artificial Neural Networks*, Amsterdam, North Holland, p. 397-402.

Mild A., Reutterer T. (2003), *An Improved Collaborative Filtering Approach for Predicting Cross-category Purchases Based on Binary Market Basket Data*, „Journal of Retailing and Consumer Services", vol. 10, p. 123-133.

Orlowska M.E., Sun X., Li X. (2006), *Can Exclusive Clustering on Streaming Data be Achieved?*, „ACM SIGKDD Explorations Newsletter", vol. 8, issue 2, p. 102-108.

Papastefanou G., Schmidt P., Börsch-Supan A., Lüdtke H., Oltersdorf U. (eds.) (2001), *Social Research with Consumer Panel-Data*, Mannheim, GESIS.

Questier F., Guo Q., Walczak B., Massart D.L., Boucon C., de Jong S. (2002), *The Neural-Gas Network for Classifying Analytical Data*, „Chemometrics and Intelligent Laboratory Systems", vol. 61, no. 1-2, p. 105-121.

Qin A.K., Suganthan P.N. (2004), *Robust Growing Neural Gas Algorithm with Application in Cluster Analysis*, „Neural Networks", vol. 17, p. 1135-1148.

Rubin D.B. (2004), *Multiple Imputation for Nonresponse in Surveys*, Hoboken, John Wiley & Sons.
Russell G.J., Petersen A. (2000), *Analysis of Cross Category Dependence in Market Basket Selection*, „Journal of Retailing", vol. 76, no. 3, p. 367-392.
SAS (2003), *Online User Manual – Release 9.1*, SAS Institute, Cary/NC.
Schafer J.L. (1997), *Analysis of Incomplete Multivariate Data*, New York, Chapman & Hall.
Schwager S.J. (1985), *Testing for Multivariate Normality*, [in:] S. Kotz, N.L. Johnson (eds.), *Encyclopedia of Statistical Sciences*, vol. 6, New York, Wiley, p. 122.
Sharma S. (1996), *Applied Multivariate Techniques*, New York, Wiley.

ROZWÓJ SIECI NEURONOWYCH TYPU „GAS" I ICH ZASTOSOWANIE DO ANALIZY DANYCH I KLASYFIKACJI W BADANIACH MARKETINGOWYCH

Streszczenie

Artykuł przedstawia najnowsze osiągnięcia teoretyczne w obszarze sieci neuronowych typu „gas" i wskazuje, jak ta stosunkowo nowa i skuteczna klasa algorytmów może być zastosowana do analizy danych i klasyfikacji. Możliwość reprezentowania różnych struktur danych jest użyteczna w klasyfikacji, np. w segmentacji rynku. W artykule przedstawiona jest efektywność różnych algorytmów w rozpoznawaniu obrazów i klasyfikacji na przykładzie hipotetycznych oraz rzeczywistych danych marketingowych. Omawiane zastosowania obejmują typologie konsumentów, analizę koszyka rynkowego oraz kwestię brakujących danych.