J Mol Model (2004) 10:204–211
DOI 10.1007/s00894-004-0186-9

ORIGINAL PAPER

Alireza Givehchi · Gisbert Schneider

Impact of descriptor vector scaling on the classification of drugs and nondrugs with artificial neural networks

Received: 29 August 2003 / Accepted: 3 February 2004 / Published online: 6 April 2004

Abstract The influence of preprocessing of molecular descriptor vectors for solving classification tasks was analyzed for drug/nondrug classification by artificial neural networks. Molecular properties were used to form descriptor vectors. Two types of neural networks were used, supervised multilayer neural nets trained with the back-propagation algorithm, and unsupervised self-organizing maps (Kohonen maps). Data were preprocessed by logistic scaling and histogram equalization. For both types of neural networks, the preprocessing step significantly improved classification compared to nonstandardized data. Classification accuracy was measured as prediction mean square error and Matthews correlation coefficient in the case of supervised learning, and quantization error in the case of unsupervised learning. The results demonstrate that appropriate data preprocessing is an essential step in solving classification tasks.

Keywords Cheminformatics · Drug-likeness · Prediction · Self-organizing map · Structure–activity relationship

Abbreviations *BP:* Back-propagation algorithm · *GDA:* Gradient descent with adaptive learning rate · *GDM:* Gradient descent with momentum · *HTS:* High-throughput screening · *LM:* Levenberg–Marquardt · *mcc:* Matthews correlation coefficient · *MFFN:* Multilayer feedforward neural network · *mse:* Mean square error · *QE:* Quantization error · *QSAR:* Quantitative structure–activity relationship · *RP:* Resilient back-propagation algorithm · *SOM:* Self-organizing map · *SVM:* Support vector machine · *TE:* Topology error

A. Givehchi (✉) · G. Schneider
Institut für Organische Chemie und Chemische Biologie,
Johann Wolfgang Goethe-Universität,
Marie-Curie-Str. 11, 60439 Frankfurt, Germany
e-mail: alireza.givehchi@chemie.uni-frankfurt.de
Tel.: +49 (0 69 79829824)

A. Givehchi
Klinik und Poliklinik für Neurochirurgie,
Klinik und Poliklinik für Neurologie,
Universitätsklinikum Münster,
Albert-Schweitzer-Str. 33, 48129 Münster, Germany

Introduction

Ever since the "drug-likeness" approach was pioneered in independent studies of Ajay and coworkers [1] as well as Sadowski and Kubinyi [2] some years ago, several machine learning systems performing various kinds of "likeness" predictions have been developed. [3, 4] The basic idea is to define two classes of compounds, one sharing a desired property (the positive set), and another lacking this property (the negative set). Then, a binary classifier, e.g. a separating hyperplane, is developed which is applied to early-phase virtual screening and compound library shaping. [5] The theory of artificial neural networks provides us with several methods for classification and function approximation. [6] Such systems have been successfully applied to many QSAR studies. [6, 7, 8] A more recent addition to the cheminformatics toolbox is the support vector machine (SVM) which was originally developed as a binary classifier system and is often compared to neural network-based prediction routines. [9] Independent of the particular classification method used and the particular project under investigation, appropriate preprocessing of data is essential for successful feature extraction. In the present study we used drug/nondrug classification as an example application to investigate the influence of data scaling on the classification ability of two types of neural networks: (i) the supervised, multilayer feedforward neural network (MFFN) [10, 11] trained by the back-propagation (BP) algorithm, [12] and (ii) the unsupervised self-organizing map (SOM). [13, 14, 15, 16] To measure the influence of data preprocessing on the classification results obtained by both methods, we determined the correlation coefficient according to Matthews [17] and calculated the topology error and quantization error in the case of SOM, [18] and the prediction mean square error in the case of the BP-networks. As a result of this work we found that

Table 1 MOE abbreviations and short description of the descriptors used (for details, see URL: http://www.chemcomp.com/Journal_of_CCG/Features/descr.htm)

No.	Abbreviation		No.	Abbreviation	
1	VDistEq	Vertex distance equality index	17	SlogP_VSA6	Bin 6 SlogP (0.20, 0.25)
2	VDistMa	Vertex distance magnitude index	18	SlogP_VSA7	Bin 7 SlogP (0.25, 0.30)
3	wienerPath	Wiener path number	19	SlogP_VSA8	Bin 8 SlogP (0.30, 0.40)
4	wienerPol	Wiener polarity number	20	SlogP_VSA9	Bin 9 SlogP (0.40, 1.0)
5	Weight	Molecular weight	21	SMR	Molecular refractivity
6	VAdjEq	Vertex adjacency equality	22	SMR_VSA0	Bin 0 SMR (0.000, 0.110)
7	VAdjMa	Vertex adjacency magnitude	23	SMR_VSA1	Bin 1 SMR (0.110, 0.260)
8	zagreb	Zagreb index	24	SMR_VSA2	Bin 2 SMR (0.260, 0.350)
9	vsa_acc	Approximation to the sum of VDW surface areas of pure hydrogen bond acceptors	25	SMR_VSA3	Bin 3 SMR (0.350, 0.390)
10	vsa_acid	Approximation to the sum of VDW surface areas of acidic atoms	26	SMR_VSA4	Bin 4 SMR (0.390, 0.440)
11	vsa_base	Approximation to the sum of VDW surface areas of basic atoms	27	SMR_VSA5	Bin 5 SMR (0.440, 0.485)
12	vsa_don	Approximation to the sum of VDW surface areas of pure hydrogen bond donors	28	SMR_VSA6	Bin 6 SMR (0.485, 0.560)
13	vsa_hyd	Approximation to the sum of VDW surface areas of hydrophobic atoms	29	SMR_VSA7	Bin 7 SMR (0.560, 1.0)
14	vsa_other	Approximation to the sum of VDW surface areas of atoms typed as "other" (see 9–13)	30	TPSA	Topological polar surface area
15	vsa_pol	Approximation to the sum of VDW surface areas of polar (both hydrogen bond donors and acceptors) atoms (such as -OH)	31	vdw_area	Area of van der Waals surface calculated using a connection table approximation
16	SlogP_VSA	Bin 5 SlogP (0.10, 0.15)	32	vdw_vol	VDW volume calculated using a connection table approximation

classification accuracy can be significantly improved when data preprocessing by scaling is applied.

Materials and methods

Data preparation and preprocessing

The data set which was used here was based on a compilation of "drugs" and "nondrugs" by Sadowski and Kubinyi. [2] For the present study, 4,998 drugs and 4,210 nondrugs were used. [19] 32 descriptors were generated for each compound with the software package MOE (Molecular Operation Environment [20]), resulting in a 32-dimensional vector representation (Table 1). This set of descriptor belongs to the 2D descriptors provided by MOE. [20]

For data preprocessing we used either logistic scaling or histogram equalizing, as implemented in the SOM toolbox of the MatLab software package. [21, 22] These scaling methods are part of the toolbox function "som_scale". Both method scale the vectors to the range [0,1].

Logistic scaling, which is also called "softmax" transformation, first performs variance scaling and then transforms the data with the following logistic function.

$$v = \frac{1}{1 + e^{-\left(\frac{v-\bar{v}}{\sigma}\right)}} \tag{1}$$

where $\mathbf{v}$ is the descriptor vector, $\bar{v}$ is the mean value of $\mathbf{v}$, and σ is the standard deviation of $\mathbf{v}$. Histogram equalization [21] transforms the descriptor vectors in three steps: (i) ascending ordering of the values of the descriptor vector; (ii) replacing the values by their ordinal numbers, and (iii) scaling of the ordinal numbers to the interval [0,1].

Multilayer feedforward neural networks with back-propagation

MFFNs with up to two layers of hidden units were used to find a classifier separating drugs from nondrugs. [23] There exists a variety of algorithms to train a multilayer neural net. In this study we relied on the BP method [10, 11, 12, 24] in combination with several algorithms for updating weights and biases: (i) Levenberg–Marquardt (LM), [24, 25] (ii) gradient descent with momentum (GDM) [24, 26], (iii) gradient descent with adaptive learning rate (GDA), [24, 26] and (iv) resilient back-propagation (RP). [24, 26, 27] All networks were implemented within the MATLAB programming environment using the Neural Network Toolbox. [22, 24] Below we describe how the BP algorithm works for the particular implementation used in the present study.

If we try to predict the affiliation of the M descriptor vectors $\vec{x}^{\alpha}$ to the classes $\vec{y}^{\alpha}$ through the network function $\vec{f}$ and the weights $\{\vec{w}\}$, then we will have a prediction error E:

$$E = \lim_{M\to\infty} \frac{1}{2M} \sum_{\alpha=1}^{M} \left\{ \vec{y}^{\alpha} - \vec{f}(\vec{x}^{\alpha}, \{w\}) \right\}^2 \tag{2}$$

In order to minimize E through the variation of the weight vectors $\{\vec{w}\}$, we can change the weights according to Eq. (3).

$$\Delta w_{ij}^{\nu\gamma} = -\varepsilon \frac{\partial E^{\alpha}}{\partial w_{ij}^{\nu\gamma}} \tag{3}$$

where $w_{ij}^{\nu\gamma}$ are the weights between the jth neuron of the layer γ to the ith neuron of layer ν. Equation (3) is a form of the general "Delta-rule" (or Widrow-Hoff rule) for supervised parameter update. [28] As one can see, the alteration of the weight vectors depends on the negative gradient of the error surface. Some of the weight update methods use additional values for the calculation of $w_{ij}^{\nu\gamma}$, e.g. gradient descent with momentum (GDM), to prevent premature convergence of the weight optimization process. ϵ is called the "learning rate" which is changed during the training procedure from higher values to lower value. In this way we will have higher weight vector alteration at the beginning of the net

training and lower changes at the end. In contrast, gradient descent with adaptive learning rate (GDA) changes the learning rate dependent on the difference from the old error and actual error to have a faster convergence. The smaller the term $\partial E^{\alpha} \Big/ \partial w_{ij}^{\nu\gamma}$ is, the smaller the alteration of the weight vectors will be. Obviously, if the optimum is not near the actual position then a small value of $\partial E^{\alpha} \Big/ \partial w_{ij}^{\nu\gamma}$ will not be advantageous. Resilient back-propagation (RP) is one of the methods to prevent this behavior.

Having defined the method used for calculating weight changes (Eq. 3), the technique of back-propagation of errors was applied to perform the actual network update. In BP, the network error E is promoted from the output layer to the input layer of the network. The error calculated at the output will be calculated from the desired output values y and the actual output values produces by the network (Eq. 4). For reasons of simplicity we assume that all neuron thresholds Θ_i^{ν} are zero and all transfer functions $\sigma_i^{\nu}(x)$ are identical, obtaining $\sigma_i^{\nu}(x) = \sigma(x)$; $\Theta_i^{\nu} = 0 \,\forall\, i, \nu$. (*Note*: In this study we used the standard sigmoidal transfer function for the hidden neurons, a single linear output neuron, and fan-out input neurons [23]). The desired output values were 1 for drugs and 0 for nondrugs.

$$E = \frac{1}{2}\sum_i \left\{ y_i - \underbrace{\sigma_i\left(\sum_j w_{ij} s_j^{L-1}\right)}_{f_i\left(\underline{x},\{w\}\right)} \right\}^2 \tag{4}$$

where s_j^{L-1} are the outputs of the layer before the output layer (the last hidden layer). L is the total number of layers (three layers were used in this study). Inserting Eq. (4) in Eq. (5) to calculate the weight changes,

$$\Delta w_{ij}^{LL-1} = -\varepsilon \frac{\partial E}{\partial w_{ij}^{LL-1}} \tag{5}$$

we obtain for the weight update in the output layer (Eq. 6):

$$\begin{aligned} \Rightarrow \quad \Delta w_{ij}^{LL-1} &= -\frac{\varepsilon}{2} 2(y_i - f_i)(-1)\sigma'\left(\sum_j w_{ij}^{LL-1} s_j^{L-1}\right) s_j^{L-1} \\ &= \varepsilon \underbrace{\sigma\prime(\%)(y_i - f_i)}_{\delta_i^L} s_j^{L-1} \\ &= \varepsilon \delta_i^L s_j^{L-1} \end{aligned} \tag{6}$$

For the next layer the weight changes are given by Eq. (7),

$$\begin{aligned} \Delta w_{ik}^{L-1L-2} &= -\varepsilon \frac{\partial E}{\partial w_{jk}^{L-1L-2}} = -\varepsilon \underbrace{\frac{\partial E}{\partial s_j^{L-1}} \frac{\partial s_j^{L-1}}{\partial w_{jk}^{L-1L-2}}}_{\text{partial derivation because of variable output}} \\ &= -\frac{\varepsilon}{2}\sum_i 2(y_i - f_i)(-1)\sigma'\left(\sum_l w_{il}^{LL-1} s_l^{L-1}\right) w_l^{LL-1} \cdot \\ &\quad \cdot \sigma'\left(\sum_l w_{il}^{LL-2} s_l^{L-2}\right) s_k^{L-2} \\ &= \varepsilon \underbrace{\delta_j^{L-1}}_{\sigma'\left(\sum_l w_{il}^{LL-2} s_l^{L-2}\right)\sum_i w_{ij}^{LL-1}\delta_i^L} s_k^{L-2} \end{aligned} \tag{7}$$

And for all other layers we can calculate the weight changes in the following way (Eq. 8), only for the connections to the input layer we must make the assumption $\nu = 1,\ s_b^0 = x_b^{\alpha}$:

$$\begin{aligned} \Delta w_{ab}^{\nu\nu-1} &= \varepsilon \underbrace{\delta_a^{\nu}}_{\sigma'\left(\sum_l w_{il}^{LL-2} s_l^{L-2}\right)\sum_i w_{ij}^{LL-1}\delta_i^L} s_b^{\nu-1} \\ \delta_b^{\nu-1} &= \sigma\prime(\%)\sum_a w_{ab}^{\nu\nu-1}\delta_a^{\nu} \\ s_a^{\nu} &= \sigma\left(\sum_b w_{ab}^{\nu\nu-1} s_b^{\nu-1}\right) \end{aligned} \tag{8}$$

The mean square error (*mse*) of the prediction gives the information how good the predicted outputs for each molecule are fitted to the desired output. To determine the accuracy of a binary classification of the molecules (here: drug or nondrug) we used a threshold of 0.5 to convert the neural network output to binary values, and the correlation coefficient according to Matthews, [17] *mcc*, was calculated (Eq. 9).

$$mcc = \frac{tp \cdot tn - fp \cdot fn}{\sqrt{(tn + fn)\cdot(tn + fp)\cdot(tp + fn)\cdot(tp + fp)}} \tag{9}$$

where *tp* is the number of true positive, *tn* is the number of true negative, *fp* is the number of false positive, and *fn* is the number of false negative predictions. The value of *mcc* is between −1 and 1. If the prediction is perfect then the value of *fp* and *fn* is 0, and therefore the value of *mcc* will be equal to 1. All MFFN models were developed using the MatLab software package. [22]

The self-organizing map

The SOM can be used for unsupervised classification of data with nonlinear distribution of classes. [13,14] Since we assume that our data have nonlinear features, we use this method to obtain accurate classification results. We used the SOM Toolbox from the Laboratory of Computer and Information Science, Helsinki University. [21] This Toolbox is also implemented in MATLAB scripting language. It includes different procedures for preprocessing of the descriptor vectors, SOM training, classification of descriptor vectors, and visualization of the results. The algorithm implemented in the SOM Toolbox is based on the Kohonen neural network or Kohonen map: [13]

1. Initialize weight vectors in the range of [−1,1]
2. Randomly select one of the training descriptor vectors $\vec{\mathbf{x}}$.
3. Find the neuron $\vec{w}_s$ with the smallest distance to the selected descriptor vector.
4. Change the weight vectors according to

$$\vec{w}_r = \vec{w}_r + \Delta\vec{w}_r$$

where

$$\Delta\vec{w}_r = \varepsilon(t) h_{rs}(\vec{x} - \vec{w}_r)$$

and

$$h_{rs} = \exp\left(\frac{-d_1(r,s)^2}{2\sigma^2}\right)$$

with

$$\sigma(t) = \sigma_{\text{initial}}\left(\sigma_{\text{initial}}/\sigma_{\text{final}}\right)^{t/t_{\max}}$$

$$\varepsilon(t) = \varepsilon_{\text{initial}}\left(\varepsilon_{\text{initial}}/\varepsilon_{\text{final}}\right)^{t/t_{\max}}$$

5. Increase the learning step: t=t+1.
6. If t<$t_{\max}$ then go to step 2, else stop.

Since this algorithm is an unsupervised learning algorithm, it does not need assertions about the classes of the descriptor vectors to train the net. In other words, this algorithm finds classes in the training data without the knowledge about the class assignments of

the molecules (e.g. the knowledge if one compound is drug or nondrug). More details about the SOM concept and implementation can be found elsewhere. [6, 23]

The U-Matrix was used to analyze the distribution of classes of a trained SOM. [14, 15, 16, 21] It is the unified distance matrix of the unit weights of the Kohonen map. For an $N\times M$ dimensional map it has the dimension $(2N-1\times 2M-1)$. The matrix elements u_{ij} represent the distance between the map units u_i and u_j (with $i\neq j$), where u_{ij} were calculated as the mean value of the surrounding matrix elements. We applied fuzzy coloring [29] to displaying the U-Matrix. With the help of the U-Matrix we wanted to identify areas of the map which have a preference for data classes and see if the two classes (drug, nondrug) are well separated. This can help understanding whether the net is well trained and the descriptors are appropriate for the classification goal.

The resolution of the trained map must be sufficiently high to be able to distinguish the descriptor vectors of the different classes. The quantization error, *QE*, gives information about this resolution (Eq. 10). [18] It is the average distance between the descriptor vectors and their respective map neurons (cluster centroids):

$$QE = \frac{1}{N}\sqrt{\sum_{i}^{N}\left(\vec{x}^{i} - \vec{m}^{i(\text{best})}\right)^2} \tag{10}$$

where $\vec{x}^{i}$ is the ith descriptor vector and $\vec{m}^{i(\text{best})}$ is the map the winner neuron for $\vec{x}^{i}$. N is the number of descriptor vectors.

The proportion of all descriptor vectors for which the first and second neuron centroids are not adjacent was calculated to measure the topology preservation of a trained SOM. This measure value is called "topology error", *TE* (Eq. 11). [18] The lower its values are, the smaller is the number of similar classes which are not adjacent on the map.

$$TE = \frac{n_{1,2\text{na}}}{N} \tag{11}$$

where $n_{1,2\text{na}}$ is the number of descriptor vectors for which the first and second neuron centroids are not adjacent. N is the number of descriptor vectors.

Results and discussion

To show the impact of data scaling on classification and prediction with neural networks, we chose two types of artificial neural networks, multilayer neural nets and the self-organizing map (Kohonen network), in combination with two scaling algorithms, histogram equalization and logistic (softmax) scaling. Classifiers were developed for the task of drug-likeness prediction. We used the neural networks to perform binary classification of the compounds (drug and nondrug). The quantization error (*QE*) and the topology error (*TE*) were calculated to assess the accuracy of the SOMs. For multilayer neural nets we employed the mean square error (*mse*), and for the net with the best results the Matthews correlation coefficient (*mcc*) was calculated.

Figure 1 shows the influence of the two scaling methods on the descriptor vectors. Both scaling methods transform the data in such a way that the data columns are

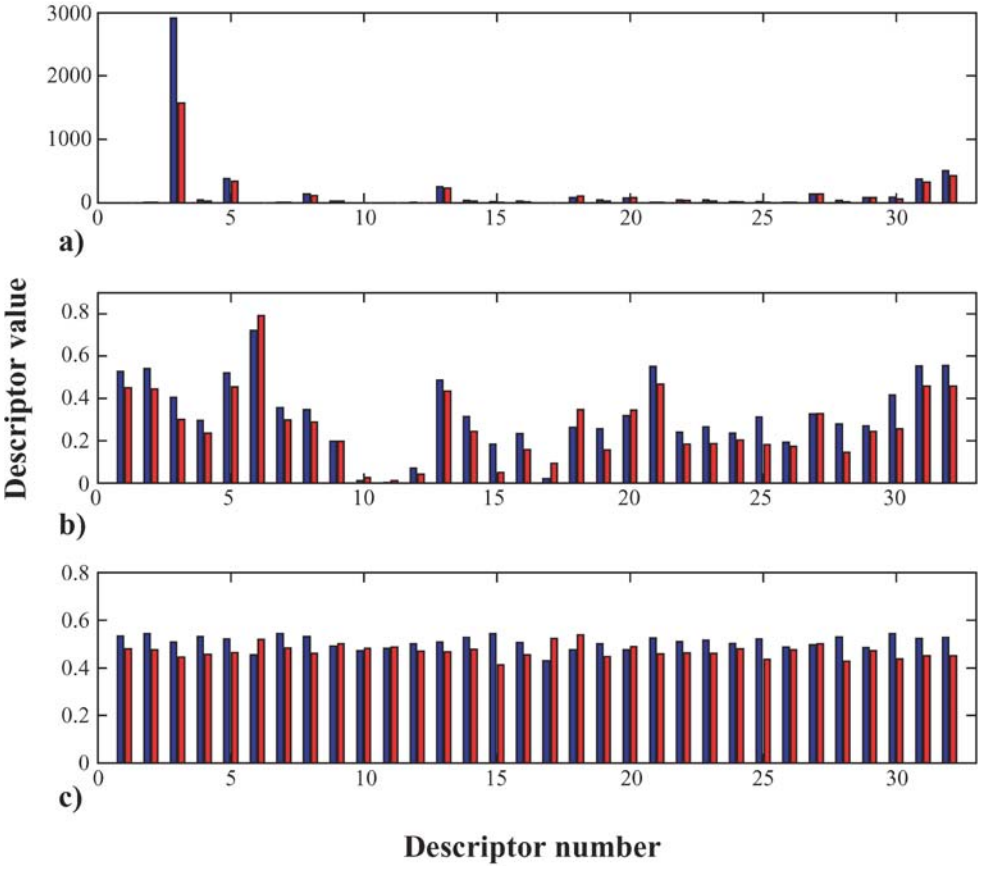

Fig. 1 Mean values of each descriptor **a** before and **b** and **c** after scaling (*blue*, drugs; *red*, nondrugs). Without scaling (**a**) an individual descriptor (the third descriptor, Wiener Path Index) dominates all other descriptors. After scaling, **a** histogram equalization, **b** logistic scaling, the contribution of the descriptors are balanced

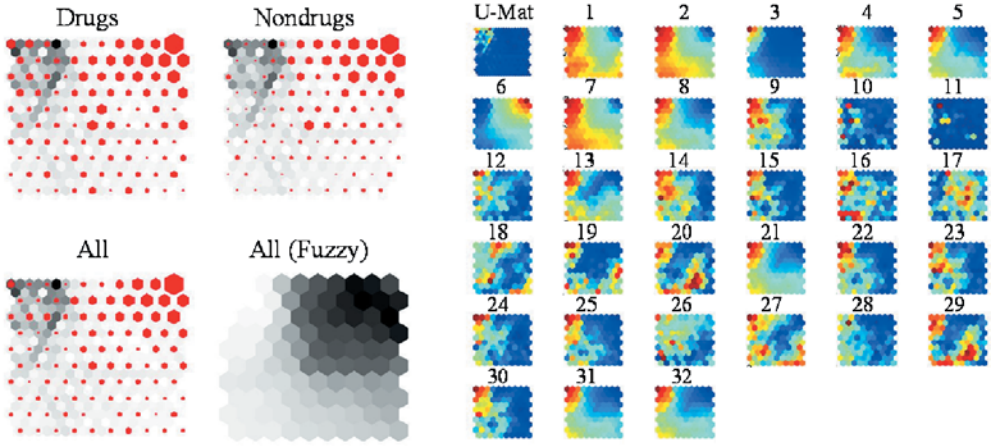

Fig. 2 SOM trained with the original, nonscaled data (*QE*=194.9; *TE*=0.42). The large windows show the distribution of sets of compounds (*red* circles) with the U-Matrix in the background (*gray*-scale). The *red* circles indicate the fraction of the selected compounds on the map areas. The lower right window is a plot of the U-matrix with "fuzzy coloring". This plot is only for better visualization of clusters on the map (*white*, no molecules; *black*, many molecules). The small maps show the U-matrix (upper left map) and the influence of each individual descriptor on the classification result (cf. Table 1)

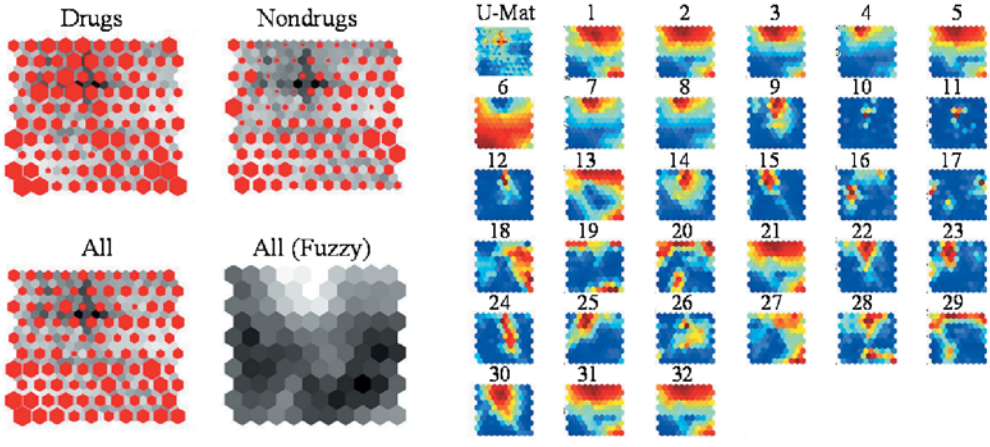

Fig. 3 Self-organizing map trained with scaled data: histogram equalization (*QE*=0.69; *TE*=0.23). See legend to Fig. 2 for explanation

standardized and thus no descriptor dominates the molecular representation simply due to large absolute original values. In other words, after scaling the features represented by the set of descriptors have equal a priori importance.

Self-organizing maps were developed using the complete data set for training. Drugs and nondrugs could not be separated using raw input data (Fig. 2). This might have led to the conclusion that the descriptors used or the particular method are not suited for this task. If, however, scaled data were used, a separation tendency of the two classes was observed on the resulting maps (Figs. 3 and 4). Obviously the scaling procedure had a positive influence on the separation of drugs and nondrugs. Logistic scaling yielded the lowest error values (*QE*=0.54, *TE*=0.29), followed by histogram equalization (*QE*=0.69, *TE*=0.23). Without scaling both error measures produced significantly higher values (*QE*=195, *TE*=0.41).

The SOM allows for an inspection of the importance of individual descriptor variables on the classification. For example, areas on the map with relatively high values of Wiener polarity (variable 4), molecular weight (variable 5), and topological polar surface area (variable 30) colocate with drug molecules (Figs. 3 and 4), whereas the descriptor "vertex adjacency equality" (variable 6) seems to be dominant in nondrug molecules. Similar attempts to understanding the underlying features were made in other studies of "drug-likeness" resulting in comparable findings. [1, 30, 31, 32] Most importantly, such an analysis is meaningless for nonscaled data (Fig. 2). We conclude that the SOM variable projection can be an additional tool to identify variables which are important for a classification and might be part of a variable selection procedure. This interpretation is supported by U-matrix inspection. By visualization of the U-matrix the distribution of the drugs and nondrugs on the map can be seen. It is evident that with normalization the map areas which belong to drug and nondrug are less overlapping than it is in the case without normalization (Figs. 2, 3, 4).

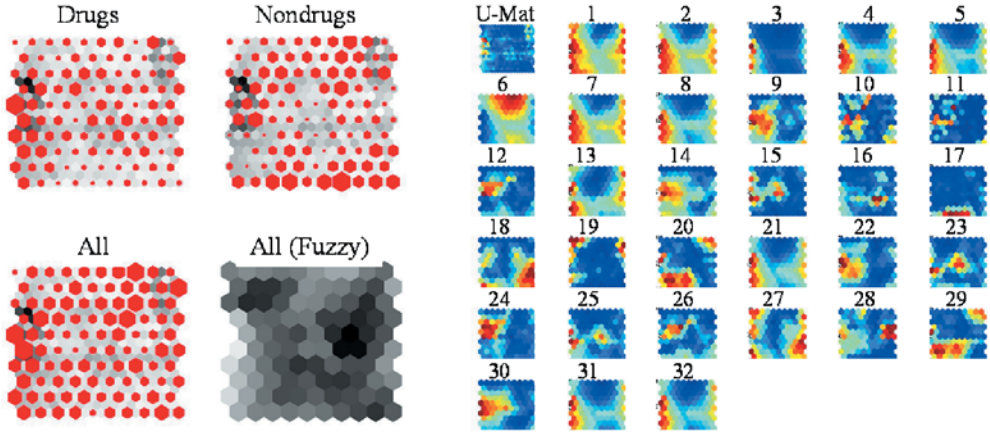

Fig. 4 Self-organizing map trained with scaled data: logistic scaling (*QE*=0.55; *TE*=0.29). See legend to Fig. 2 for explanation

Table 2 Mean square error (*mse*) of the prediction of drugs and nondrugs with a multilayer neural network. The training algorithm was back-propagation with parameter update by gradient descent with momentum (GDM), gradient descent with adaptive learning rate (GDA), and resilient back-propagation (RP)

	mse							
	GDM		GDA		RP			
No. of hidden units	6	10	6	10	6	10	(48+6)	(10+6)
Histogram equalization	0.22	0.21	0.20	0.25	0.15	0.16	0.12	0.14
Logistic scaling	0.22	0.20	0.21	0.21	0.16	0.16	0.14	0.16
No scaling	0.25	0.50	0.24	0.26	0.25	0.25	0.25	0.24

Table 3 Mean square error (*mse*) and Matthews correlation coefficient (*mcc*) of a network with (50+10) hidden units. The number of training patterns was 4,600, 2,299 for validation, and 2,300 for testing

	mse			*mcc*		
	Training	Validation	Test	Training	Validation	Test
Histogram equalization	0.13	0.16	0.15	0.66	0.55	0.57
Logistic scaling	0.14	0.15	0.16	0.61	0.57	0.58
No scaling	0.18	0.18	0.18	0.48	0.45	0.48

In the next step we trained a supervised multilayer neural network for drug/nondrug classification. In contrast to the SOM algorithm we employed a supervised learning method, the back-propagation algorithm. For updating the weights and biases we began with Levenberg–Marquardt back-propagation (LM). The network had one hidden layer containing six hidden units. The *mse* values obtained were 0.14 for histogram scaling, 0.13 for logistic scaling, and 0.41 without scaling. With logistic scaling the network model produced the most accurate prediction. To see if the scaling would deliver a lower error and better prediction with other update algorithms we employed gradient descent with momentum (GDM), gradient descent with adaptive learning rate (GDA), and resilient back-propagation algorithm (RP). Table 2 gives the *mse* for all cases. Two numbers of hidden units were tested, six and ten hidden neurons. Generally, logistic scaling and histogram equalization resulted in predictions with lower error than without scaling. In some cases logistic scaling was favorite (e.g., with GDA, ten hidden units) and in another case (e.g. with GDA, six hidden units) with histogram equalization yielded better prediction. But without scaling the highest *mse* values were obtained, irrespective of the training method and the size of the hidden layer. Overall, RP resulted in the best model. For this reason we used the RP algorithm and trained two additional networks containing two hidden layers with (48+6) hidden units and (10+6) hidden units. The results show that in all cases the scaling procedure led to a better prediction.

Since a greater number of hidden units improved the results, we then trained a net with two hidden layers and more hidden units (50+10). This time we calculated the *mse* and *mcc* for 4,600 training, 2,299 validation, and 2,300 test data (Table 3). The *mse* values were further reduced compared to smaller networks, and again without exception the scaling caused comparably smaller *mse* and greater *mcc* values for all data sets. Logistic scaling produced the best results overall.

In Figs. 5 (histogram equalization), 6 (logistic equalization) and 7 (without scaling) the output of the nets before (blue curves) and after (red points) hard-limiting for (a) training data, (b) validation data, and (c) test data are viewed. In all cases the influence of the hard limiting can be seen.

In all cases studied in this work scaling of the data led to a reduced error rate. The Matthews correlation coefficient was improved in all cases with scaling compared to

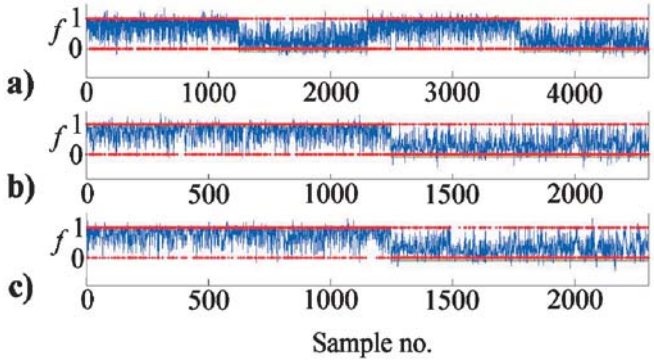

Fig. 5 Output of the multilayer neural network, *f*, trained with scaled (histogram equalization) data. Prediction of **a** training data, **b** validation data, **c** test data. Blue curves show the output values without hard-limiting. *Red* points show the hard-limited output (1 if > 0.5 and 0 if <0.5). The desired output is plotted as a *black* line

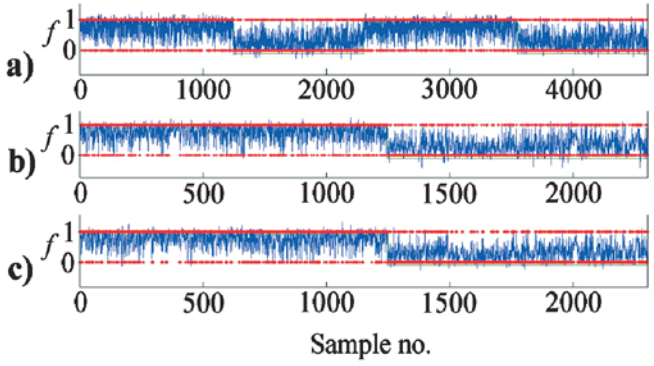

Fig. 6 Output of the multilayer neural network, *f*, trained with scaled (logistic equalization) data. Prediction of **a** training data, **b** validation data, **c** test data. *Blue* curves show the output values without hard-limiting. *Red* points show the hard-limited output (1 if > 0.5 and 0 if <0.5). The desired output is plotted as a *black* line

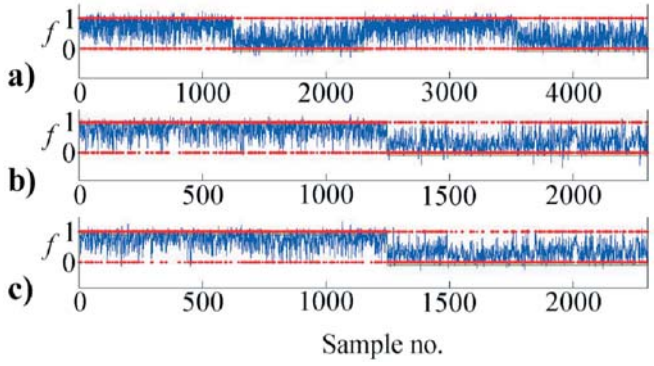

Fig. 7 Output of the multilayer neural network, *f*, trained with raw data. Prediction of **a** training data, **b** validation data, **c** test data. *Blue* curves show the output values without hard-limiting. *Red* points show the hard-limited output (1 if >0.5 and 0 if < 0.5). The desired output is plotted as a *black* line

using nonscaled data. It must be stressed that it was not the purpose of this study to find the best scaling or the best network topology for the classification task; rather we intended to demonstrate the importance of scaling whenever neural networks are used for classification. Also, we are well aware that only a single test set was used instead of multiple cross-validation, and results were not subjected to a sound statistical analysis. Keeping this in mind, we presume that the choice of an appropriate scaling method critically depends on the data set, on the set of descriptors and on the classification goal. This should be taken to account when the performance of neural networks is compared to other methods.

Acknowledgement Jens Sadowski is thanked for providing us his drug/nondrug data for the purpose of this study. This work was supported by the Beilstein-Institut zur Förderung der Chemischen Wissenschaften, Frankfurt.

References

1. Shah AV, Walters WP, Murcko MA (1998) J Med Chem 41:3314–3324
2. (a) Sadowski J, Kubinyi H (1998) J Med Chem 41:3325–3329; (b) Sadowski J (1998) In: Böhm HJ, Schneider G (eds) Virtual screening for bioactive molecules. Wiley-VCH, Weinheim, pp 117–130
3. Zuegge J, Fechner U, Roche O, Parrott NJ, Engkvist O, Schneider G (2002) Quant Struct Act Relat 21:249–256
4. Roche O, Schneider P, Zuegge J, Guba W, Kansy M, Alanine A, Bleicher K, Danel F, Gutknecht EM, Rogers-Evans M, Neidhart W, Stalder H, Dillon M, Sjögren E, Fotouhi N, Gillespie P, Goodnow R, Harris W, Jones P, Taniguchi M, Tsujii S, von der Saal W, Zimmermann G, Schneider G (2002) J Med Chem 45:137–142
5. Schneider G, Böhm HJ (2002) Drug Discov Today 7:64–70
6. Zupan J, Gasteiger J (1999) Neural networks in chemistry and drug design. An introduction. Wiley-VCH, Weinheim
7. Devillers J (ed) (1996) Neural networks in QSAR and drug design (principles of QSAR and drug design). Academic Press, New York
8. Schneider G, Wrede P (1998) Prog Biophys Mol Biol 70:175–222
9. Byvatov E, Schneider G (2003) Appl Bioinf (in press)
10. Anderson JA, Pellionisz A, Rosenfield E (eds) (1990) Neurocomputing 2: directions for research. MIT Press, Cambridge MA
11. Churchland PS, Sejnowski TJ (1992) The computational brain. MIT Press, Cambridge MA
12. Rumelhart DE, Hinton GE, Williams RJ (1986) In: Rumelhart DE, McClelland JL (eds) Parallel distributed processing, vol 1. MIT Press, Cambridge MA, pp 318–362
13. Kohonen T (1995) Self-organizing map, 2nd edn. Springer, Berlin Heidelberg New York, pp 117–119
14. Ultsch A, Siemon HP (1990) Proceedings of INNC, pp 305–308
15. Iivarinen J, Kohonen T, Kangas J, Kaski S (1994) Proceedings of Conference on Artificial Intelligence Research in Finland. pp 122–126
16. Kraaijveld MA, Mao J, Jain AK (1995) IEEE Trans Neural Networks 6:548–559
17. Matthews BW (1975) Biochim Biophys Acta 405:442–451
18. Kiviluoto K (1996) Proceedings of ICNN. pp 294–299
19. Givehchi A, Dietrich A, Wrede P, Schneider G (2003) QSAR Comb Sci 5:549–559
20. Chemical Computing Group Inc, 1010 Sherbrooke Street West, Suite 910, Montreal, Quebec, Canada, H3A 2R7; URL:http://www.chemcomp.com/Journal_of_CCG/Features/descr.htm

21. URL:http://www.cis.hut.fi/projects/somtoolbox/
22. Mathworks Inc, 3 Apple Hill Drive, Natick, MA 01760–2098, USA; URL:http://www.mathworks.com
23. Hertz J, Krogh A, Palmer R (1991) Introduction to the theory of neural computation. Addison-Wesley, Redwood City, CA
24. Demuth H, Beal M (2001) Neural network toolbox, user's guide version 4. Mathworks Inc, Natick, MA
25. Hagan MT, Menhaj M (1994) IEEE Trans Neural Networks 5:989–993
26. Hagan MT, Demuth HB, Beale MH (1996) Neural network design. PWS Publishing, Boston, MA
27. Riedmiller M, Braun H (1993) Proceedings of the IEEE International Conference on Neural Networks 1:586–591
28. Widrow B, Lehr MA (1995) Perceptrons, adalines, and backpropagation. In: Arbib MA (ed) The handbook of brain theory and neural networks. MIT Press, Cambridge, MA, pp 719-724
29. Himberg J (2000) Proceedings of the International Joint Conference on Neural Networks (IJCNN) 3:587–592
30. Takaoka Y, Endo Y, Yamanobe S, Kakinuma H, Okubo T, Shimazaki Y, Ota T, Sumiya S, Yoshikawa K (2003) J Chem Inf Comput Sci 43:1269–1275
31. Ajay (2002) Curr Top Med Chem 2:1273–1286
32. Brüstle M, Beck B, Schindler T, King W, Mitchell T, Clark T (2002) J Med Chem 45:3345-3355