PERGAMON

Neural Networks 15 (2002) 945–952

Neural Networks

www.elsevier.com/locate/neunet

2002 Special Issue

How to make large self-organizing maps for nonvectorial data

Teuvo Kohonen*, Panu Somervuo

Neural Networks Research Centre, Helsinki University of Technology, P.O. Box 5400, FIN-02015 Hut, Finland

Abstract

The self-organizing map (SOM) represents an open set of input samples by a topologically organized, finite set of models. In this paper, a new version of the SOM is used for the clustering, organization, and visualization of a large database of symbol sequences (viz. protein sequences). This method combines two principles: the batch computing version of the SOM, and computation of the generalized median of symbol strings. © 2002 Elsevier Science Ltd. All rights reserved.

Keywords: Batch map; Clustering; Data mining; Database organization; Generalized median; Protein sequence; Self-organizing map; Visualization

1. Introduction

The traditional nonlinear projection method for the ordered display of general data items between which an arbitrary similarity measure has been defined is the multi-dimensional scaling, MDS (Kruskal & Wish, 1978) with its variations. In this paper we point out that unlike in the MDS and most clustering methods, each original sample need not be represented separately, if the samples are approximated by a much smaller set of topologically ordered *model representations*. The self-organizing map (SOM) (Kohonen, 1982, 1990, 1995; Kohonen, Oja, Simula, Visa, & Kangas, 1996) is a nonlinear projection method, too, which uses such models and has been applied to a diversity of problems. In this paper it is shown how an extension of the SOM (Kohonen, 1996) can be used for the clustering, organization, and visualization of a large database of nonvectorial items, viz. protein sequences.

Usually the SOMs are defined in metric vector spaces. The new method suggested by one of the authors (Kohonen, 1996) allows the construction of the SOM for items with an arbitrary similarity measure defined between them, and thus for nonvectorial data items, too. In order to define an ordered projection, like in the MDS, it will be sufficient to compare the pairwise distances or similarities between the items. For the practical example in the present work the FASTA method (Pearson & Lipman, 1988) was used for the computation of similarities between protein sequences, picked up from the SWISS-PROT database (Bairoch & Apweiler, 1999) publicly available via the Internet.[1]

2. The self-organizing map for nonvectorial data sets

In its original form the SOM is a nonlinear projection method that maps a high-dimensional metric vector space, or actually only the manifold in which the vectorial samples are really located, onto a two-dimensional regular grid in an orderly fashion (Kohonen, 1982, 1995). A *model* representing a local subset of the data in the manifold is associated with each grid point. In an *unsupervised learning process* the models on the map will be tuned to the input data. This is implemented by *competitive learning* and the use of a *topological neighborhood* within which the models are adapted. The following two steps are repeated for each input item in the data set:

1. Find the best-matching model using the chosen similarity measure.
2. Update this model and the models belonging to its topological neighborhood in the map grid towards the prevailing input.

Let the grid points be indexed by i, and let c be the index of the model ('winner') that matches best with the prevailing input item. The updating at step 2 is controlled in such a way that the rate of the updating of the neighboring models shall be proportional to the *neighborhood function* $h_{ci}(t)$, which is a function of the input sample, and of time, too (cf. Fig. 1).

The above two steps, in a general way, define an iterative regression-type process. Before its application, the models have to be *initialized* properly. It is characteristic of the SOM processes that the initial models can even be selected randomly. The mathematical proof of this result is, however, very complicated even in the simplest cases

* Corresponding author. Tel.: +358-9-451-3268; fax: +358-9-451-5783. *E-mail address:* teuvo.kohonen@hut.fi (T. Kohonen).

[1] See e.g. http://www.expasy.ch/sprot/.

Nomenclature

$c, c(x)$	index of the best-matching unit for input x
$d(\cdot,\cdot)$	distance measure
$h_{ci}(t)$, $h_{c(\mathbf{x}(j)),i}$	neighborhood function
i,j	indices
I	the set of map unit indices
m	generalized median over set $\mathcal{S}$
m_i	general model associated with map unit i
$\mathbf{m}_i$	model vector of map unit i
t	time index
$x, x(i)$	general input item
$\mathbf{x}, \mathbf{x}(i)$	input vector

(Cottrell, Fort, & Pagès, 1997) and cannot be presented here.

It has turned out in practice that the convergence of the models to their final, stationary values proceeds significantly faster if, instead of completely random choice for their initial values, the models are even roughly ordered. For instance, if the input items and the models were Euclidean vectors, as in most applications, one can select the models as a regular array of values picked up from the two-dimensional hyperplane spanned by the two largest principal components of input.

In this paper we shall only make use of the batch-learning version of the SOM (Kohonen, 1995). Assume first that a finite set of vectorial training samples $\{\mathbf{x}(j)\}$ is available, and the models associated with the grid nodes are called $\mathbf{m}_i$. The first task is to find the best-matching model $\mathbf{m}_c$ for each $\mathbf{x}(j)$ according to

$$c(\mathbf{x}(j)) = \arg\min_i \{d[\mathbf{x}(j), \mathbf{m}_i]\}. \tag{1}$$

The new updated model is then

$$\mathbf{m}_i = \frac{\sum_j h_{c(\mathbf{x}(j)),i}\mathbf{x}(j)}{\sum_j h_{c(\mathbf{x}(j)),i}}. \tag{2}$$

However, the SOM principle is not restricted to metric vector spaces. It has been pointed out (Kohonen, 1996) that any set of items, for which a similarity or distance measure between its elements is definable, can be mapped on the SOM grid in an orderly fashion. This is made possible by the following principle, which combines the concept of the *generalized median* of a set (Kohonen, 1985, 1995) with the batch computation of the SOM.

Assume a fundamental set $\mathcal{S}$ of any kind of items $x(i)$ and let $d[x(i), x(j)]$ be some distance measure between $x(i)$ and $x(j) \in \mathcal{S}$. The generalized median m over $\mathcal{S}$ is defined

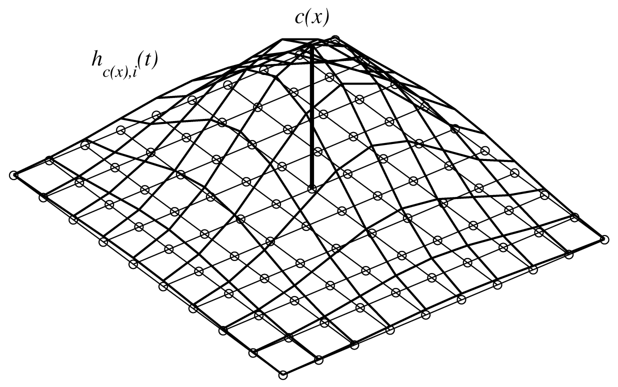

Fig. 1. A Gaussian neighborhood function on the rectangular SOM grid around the winner node. A model representing the local input manifold is associated with each grid point. The neighborhood function determines the learning rates of the models.

as the item that minimizes the objective function

$$\mathcal{D} = \sum_{x(i)\in\mathcal{S}} d[x(i), m]. \tag{3}$$

In this work, m is restricted to being an element of $\mathcal{S}$.

Notice that if the input samples had been real scalars and the distance measure were the absolute value of their difference, it is easy to show that the generalized median coincides with the arithmetic median. On the other hand, if the input samples were real vectors, if the distance measure were Euclidean, and if the item with the smallest sum of the *squares* of distances from the other items were sought, the generalized median would coincide with the arithmetic mean of the $\mathbf{x}(i) \in \mathcal{S}$.

Let us now concentrate on the special SOM that is able to map nonvectorial items. Consider Fig. 2 in which a regular grid is shown, with some general model $m_c \cdots m_p$ associated with each grid node. Assume that a sublist that contains a subset of input items $x(i)$ can be associated with each model. Each of the input items $x(1), x(2), \ldots$ is compared with all the models and listed under that one that has the smallest distance from the respective input item. The $x(1), x(2), \ldots$ will thus be distributed under the closest models.

Define for each model, say m_i, a neighborhood set N_i (the set of models located within a certain radius from the node i in the grid). Consider the union of all the sublists within N_i (shown by the set line in Fig. 2). The generalized median of N_i is defined to be identical with the input sample in the union of the sublists in N_i that has the smallest sum of distances from all the other samples of N_i, see Eq. (3).

In forming the sum of distances, the contents of the sublists within N_i can be *weighted* by the neighborhood function $h_{ci}(t)$.

For each N_i in Fig. 2, $i = c, d, \ldots, p$ the generalized median is now determined, and the old models $m_c \cdots m_p$ are replaced by the respective generalized medians, in a concurrent operation.

After this replacement, the original models have now been changed, and if the same input samples were compared

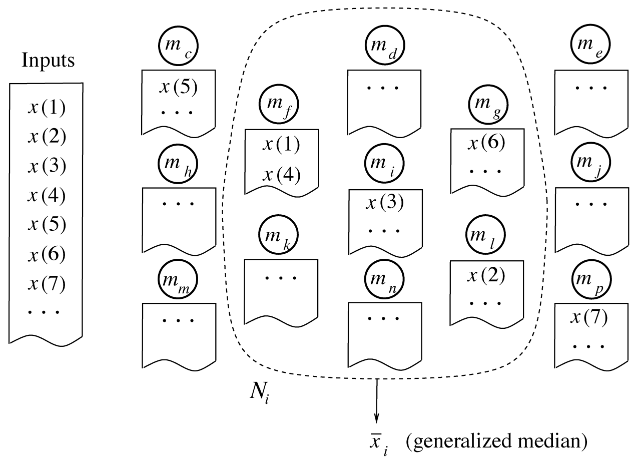

Fig. 2. Illustration of the SOM algorithm for nonvectorial data. Each of the input items $x(1), x(2), \ldots$ is copied into the sublist under that model that has the smallest distance from the respective input item. After that, the generalized median $\bar{x}_i$ in each neighborhood set is determined, and the old value, say m_i, is replaced by $\bar{x}_i$. This cycle is repeated from the beginning as many times as the models are not changed any longer.

with them, they would be redistributed in a different way in the sublists. Eventually, however, in a finite number of iterations of this type the process will converge, after which the models approximate the input samples in an orderly fashion, because each model then coincides with the generalized median of the input items mapped into its neighborhood.

It is not yet mathematically proven that the above process converges, at least into a unique equilibrium. In practice, convergence means that the lists will not be changed any longer in further iterations, and this, of course, can be observed. However, there may exist alternative states into which the map may converge, and only one of them is the global optimum. This same problem is associated with most 'neural networks'. A proof of the 'batch map' process has only been presented for vectorial items (Cheng, 1997).

Comment 1. Several criteria for the evaluation of the degree of ordering have been developed (Kaski & Lagus, 1996; Kiviluoto, 1996; Villmann, Der, & Martinetz, 1994; Zrehen, 1993).

Comment 2. Like in the traditional SOM for vectorial items (Kohonen, 1995), the radius of the neighborhood set N_i at the beginning of the process may be selected as fairly large and put to shrink monotonically in further iterations. The optimal speed of shrinking should be determined experimentally.

Comment 3. Since the generalized medians are discrete entities, some models on the map may be identical, which will result in ties between the best-matching models. A simple solution to make the winner unique is to define the winner as that model, for which the sum of the distances of the input to all models in a small neighborhood of the winner candidate is smallest, cf. Fig. 3.

3. Example: the SOM of protein sequences

To exemplify the method presented in this paper and some computational tricks that accelerate the convergence of very large nonvectorial SOMs, we applied the algorithm described in Section 2 to a very large database of 77 977 protein sequences, obtained from the SWISS-PROT database, release 37 (Bairoch & Apweiler, 1999; Somervuo & Kohonen, 2000).

In order to construct a mapping of the protein sequences, some kind of similarity measure for them had to be defined first. For arbitrary amino acid sequences it is difficult to define the 'best' measure since it depends on, for instance, whether global or local relations of the sequences are of interest. However, if the database is representative and the data are distributed densely enough in the 'sequence space', it may be argued (Pearson, 1999; Pearson & Lipman, 1988) that the method for finding close evolutionary connections and similarities is adequate, since all items in the database can then be connected to each other by the chains of local connections and close similarities. Similarities between remote sequences can be tracked by means of the chains of the closely related sequences in the sequence space. For protein sequences, e.g. the Smith–Waterman (Smith and Waterman, 1981), BLAST (Altschul, Gish, Miller, Myers, & Lipman, 1990), or FASTA (Pearson & Lipman, 1988) methods have been used. In the present work, the FASTA method was chosen.

In the earlier works where the SOM has been used for the clustering of protein sequences, the data have been converted into vectorial representations. The sequences were converted into 400-dimensional dipeptide histogram vectors by Ferrán and Ferrara (1991). Similar amino acids were grouped together before computing the histogram vectors by Ferrán, Pflugfelder, and Ferrara (1994). The aligned sequences were converted into vectors through fractal encoding by Hanke and Reich (1996). Each position of the sequence was represented as a 20-dimensional vector with each vector component corresponding to one amino acid symbol by Andrade, Casari, Sander, and Valencia (1997). The whole sequence was then converted into an L-by-20-dimensional vector, where L was the length of the global alignment of all sequences.

The method described in Section 2 allows the organization of nonvectorial data items according to their true similarities without any approximative vector-encoding.

As our experiment serves mainly as a demonstration of the new method, we decided to use a hexagonal 30-by-20-unit SOM, but there are no restrictions in using a larger map.

We have observed experimentally that the convergence of the nonvectorial SOM algorithm is significantly faster and safer, if the initial models are already two-dimensionally ordered, roughly at least. We found that the method in which the protein sequences were ordered according to the similarity of their dipeptide histograms (Ferrán & Ferrara, 1991; Ferrán et al., 1994) is useful for the

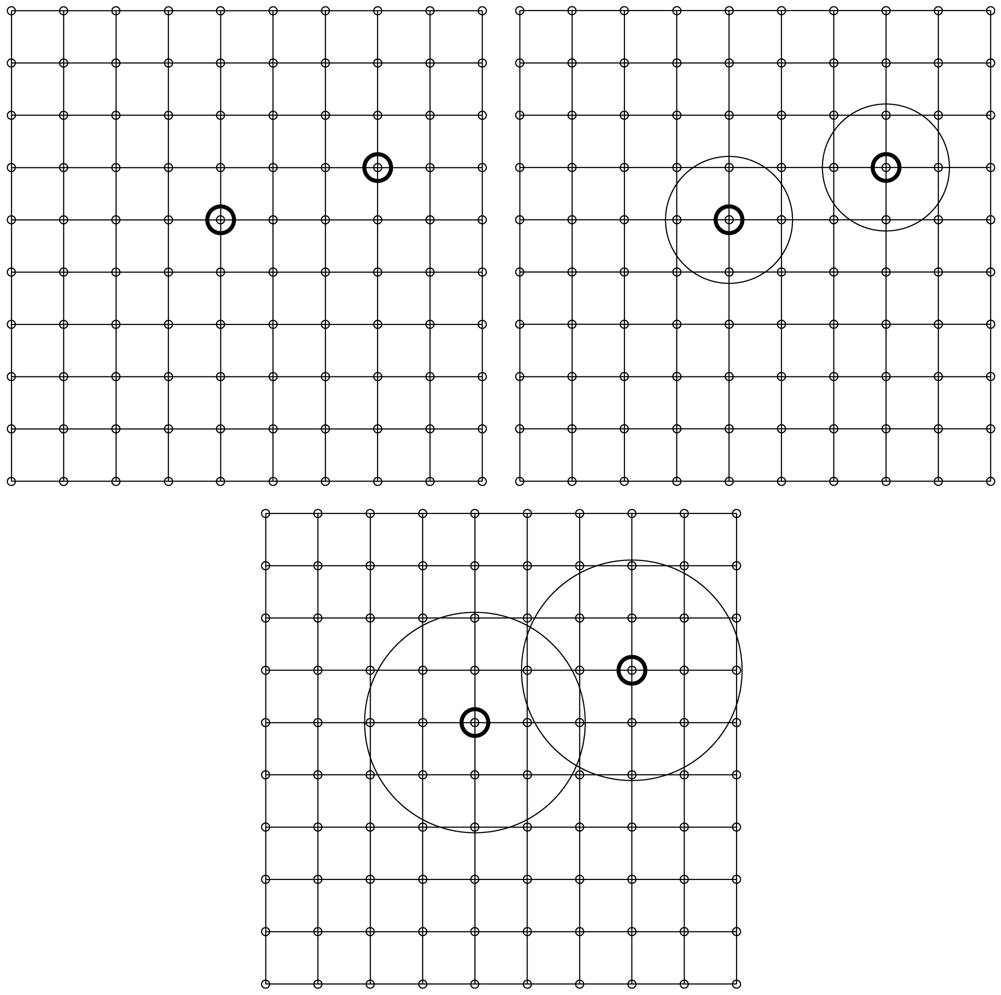
Fig. 3. Tie-breaks in the winner determination. In case two (or more) map units have the same distance to the input item, the winner is defined to be that model for which the sum of the distances of the input to all models in a small neighborhood around the winner candidate is minimum. The size of the neighborhood is increased until one of the map units becomes an unambiguous winner.

tentative definition of a rough initial order to the SOM. Then, however, extra *auxiliary model vectors* had to be introduced and associated with the nodes. The initial ordering of the vectorial models in this auxiliary SOM proceeded in the traditional way in our experiment. Each map node was first provided with a 400-dimensional model vector, each component of which was initialized with a random value between zero and unity, whereafter the vectors were normalized to unit length. Training was made by the 400-dimensional dipeptide histograms using 30 batch cycles. A Gaussian neighborhood kernel, the standard deviation of which decreased linearly from 30 to 1 during training, was used.

Next the nodes were labeled by those protein sequences that represented the medians in the sublists under the respective nodes (cf. Fig. 2). When this labeling was ready, the vectorial parts of the models could be abandoned, and the ordering could be continued by the method described in Section 2.

In the subsequent phases of learning, the true protein sequences were used as inputs as described in Section 2 and the winner nodes were determined by the FASTA method. The source code for the FASTA computation was extracted from the FASTA program package, version 3.0 (Pearson, 1999). The parameter *ktup* was set to 2, the amino acid substitution scores were taken from the BLOSUM50 matrix, and the final optimized score for the sequence similarity was computed by dynamic programming.

The symbol-SOM was trained for 20 batch cycles, using the neighborhood radius of 1. (Since the SOM was already roughly ordered, there was no need to use a shrinking kernel any longer.) Since the *sequence similarities* instead of their distances were actually computed, for the ‘median’ we had to take that sequence in the union of the neighboring sublists that had the *largest* (*instead of smallest*) sum of similarity values with respect to all the other sequences in the neighboring lists. The Gaussian neighborhood function was applied to the weighting of the similarities.

It would have presented a very high computing load to the algorithm if all the 77 977 protein sequences had been used as inputs at each batch computation cycle. The computing load could be reduced to less than 10%, without

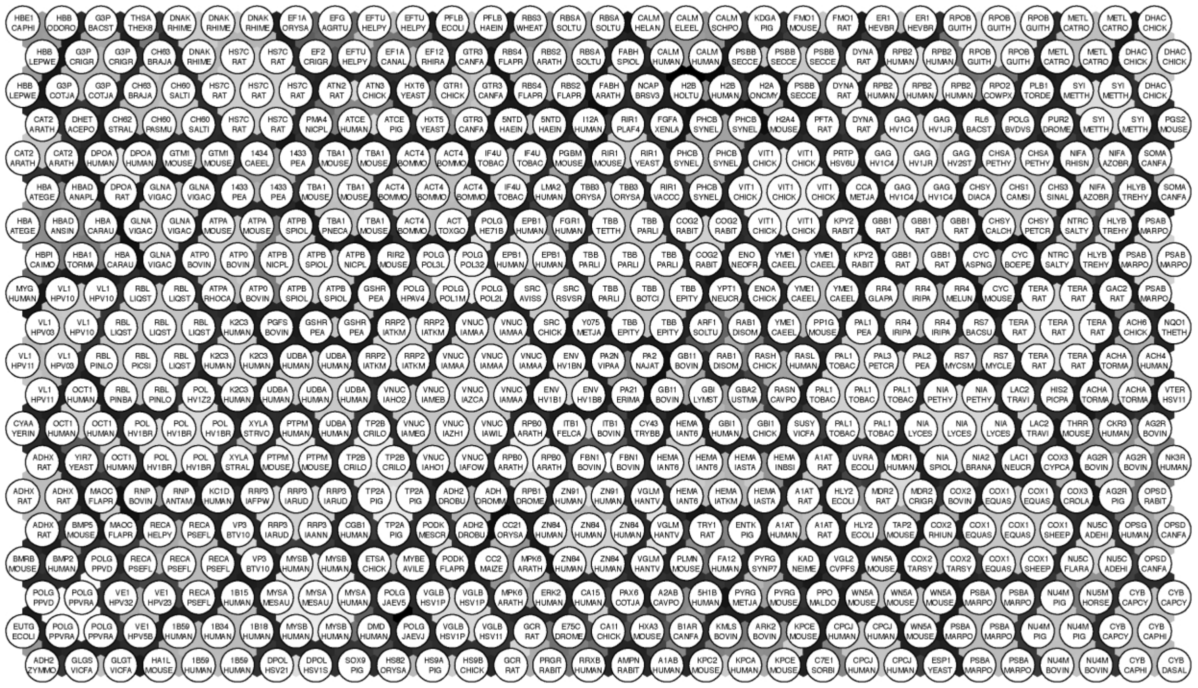
Fig. 4. A 30-by-20-unit hexagonal SOM grid. The SOM was constructed using all the 77 977 protein sequences of the SWISS-PROT release 37. Each node contains a prototype sequence and a list of data sequences. The labels on the map nodes are the SWISS-PROT identifiers (Bairoch & Apweiler, 1999) of the prototype sequences. The upper label in each map node is the mnemonic of the protein name and the lower label is the mnemonic of the species name. The similarities of the neighboring prototype sequences on the map are indicated by shades of gray. The light shades indicate a high degree of similarity, and the dark shades a low degree of similarity, respectively. Light areas on the map reveal large clusters of similar sequences.

essentially deteriorating the (statistical) accuracy of the batch computation, by randomly picking up 6000 sample sequences from the 77 977 ones for each batch cycle. After 20 such sampled training cycles, one final training cycle was carried out using all the available sequences as the inputs.

The resulting SOM is shown in Fig. 4. The map nodes that have been labeled according to the identifiers of the

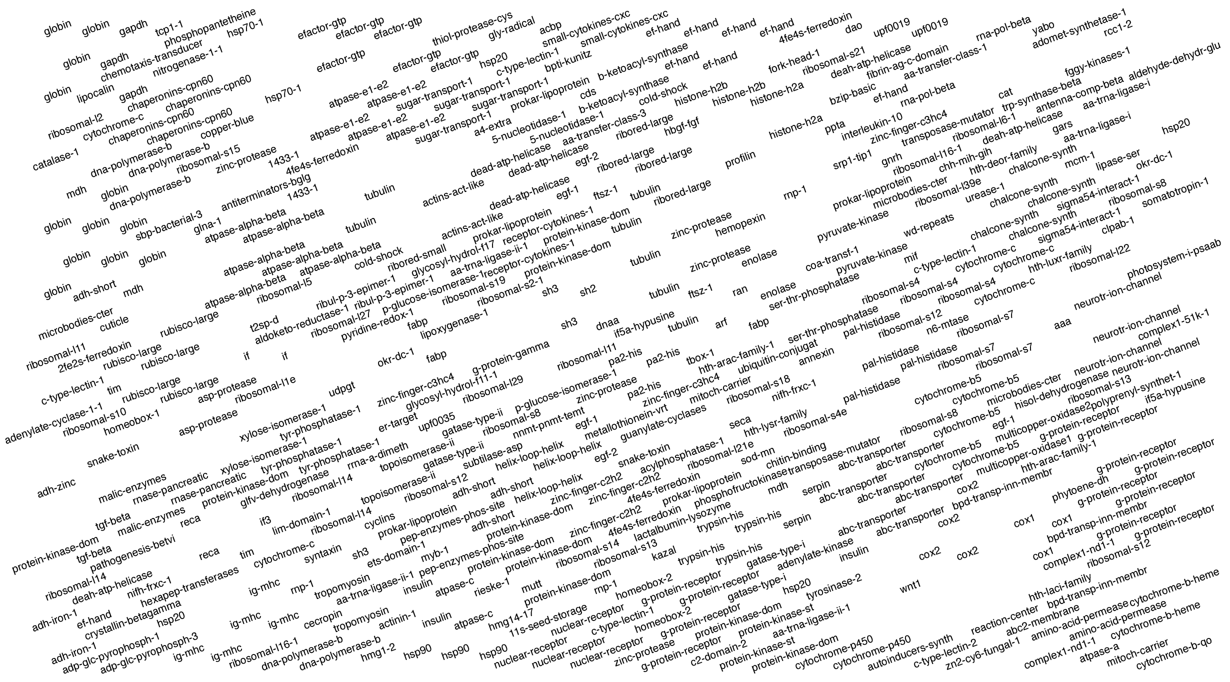
Fig. 5. Clustering of 77 977 protein sequences using a 30-by-20 unit SOM. The prototype sequences of the map nodes are the same as in Fig. 4. Each node is labeled by majority voting of the sequences having that node as their best-matching unit. The labels are the PROSITE classes (Hofmann et al., 1999) of the sequences.

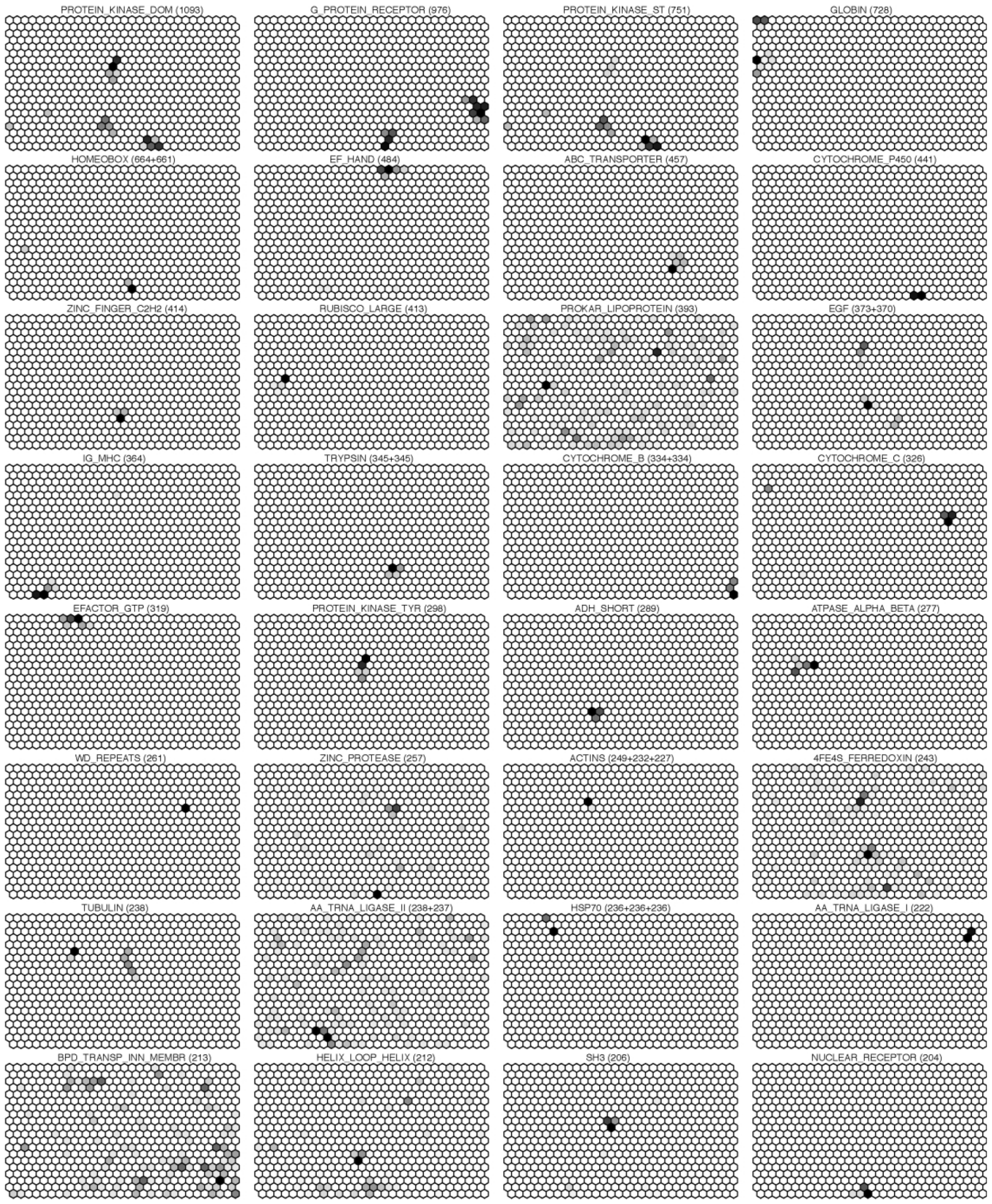

Fig. 6. Projections of the 32 most frequent PROSITE classes of the SWISS-PROT database on the SOM. Each subfigure represents the distribution of one class. The prototype sequences of the map nodes are the same as in Fig. 4. The shades of gray indicate the number of protein sequences belonging to the given class in each map node. The maximum value (darkest shade of gray) is scaled to unity in each subfigure. The total number of the sequences in each class is shown in the parentheses after the PROSITE name.

final prototypes resulted in the 'median map' method. Now consider that the $x(i)$ are generally clustered; then the probability for the occurrence of input values between the clusters is small, and there is no need to allocate space on the SOM for the data space between the clusters. In the SOM, the clusters will automatically be mapped closely nearby. In order to distinguish the clusters in the display and to demarcate their borders, one may use, e.g. special coloring methods such as the U-matrix (Kraaijveld, Mao, & Jain, 1992, 1995; Ultsch & Siemon, 1989), where the clusters are indicated by light shades and the borders with darker shades, respectively. In Fig. 4 the background areas between the models are shaded.

Once the SOM has been trained and is ordered, it is very fast to compute the projection of any new sequence on it. This requires only as many sequence comparisons as there are prototype sequences on the map. In the current work, the SOM contained 600 prototype sequences. Thus the work needed for classifying the new sequence into a prototype class is considerably lighter than comparison with all the 77 977 sequences of the whole database.

As it may be desirable to characterize the quality of the mapping produced by the nonvectorial SOM, we show the clustering of some known protein families on it.

For the map shown in Fig. 4, another labeling was carried out by listing all data sequences under the best-matching nodes and then performing a majority voting for each list according to the PROSITE classes, release 15 (Hofmann, Bucher, Falquet, & Bairoch, 1999) of the sequences. This result is shown in Fig. 5. Since the PROSITE database did not give any class for 37 743 sequences of the SWISS-PROT database, the PROSITE label of the node does not necessarily characterize all sequences of the node.

Those classes whose members are strongly similar have been mapped to a small area on the map, while some other classes are spread out more. Actins and rubisco-large are examples of the classes which form sharp areas on the map. Globin is a large family which is composed of subfamilies. The globin sequences are mostly mapped on the top-left corner of the SOM. Hemoglobin beta chains are represented on the corner, hemoglobin alpha chains are in the cluster below catalases, and myoglobins are located below hemoglobin alpha chains. One sharp cluster on top of the map consists of efactor-gtp sequences. Between globins and efactor-gtp there is a cluster of the hsp70 family. Tubulins are mapped to two closely located areas, one of which is characterized by alpha subunits and another by beta subunits, respectively.

Since there are altogether 1352 classes in the PROSITE database, not all of them can be discussed in detail. But a general idea of the capability of the SOM can be gained by investigating the projections of the most frequent classes. Therefore the PROSITE classes were sorted according to their frequency in the SWISS-PROT database. The 32 most frequent classes were then projected on the SOM by finding the best-matching unit of each sequence belonging to the given class. The resulting class distributions are shown in Fig. 6.

In the visualization of the class distributions, some PROSITE classes were combined. For example, the actins class consists of 249 sequences of the family actins_act_like, 232 sequences of actins_2, and 227 sequences of actins_1. Trypsin_ser and trypsin_his were combined to the single trypsin class. Thiol_protease_asn, thiol_protease_his, and thiol_protease_ser were combined to the single thiol_protease class. Cytochrome_b class in the figure consists of both cytochrome_b_qo and cytochrome_b_heme. The distribution of protein_kinase_atp (1040 sequences) is not shown, because it was identical with the distribution of the protein_kinase_dom (1093 sequences).

Analyzing the cluster contents according to known protein families can give information about the specificity of the prototype sequences, like in the organization of the database performed on the basis of the sequence similarities. The classification of the sequences according to the PROSITE classes, however, may also include structural information about the protein molecules. At any rate, many PROSITE classes were mapped to small and sharp areas on the SOM display.

4. Concluding remarks

Contrasted with the earlier works on SOMs, the principle used in this work makes it possible to apply any similarity measure for the mapped items. The resulting clustering and ordering of the data is expected to reflect the properties of the chosen similarity measure. The present example, where the similarities between protein sequences were computed by the FASTA method, is a two-dimensional map where similar sequences are mapped to the same node or neighboring nodes, and the structures of the clusters thereby formed are clearly visible.

An advantage of the SOM, compared with some other projection methods, is that the basic form of the algorithm is very simple, straightforward to implement, and fast to compute. The SOM can therefore be used as a data mining and visualization tool for a wide variety of data sets, for which a similarity or distance measure between its elements can be defined.

References

Altschul, F., Gish, W., Miller, W., Myers, E., & Lipman, D. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*, 403–410.

Andrade, M., Casari, G., Sander, C., & Valencia, A. (1997). Classification of protein families and detection of the determinant residues with an improved self-organizing map. *Biological Cybernetics*, *76*, 441–450.

Bairoch, A., & Apweiler, R. (1999). The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999. *Nucleic Acids Research*, *27*, 49–54.

Cheng, Y. (1997). Convergence and ordering of Kohonen's batch map. *Neural Computation*, *9*, 1667–1676.

Cottrell, M., Fort, J. C., & Pagès, G. (1997). Theoretical aspects of the SOM algorithm. *Proceedings of the Workshop on Self-Organizing Maps* (pp. 246–267).

Ferrán, E., & Ferrara, P. (1991). Topological maps of protein sequences. *Biological Cybernetics*, *65*, 451–458.

Ferrán, E., Pflugfelder, B., & Ferrara, P. (1994). Self-organized neural maps of human protein sequences. *Protein Science*, *3*, 507–521.

Hanke, J., & Reich, J. (1996). Kohonen map as a visualization tool for the analysis of protein sequences: multiple alignments, domains and segments of secondary structures. *Computer Applications in the Biosciences*, *12*(6), 447–454.

Hofmann, K., Bucher, P., Falquet, L., & Bairoch, A. (1999). The PROSITE database, its status in 1999. *Nucleic Acids Research*, *27*, 215–219.

Kaski, S., & Lagus, K. (1996). Comparing self-organizing maps. *Proceedings of the International Conference on Artificial Neural Networks* (pp. 809–814).

Kiviluoto, K. (1996). Topology preservation in self-organizing maps. *Proceedings of the IEEE International Conference on Neural Networks* (pp. 294–299).

Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, *43*, 59–63.

Kohonen, T. (1985). Median strings. *Pattern Recognition Letters*, *3*, 309–313.

Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, *78*, 1464–1480.

Kohonen, T. (1995). *Self-organizing maps*. Berlin: Springer, 3rd extended ed 2001.

Kohonen, T. (1996). Self-organizing maps of symbol strings. *Technical Report A42*, Laboratory of Computer and Information Science, Helsinki University of Technology, Finland.

Kohonen, T., Oja, E., Simula, O., Visa, A., & Kangas, J. (1996). Engineering applications of the self-organizing map. *Proceedings of the IEEE*, *84*, 1358–1384.

Kraaijveld, M., Mao, J., & Jain, A. (1992). A non-linear projection method based on Kohonen's topology preserving maps. *Proceedings of the 11th International Conference on Pattern Recognition* (pp. 41–45).

Kraaijveld, M., Mao, J., & Jain, A. (1995). *IEEE Transactions on Neural Networks*, *6*, 548–559.

Kruskal, J., & Wish, M. (1978). *Multidimensional scaling*. Newbury Park, CA: Sage Publications.

Pearson, W. (1999). The FASTA program package. ftp://ftp.virginia.edu/pub/fasta.

Pearson, W., & Lipman, D. (1988). Improved tools for biological sequence comparison. *Proceedings of the National Academy of Sciences of the United States of America*, *85*, 2444–2448.

Smith, T., & Waterman, M. (1981). Comparison of biosequences. *Advances in Applied Mathematics*, *2*, 483–489.

Somervuo, P., & Kohonen, T. (2000). Clustering and visualization of large protein sequence databases by means of an extension of the self-organizing map. In S. Arikawa, & S. Morishita (Eds.), *Proceedings of the Discovery Science* (pp. 76–85). Berlin: Springer.

Ultsch, A., & Siemon, H. (1989). Exploratory data analysis: Using Kohonen's topology preserving maps. *Technical Report 329*, University of Dortmund, Germany.

Villmann, T., Der, R., & Martinetz, T. (1994). A new quantitative measure of topology preservation in Kohonen's feature maps. *Proceedings of the IEEE International Conference on Neural Networks* (pp. 645–648).

Zrehen, S. (1993). Analyzing Kohonen maps with geometry. *Proceedings of the International Conference on Artificial Neural Networks* (pp. 609–612).