SOM^2 AS "SOM OF SOMs"

T. Furukawa
Department of Brain Science and Engineering, Kyushu Institute of Technology
Hibikino, Wakamatsu-ku, Kitakyushu 808-0196, Japan
furukawa@brain.kyutech.ac.jp

Abstract - *In this paper, an extension of SOM is proposed in which the mapping objects themselves are self-organizing maps. Thus, a "SOM of SOMs," referred to in this paper as SOM^2, is presented. In SOM^2, each nodal unit of the conventional SOM is replaced by a function module of SOM. Therefore, SOM^2 can be regarded as a variation of a modular network SOM (mnSOM). Each child SOM of SOM^2 is trained to represent a distribution of a data class, while the parent SOM generates a self-organizing map of the group of distributions modeled by the child SOMs. This extension of SOM is easily generalized to any combination of SOM families, including cases of neural gas (NG), in which, for example, "NG^2 as NG of NGs," and "NG-SOM as SOM of NGs" are possible. Furthermore, SOM^2 can be extended to the case of SOM^n, such as "SOM^3 as SOM of SOM^2." In this paper, the algorithms of SOM^2 and its variations are introduced and some simulation results are reported.*

Key words - mnSOM, modular network, manifold, neural gas

1 Introduction

The modular network SOM (mnSOM) proposed by Tokunaga *et al.* is a generalization of the conventional SOM, and it has enlarged the application field of self-organizing maps [1, 2]. The mnSOM has a modular network structure in which function modules of multi-layer perceptrons (MLPs) are arrayed on a lattice. Each MLP module of the mnSOM represents a "codebook function" instead of a codebook vector of the conventional SOM. Thus, the mnSOM is an SOM in function space rather than vector space. Further, because the mnSOM is good at dealing with functions, systems, and logics, etc., the mnSOM can call forth tasks such as time series classifications, control problems, and multisystem identifications [1, 2, 3].

These are what the mnSOM has given us. However, the mnSOM has broadened another new horizon. By adopting various kinds of neural networks or adaptive learning algorithms other than MLPs, one can easily create new types of SOM architectures. This means that the mnSOM allows one to design SOMs in the way most suitable for one's tasks. In fact, some extensions of SOM proposed in the past can actually be re-described as variations of the mnSOM. For example, the operator map is described as an mnSOM with linear transformation modules [4], whereas the PCA-module-mnSOM is equal to an adaptive subspace SOM (ASSOM) [5]. If one employs Hebbian neurons as the function modules, then the mnSOM becomes a conventional SOM.

Above all, however, the most curious variations are those in which the function modules are the SOMs themselves, i.e., SOM-module-mnSOM. In such a case, the mnSOM becomes an assembly of basic SOMs arrayed on a lattice (Fig. 1). Thus, this kind of mnSOM comprised of SOM modules

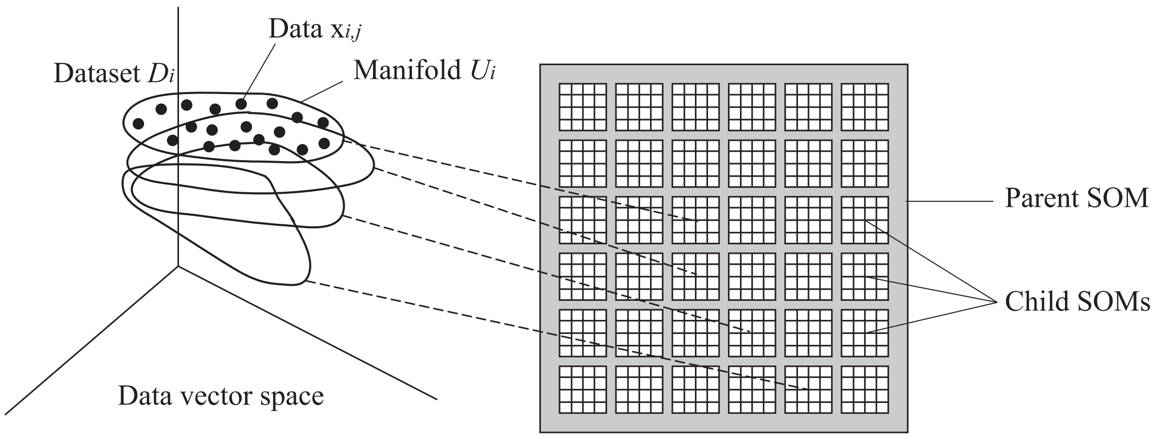


Figure 1: The scheme and the architecture of the SOM^2

can be regarded as an "SOM of SOMs," abbreviated as "SOM^2" in this paper. Further, because each child SOM module of an SOM^2 is trained to approximate a distribution of a class dataset, the entire SOM^2 represents a set of distributions of data classes. The method therefore involves the generation of a feature map of a group of class distributions. More precisely, each class is assigned to a point in the parent map, the position of which signifies to what degree the distribution of the class is similar or different to others. This characteristic of SOM^2 is quite different from a conventional SOM, in which each data vector corresponds to a point on the map and a class distribution is represented as a region where the vectors of the class are mapped.

Such an expansion of SOM would be useful in cases when differences between class distributions were considered more essential than differences between data vectors. As an example, let us consider a classification task of 3D objects from sets of photographs. In this case, each class consists of a set of 2D images of an object taken from various viewpoints. Even if some photographs of different objects may look similar, the entire distributions should be different. In other words, each object corresponds to a unique distribution of data vectors of 2D images. Therefore, it is necessary to generate a map of classes rather than a map of data vectors. In such a way, similar cases could be identified when one measured a set of sample data from each system, object, or parameter set, etc.

Theoretically, an SOM forms a bounded manifold in the high-dimensional data space as an approximation of the data distribution. Thus, the chief task of SOM^2 is regarded as the generation of a feature map of a set of manifolds. It is easy to generalize SOM^2 to cases of SOM^n such as "SOM^3 as the SOM of SOM^{2}s." In addition, other types of neural maps, e.g., those for neural gas (NG) [7], can be employed as replacements for the parent and/or child SOMs. For example, "NG of NGs" (NG-module-mnNG), "SOM of NGs" (NG-module-mnSOM), and "NG of SOMs" (SOM-module-mnNG), which we will abbreviate here as "NG^2," "NG-SOM," and "SOM-NG," are all possible as constituents of the SOM^n family.

In this paper, the architectures and the algorithms of the SOM^n family are introduced, and some simulation results are reported.

2 Theory of the SOM^n Family

2.1 Architecture and Algorithm of SOM^2

Let us first consider the dataset dealt with by the SOM^2. Because the goal is to map a group of class distributions, all data vectors are assumed to be classified and/or labeled in advance. There is also the issue of how to deal with non-labeled datasets, but that is not the focus of this paper. Now suppose that there are I classes, each of which has J sample data. Thus, let $D_i = \{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,J}\}$ $(i = 1, \ldots, I)$ denote the dataset of the ith class. In addition, it is assumed that the distribution of D_i is approximated by the manifold U_i, the dimension of which is equal to that of the child SOMs.

Let SOM^2 have K child SOMs $\{M^1, \ldots, M^K\}$, each of which has L codebook vectors $W^k = \{\mathbf{w}^{k,1} \ldots, \mathbf{w}^{k,L}\}$. (Here, let the superscripts represent the indexes of SOM^2, while the subscripts represent the indexes of class or data.) The tasks of the SOM^2 are (i) to represent the manifold set $\{U_i\}$ by using the child SOMs and (ii) to generate a feature map of the manifold set. Tasks (i) and (ii) are processed in parallel.

As in the case of the basic SOM, the algorithm of SOM^2 consists of three processes: the *competitive process*, the *cooperative process*, and the *adaptive process*. Before describing the details of the algorithm, let us first give a brief overview. (i) In the *competitive process*, the child SOM that best represents the distribution of D_i is chosen as the *best matching map* (BMM) of the ith class. This process is then repeated for each class. (ii) In the *cooperative process*, a set of learning rates are calculated by using the neighborhood function. (iii) In the *adaptive process*, all codebook vectors are innovated such that each child SOM represents the weighted interior division of the manifolds $\{U_i\}$, the weights of which are given by the learning rates. These processes are iterated until the calculation is converged.

Considering the above points, the algorithm of the SOM^2 is formulated as follows. In the competitive process, the distance between each class (D_i) and each child SOM (M^k) is first evaluated. Now let $L^2(D_i, M^k)$ denote the square distance between D_i and M^k given by the sum of the square quantization errors.

$$L^2(D_i, M^k) = E_i^k = \frac{1}{J}\sum_{j=1}^{J} e_{i,j}^{k*} \tag{1}$$

$$e_{i,j}^{k*} = \left\| \mathbf{w}_{i,j}^{k*} - \mathbf{x}_{i,j} \right\|^2 \tag{2}$$

Here, E_i^k denotes the average quantization error, whereas $e_{i,j}^{k*}$ and $\mathbf{w}_{i,j}^{k*}$ represent the square quantization error and the codebook vector of the *best matching unit* (BMU) within M^k. After evaluating the distances between D_i and every child SOM, the least distance map M_i^* is chosen as the BMM of D_i. Thus the index k_i^* of the BMM M_i^* is given by

$$k_i^* = \arg\min_k L^2(D_i, M^k) = \arg\min_k E_i^k \tag{3}$$

In the cooperative process, a set of learning rates $\{\Phi_i^k\}$ is calculated as follows.

$$\Phi_i^k = \frac{h_p[d(k, k_i^*), T]}{\sum_{i'=1}^{I} h_p[d(k, k_{i'}^*), T]} \tag{4}$$

Here, $h_p[\cdot, \cdot]$ is the neighborhood function which shrinks with the calculation time T, whereas $d(\cdot, \cdot)$ refers to the distance between two nodes in the map space.

In the adaptive process, each child SOM is innovated so as to be the weighted interior division of the data distributions. More precisely, M^k is expected to represent the interior division of the manifolds $\{U_1, \ldots, U_I\}$ with the weights $\{\Phi_1^k, \ldots, \Phi_I^k\}$. However, there is no direct way to calculate the codebook vectors from the distributions $\{D_i\}$ as their interior divisions. To solve this problem, let us introduce an extra set of basic SOMs $\{\hat{U}_1, \ldots, \hat{U}_I\}$ which tentatively approximates the manifold set $\{U_i\}$. Thus, the codebook vectors $\{\mathbf{w}^{k,l}\}$ are assumed to be innovated by the interior divisions of the sets of codebook vectors $\{\hat{U}_i\}$ as follows.

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \mathbf{v}_i^l \tag{5}$$

Here, $\mathbf{v}_i^l$ represents the lth codebook vector of $\hat{U}_i$. In addition, let us assume that $\hat{U}_i$ is estimated from the ith BMM M_i^*. Therefore, $\{V_i\}$ are calculated by the following equations.

$$\phi_{i,j}^l = \frac{h_c[d(l, l_{i,j}^{**}), T]}{\sum_{j'=1}^{J} h_c[d(l, l_{i,j'}^{**}), T]} \tag{6}$$

$$\mathbf{v}_i^l = \sum_{j=1}^{J} \phi_{i,j}^l \mathbf{x}_{i,j} \tag{7}$$

Here $h_c[\cdot, \cdot]$ is the neighborhood function of the child SOMs, and $l_{i,j}^{**}$ represents the index of the BMU of $\mathbf{x}_{i,j}$ within the BMM of D_i. By combining (5) and (7) together, the adaptive process is formulated as follows.

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \left\{ \sum_{j=1}^{J} \phi_{i,j}^l \mathbf{x}_{i,j} \right\} = \sum_{i=1}^{I} \sum_{j=1}^{J} \Phi_i^k \, \phi_{i,j}^l \mathbf{x}_{i,j} \tag{8}$$

Please note that the extra set of SOMs $\{\hat{U}_i\}$ has disappeared in (8). This is because $\{\hat{U}_i\}$ is defined for the convenience of explanation and is therefore not necessary in the practical calculation. These three processes are iterated until all codebook vectors are converged.

The task of the SOM^2 is interpreted as the estimation problem of the probability density function (pdf) varied by hidden parameters. By accepting this interpretation, each child SOM thereby represents the pdf at each hidden parameter state. Thus, the algorithm of SOM^2 described above is regarded as a modified EM algorithm, in which $\{M^k\}$ and $\{\hat{U}_i\}$ are estimated reciprocally by each other.

2.2 Generalization from SOM^2 to SOM^n

Equation (8) has a recursive structure similar to that of a Russian doll, in which the adaptation algorithm of the basic SOM is nested into itself. Therefore it is easy to generalize to cases of SOM^n, such as SOM^3 as the SOM of SOM^2, through further nesting.

Now let us consider the case of SOM^3. Let $\mathbf{x}_{i,j,k}$ denote the kth data vector of the jth sub-class of the ith class, while $\mathbf{w}^{l,m,n}$ denotes the nth codebook vector (the 3rd level) of the mth grandchild SOM (the 2nd level) of the lth child SOM (the 1st level). Then the codebook vector $\mathbf{w}^{l,m,n}$ is innovated by the following equation.

$$\mathbf{w}^{l,m,n} = \sum_{i=1}^{I} \alpha_i^l \left\{ \sum_{j=1}^{J} \beta_{i,j}^m \left\{ \sum_{k=1}^{K} \gamma_{i,j,k}^n \mathbf{x}_{i,j,k} \right\} \right\} \tag{9}$$

$$= \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} \alpha_i^l \beta_{i,j}^m \gamma_{i,j,k}^n \mathbf{x}_{i,j,k} \tag{10}$$

Here, α, β and γ are the learning rates of each level given by

$$\alpha_i^l = \frac{h_1[d(l, l_i^*), T]}{\sum_{i'=1}^{I} h_1[d(l, l_{i'}^*), T]} \tag{11}$$

$$\beta_{i,j}^m = \frac{h_2[d(m, m_{i,j}^{**}), T]}{\sum_{j'=1}^{J} h_2[d(m, m_{i,j'}^{**}), T]} \tag{12}$$

$$\gamma_{i,j,k}^n = \frac{h_3[d(n, n_{i,j,k}^{***}), T]}{\sum_{k'=1}^{K} h_3[d(n, n_{i,j,k'}^{***}), T]} \tag{13}$$

Here, h_1, h_2, h_3 represent the neighborhood functions of corresponding levels, while $l_i^*, m_{i,j}^{**}, n_{i,j,k}^{***}$ each denote the index of the best matching unit/module of the corresponding level. For the case of SOM^n, all one has to do is to repeat the above equations n times.

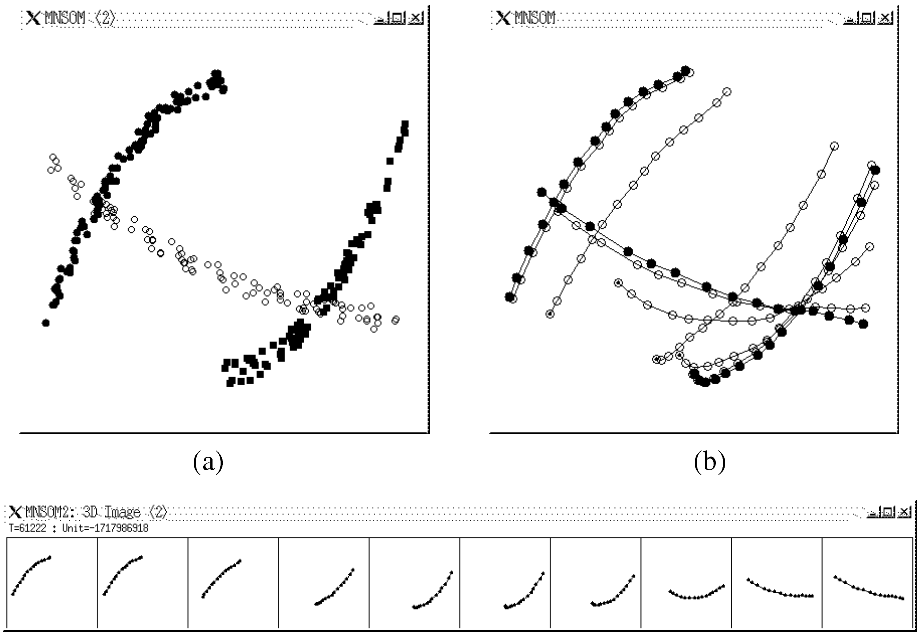
2.3 Variations of SOM^2 with the Neural Gas Algorithm

By adopting the algorithm of neural gas (NG), several variations of SOM^2 can be created. First let us consider the case of "NG of NGs," i.e., NG^2. In this case, the cooperative process of SOM^2 is replaced by the NG algorithm. Thus, equation (4) is replaced by

$$\Psi_i^k = \frac{\exp[-s(M^k, D_i)/\lambda_p(T)]}{\sum_{i'=1}^{I} \exp[-s(M^k, D_i')/\lambda_p(T)]} \tag{14}$$

Here, $s(M^k, D_i)$ gives the order of the kth child NG for the ith class, while $\lambda_p(T)$ gives the rate of decay. The ith manifold is approximated by the 1st order child NG, i.e., the best matching neural gas. Now let $\mathbf{w}_i^{*,l}$ be the lth codebook vector of the best matching neural gas of the ith class. The learning rate $\psi_{i,j}^l$ is described as follows.

$$\psi_{i,j}^l = \frac{\exp[-s(\mathbf{w}_i^{*,l}, \mathbf{x}_{i,j})/\lambda_c(T)]}{\sum_{j'=1}^{J} \exp[-s(\mathbf{w}_i^{*,l}, \mathbf{x}_{i,j})/\lambda_c(T)]} \tag{15}$$



(c)

Figure 2: The result of mapping artificial datasets by using SOM^2. (a) The distributions of the 3 datasets. (b) The set of maps generated by the 10 child SOMs. (c) The parent map of the child SOMs

Equation (6) is replaced by (15). Therefore, by combining (14) and (15) together, the adaptation process of NG^2 is formulated as follows.

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Psi_i^k \left\{ \sum_{j=1}^{J} \psi_{i,j}^l \mathbf{x}_{i,j} \right\} = \sum_{i=1}^{I} \sum_{j=1}^{J} \Psi_i^k \psi_{i,j}^l \mathbf{x}_{i,j} \quad (16)$$

If one needs the "SOM of NGs" (NG-SOM), then the answer can be obtained by combining (4) and (15) together as

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \left\{ \sum_{j=1}^{J} \psi_{i,j}^l \mathbf{x}_{i,j} \right\} = \sum_{i=1}^{I} \sum_{j=1}^{J} \Phi_i^k \psi_{i,j}^l \mathbf{x}_{i,j} \quad (17)$$

whereas the combination of (6) and (14) becomes the "NG of SOMs" (SOM-NG). In addition, if one employs other types of mapping algorithms, then the number of variations will increase further. Of these variations, one of the most promising architectures may be the NG-SOM, i.e., the "SOM of NGs." This is because the child NGs have no restrictions on the dimensions of the target manifolds, whereas the parent SOM allows one to visualize the relationships of the classes. The NG-SOM therefore inherits the advantages of both algorithms.

3 Simulations and Results

To validate the abilities of a SOM^2 family, some simulation tasks were employed. The first task was to generate a feature map of artificial manifolds, as shown in Fig. 2(a). The number of classes was 3, each of which consisted of 100 data vectors, and the SOM^2 had 10 child SOMs consisting of 15

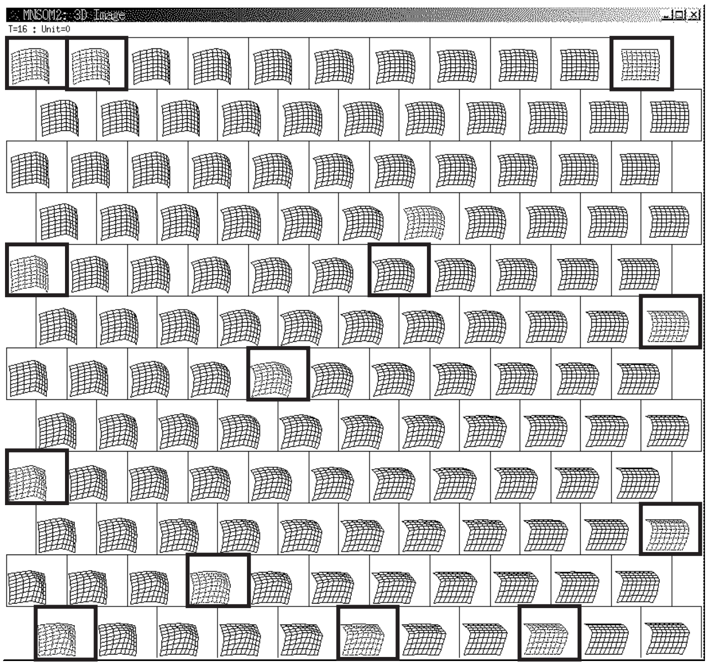


Figure 3: The map of 3D objects from 2D images by using the SOM^2

codebook vectors. Both the parent and child SOMs had one-dimensional array structures. Fig. 2(b) and (c) represent the child and parent maps respectively. The SOM^2 generated a continuous map of the manifolds in which the child maps showed change gradually. Furthermore, the SOM^2 interpolated "intermediate manifolds" between the given data classes.

The second task was to generate a map of 3D objects from 2D projected images. Because a set of 2D images of an object taken from various viewpoints forms a 2-dimensional manifold, the distribution of the image data can be modeled by a basic SOM. Therefore SOM^2 would be the desired architecture for the task. Please notice that the SOM^2 does not know how 3D objects can be reconstructed from their 2D images. Fig. 3 is the map of the 3D objects generated by the parent SOM. Each box represents the corresponding child SOM, in which the 3D objects acquired by the child SOMs are depicted. The thick boxes in the figure represent the BMM of the training data classes. The parent map was generated successfully, showing a good continuity of varying 3D shapes. Furthermore, the SOM^2 interpolated unknown "intermediate 3D objects" between the BMMs.

One of the possible applications of such 3D object mapping is face recognition. Fig. 4 shows our tentative results of the mapping of facial images using an NG-SOM. Similar to the second task, each class consisted of a set of photographs taken from various viewpoints. Results showed that each NG module learned the set of facial images of a person. Though the SOM^2 also showed similar results (not shown), the NG-SOM is more suited for this type of task because the NG-SOM can adapt not only to differences in viewpoints but also to differences in facial expressions and hairstyles, etc.

4 Conclusion

In this paper we have proposed an extension of SOMs called SOM^2 and its variations. SOM^2 provides a method for the topological mapping of a set of data distributions. Simulation results suggest that

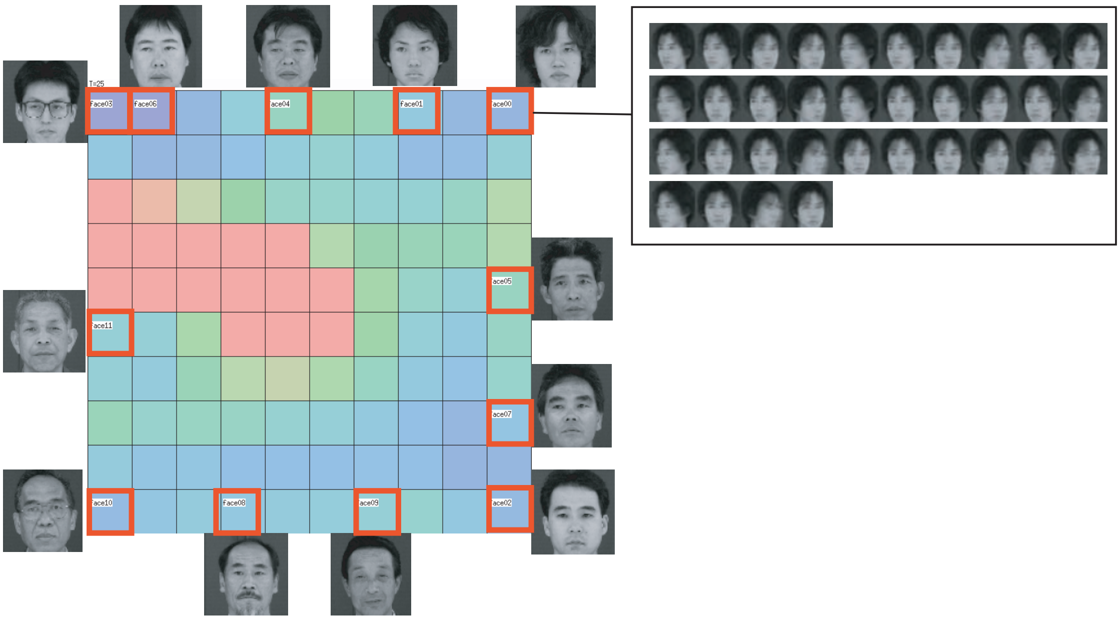


Figure 4: Map of facial images using NG-SOM

the SOM^2 family will be a powerful tool for class visualization and analysis. The validation of such abilities in actual applications, including facial image recognition tasks, is currently in progress.

Acknowledgements

This work was supported by a COE program (Center #J19) grant to the Kyushu Institute of Technology by the Japanese MEXT. The facial data in this paper are used by permission of Softopia Japan, Research and Development Division, HOIP Laboratory.

References

[1] Tokunaga, K., Furukawa, T., Yasui, S. (2003) Modular network SOM: Extension of SOM to the realm of function space. Proc. of WSOM2003, 173–178

[2] Furukawa, T., Tokunaga, K., Morishita, K., Yasui, S. (2005) Modular network SOM (mnSOM): From vector space to function space. Proc. of IJCNN2005 (*accepted*)

[3] Furukawa, T., Tokunaga, K., Kaneko, S., Kimotsuki, K., Yasui, S. (2004) Generalized self-organizing maps (mnSOM) for dealing with dynamical systems. Proc. of NOLTA2004, 231–234

[4] Kohonen, T. (1993) Generalization of the Self-organizing map. Proc. of IJCNN93, 457–462

[5] Kohonen, T., Kaski, S., Lappalainen, H. (1997) Self-organized formation of various invariant-feature filters in the adaptive-subspace SOM. Neural Computation **9**, 1321–1344

[6] Kohonen, T. (2001) Self-Organizing Maps, 3.ed., Springer

[7] Martinetz, T. M., Berkovich, S.G., Schulten, K. J. (1993) Neural-gas network for vector quantization and its application to time-series prediction. IEEE Transactions on Neural Networks **4**, 558-569