SOM of SOMs: Self-Organizing Map Which Maps a Group of Self-Organizing Maps

Tetsuo Furukawa

Kyushu Institute of Technology, Kitakyushu 808-0196, Japan,
furukawa@brain.kyutech.ac.jp,
WWW home page: http://www.brain.kyutech.ac.jp/~furukawa

Abstract. This paper aims to propose an extension of SOMs called an "SOM of SOMs," or SOM^2, in which the mapped objects are self-organizing maps themselves. 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). Since each child SOM module in SOM^2 is trained to represent a manifold, the parent SOM in SOM^2 generates a self-organizing map representing the distribution of the group of manifolds modeled by the child SOMs. This extension of SOM is easily generalized in the case of SOM^n, such that "SOM^3 as SOM of SOM^{2}s." In this paper, the algorithm of SOM^2 is introduced, and some simulation results are reported.

1 Introduction

The self-organizing map (SOM) as introduced by Kohonen has provided us with a powerful tool for data mining, classification, analysis, visualization, etc.[1]. In the case of labeled training samples, SOM should be one of the best techniques available for visualizing the distribution of each class. In such a case, an SOM shows how the sample vectors of each class are distributed in the high dimensional data space by transforming them to low dimensional map space while preserving their topological relationships. This means that, if the map is generated appropriately, a distance between two points in the map space signifies either similarity or difference between the corresponding two vectors in the data space. Though this characteristic of SOMs is effective in many applications, some cases require the visualization of such relationship *between the distributions of classes*, i.e., to what degree two class distributions are similar or different. Unfortunately, however, an SOM does not provide such information. An SOM provides a map of data vectors, but it is not a map of class distributions.

The aim of this paper is to propose a method of mapping classes which can represent the relationships between their distributions. In other words, the mapped objects of an SOM are no longer vectors but class distributions that form manifolds in the data space. Since the distribution of each class, i.e., each manifold, can be represented by a basic SOM, all the classes can be modeled by a group of basic SOMs. Thus, the method involves the generation of a self-organizing map of a group of self-organizing maps, that is, an "SOM of SOMs."

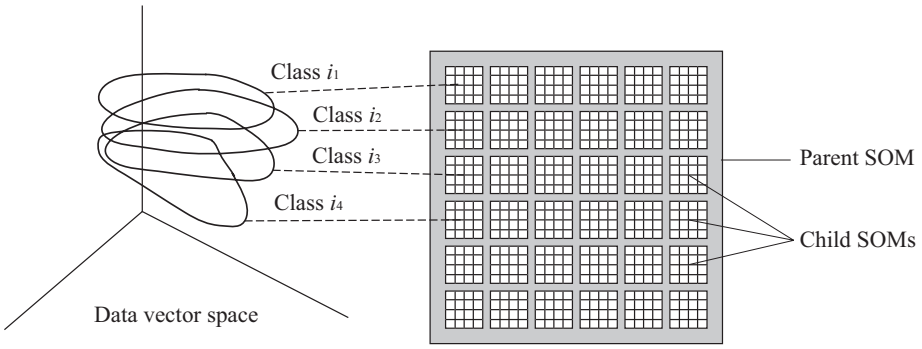


Fig. 1. The architecture and the scheme of SOM^2 as an "SOM of SOMs"

Previously, we have proposed a generalization of an SOM called a "modular network SOM" (mnSOM), in which each nodal vector unit is replaced by a function module of a neural network[2, 3]. Such a generalization was first proposed by Kohonen as an "Operator Map"[4]. However, even further generalization is possible by adopting the idea of a modular network because various types of trainable architectures, including the case of the SOM itself, can be chosen as modules. Therefore, an "SOM of SOMs" can be realized by SOM-module-mnSOM as an example of a generalized SOM.

2 Architecture and Algorithm of SOM^2

The architecture of an "SOM of SOMs" is simple — just a replacement of vector units by the basic SOMs, as is shown in Fig.1. In this paper, let us abbreviate this architecture as "SOM^2." Each child SOM learns to represent a manifold in the data space, whereas the parent SOM is expected to generate a map of the manifolds represented by the child SOMs. The learning processes of both levels of SOMs progress 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*.

Now suppose that there are I classes, each of which has J sample data. Further, let D_i be the ith dataset, which consists of $\{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,J}\}$, and let us assume that SOM^2 consists of K child SOMs, each of which has L codebook vectors $W^k = \{\mathbf{w}^{k,1}, \ldots \mathbf{w}^{k,L}\}$. (Here, the superscripts represent the indexes of SOM^2, while the subscripts represent the indexes of classes or data vectors.)

In the *competitive process*, each class dataset is first picked up one by one. Then the child SOM that minimizes the total quantization error is chosen as "the best matching map (BMM)" (i.e., the winning child SOM) of the class. The BMM of the ith class k_i^* is defined as follows.

$$k_i^* = \arg\min_k E_i^k = \arg\min_k \sum_{j=1}^{J} e_{i,j}^{k*} \tag{1}$$

Here, E_i^k denotes the total quantization error of the kth child SOM for the ith class, which is the sum of the square error $e_{i,j}^{k*}$ between $\mathbf{x}_{i,j}$ and the best matching unit (BMU)

of the kth child SOM. Thus,

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

$$l_{i,j}^{k*} = \arg\min_{l} \left\| \mathbf{w}^{k,l} - \mathbf{x}_{i,j} \right\|^2 \tag{3}$$

Here, $l_{i,j}^{k*}$ is the index of BMU of the kth child SOM for the data vector $\mathbf{x}_{i,j}$. This process is repeated for all the classes.

In the *cooperative process*, the learning rates $\{\Phi_i^k\}$ and $\{\phi_{i,j}^l\}$ are calculated by using the neighborhood functions as follows.

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

$$\phi_{i,j}^l = \frac{h[d(l, l_{i,j}^{**}), T]}{\sum_{j'=1}^{J} h[d(l, l_{i,j'}^{**}), T]} \tag{5}$$

Here, $g[\cdot,\cdot]$ and $h[\cdot,\cdot]$ are the neighborhood functions of the parent and the child SOMs, respectively, which shrink with the calculation time T. $d(\cdot,\cdot)$ refers to the distance between two nodes in the map space, while $l_{i,j}^{**}$ denotes the index of the BMU of the BMM, i.e., $l_{i,j}^{**} = l_{i,j}^{k_i^**}$.

In the *adaptive process*, all the codebook vectors of the entire child SOMs are innovated by using the learning rates Φ_i^k and $\phi_{i,j}^l$. Now suppose that $V_i = \{\mathbf{v}_i^1, \ldots, \mathbf{v}_i^L\}$ is a set of codebook vectors only for the ith class. In other words, V_i represents the tentative estimation of the ith manifold. By adopting the algorithm of the batch learning SOM, $\{\mathbf{v}_i^l\}$ are defined as follows.

$$\mathbf{v}_i^l = \sum_{j=1}^{J} \phi_{i,j}^l \mathbf{x}_{i,j} \tag{6}$$

Each child SOM is innovated so as to be the weighted interpolation of the estimated manifolds, i.e., the interpolation of $\{V_i\}$ with the learning rates Φ_i^k. Thus,

$$\mathbf{w}^{k,l} = \sum_{i=1}^{I} \Phi_i^k \mathbf{v}_i^l \tag{7}$$

By combining (6) and (7) together, the adaptive process is formulated as follows.

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

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

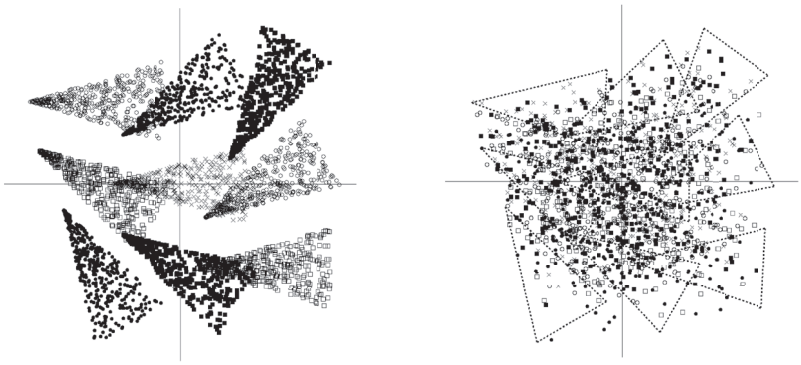
SOM^2 is expected to work in the following way. If the number of classes is larger than the number of child SOMs, i.e., $I > K$, then each child SOM is expected to be assigned to one or more classes as their BMM. In such a case, each child SOM learns the data vectors of the assigned classes in such a way that they can represent their average distribution. On the other hand, in the case of $I < K$, I out of K child SOMs become BMMs, each of which learns the distribution of the corresponding class, whereas the rest of $(K - I)$ child SOMs, which are not BMMs, are trained to represent the intermediate distributions by interpolation of the given classes. Since the number of class depends on the task, SOM^2 is expected to work well in both cases. These situations were reproduced in this study in the first simulation using artificial manifolds (Fig. 2 and Fig. 3).

3 Simulations and Results

3.1 A Case of Artificial Datasets

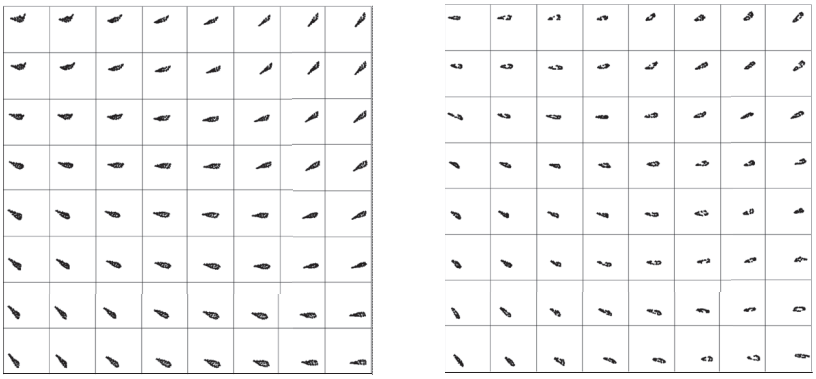
Two sets of artificial manifolds shown in Fig. 2 were used to validate the ability of SOM^2. In the first set (Fig. 2a), there was a small number of classes ($I = 9$), while each class had a large number of data vectors ($J = 400$) that were sampled randomly. The shapes of the manifolds were all congruent triangles, the positions and orientations of which were changed gradually. On the other hand, the second set (Fig. 2b) had a large number of classes ($I = 400$), each of which consisted of a very small number of samples ($J = 4$). The shapes of the manifolds were all congruent triangles just as in the first case, some of which are shown in Fig. 2b with dashed triangles. Unlike the first case, however, it is difficult to recognize the shapes of the manifolds due to the deficiency of the samples. Furthermore, the manifolds of the second set overlapped with each other in such a way that the sample vectors were distributed evenly over the area without forming clusters. In addition, the positions and orientations of the manifolds were changed in the same manner as in the first case. The number of the child SOMs was fixed in both cases ($K = 8 \times 8$), while the number of codebook vectors was 6×6 for each child SOM. Thus, $I < K$ in the first case, and $I > K$ in the second case.

Fig. 3 shows the simulation results. In the figure, the dots plotted in each box indicate the map generated by the corresponding child SOM. In the case of the first manifold set (Fig. 3a, corresponding to Fig. 2a), every child SOM represented the triangle shape well. Furthermore, the positions and orientations varied gradually in such a way that a continuous map of the manifolds was formed successfully in the parent SOM. This result also shows that the unknown intermediate manifolds were appropriately estimated by interpolation. The result was almost the same in the case of the second manifold set (Fig. 3b, corresponding to Fig. 2b). The parent map was well organized with good continuity, i.e., the positions and orientations of the child maps varied continuously. SOM^2 also succeeded in estimating the distributions, even though the number of samples per class was very small. Though not shown in the figure, the child maps were aligned so that the codebook vector of each child SOM with the same index corresponded to a congruent point of each manifold. For example, the BMUs which corresponded to the apex of the triangles had the same index in the child SOMs. Therefore it is possible to



(a) (b)

Fig. 2. The two sets of artificial manifolds used in the first simulation



(a) (b)

Fig. 3. The maps of the manifolds generated by SOM^2 from the dataset of Fig. 2

observe how the manifold gradually varied its shape by tracing the codebook vectors with the same index.

3.2 Classification of 3D objects from 2D images by SOM^2

The task of the second simulation was to generate a map of 3D objects from 2D projected images. Put more clearly, the task was to make a self-organizing map of 3D objects from a set of photo albums, each of which contains several photographs of one object from various viewpoints. Since a set of 2D images of a 3D object projected from different viewpoints form a manifold in the high dimensional data space, the distribution of the 2D images can be modeled by a basic SOM. Therefore an assembly of basic SOMs such as SOM^2 would be expected to represent a group of 3D objects. Please notice that SOM^2 does not know how 3D objects can be reconstructed from their 2D images.

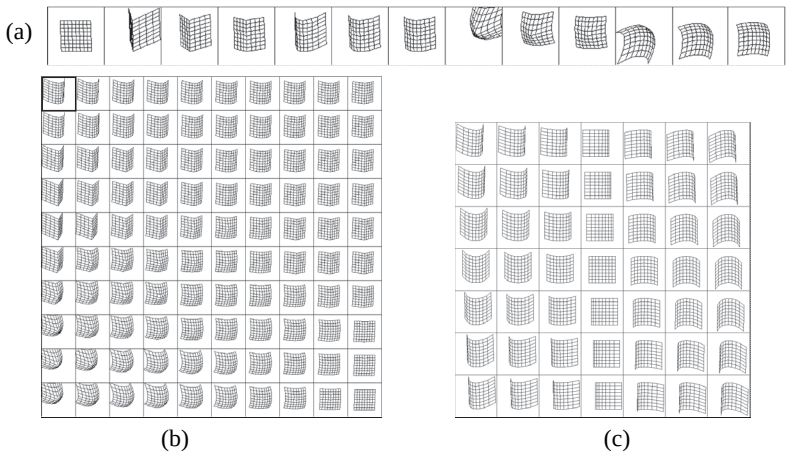


Fig. 4. A map of 3D objects generated by SOM^2 from 2D projected images

The 13 objects shown in Fig. 4a were used in the simulation, and 49 2D images from various viewpoints were prepared for each object. The objects are assumed to be flexible grids, and each data vecotor consisted of a set of coordinates of the lattice points on a 2D image. Fig. 4b is the map of the 3D objects generated by the parent SOM, while a map of a child SOM (corresponding to the thick box in Fig. 4b) is presented in Fig. 4c. The parent map was generated successfully, showing a good continuity of varying 3D shapes. This result also means that SOM^2 created 2D images of unknown intermediate 3D objects by interpolating between the given objects. Furthermore, all child SOMs were aligned with each other in such a way that all codebook vectors with the same index were assigned to the images taken from the same viewpoint.

4 Conclusion

In this paper we have proposed an extension of SOMs called SOM^2. SOM^2 provides a method for the topological mapping of classes by comparing their distributions instead of comparing individual vectors. The simulation results suggest that SOM^2 will be a powerful tool for class visualization and analysis. Further extensions of SOM^2 are currently being developed for datasets without class labels.

This work was supported by a COE program (center #J19) granted by MEXT of Japan.

References

1. Kohonen, T.: Self-Organizing Maps, 3.ed., Springer (2001)
2. Tokunaga, K., Furukawa, T., Yasui, S.: Modular network SOM: Extension of SOM to the realm of function space. Proc. of WSOM2003 (2003) 173–178
3. Furukawa T., Tokunaga K., Kaneko S., Kimotsuki K., Yasui, S.: Generalized self-organizing maps (mnSOM) for dealing with dynamical systems. Proc. of NOLTA2004 (2004) 231–234
4. Kohonen, T.: Generalization of the Self-organizing map. Proc. of IJCNN93 (1993) 457–462