Self-Organization of Feature Columns and its application to Object Classification

Satoshi Suzuki and Naonori Ueda

NTT Communication Science Labs.
Kyoto, Japan

Abstract

We propose a computational model for the self-organization of feature columns based on a modular framework. The proposed model consists of several modules; each module is composed of a collection of Gaussian units. The combination of hierarchical competition within and between modules and a smoothness constraint finds out continuity among input patterns, and topographically maps these series of patterns into different modules. Computer simulations show an example of self-organization and object classification by combining the created feature columns.

1. INTRODUCTION

Recent experimental studies have shown some new aspects of high-level visual cortices. Fujita et al. (1992) found that the inferotemporal cortex (IT) is composed of columnar modules in which cells with overlapping but slightly different selectivities to complex patterns cluster together. This result suggests that we, human beings, may use such feature columns for object recognition. Wang et al. (1996) showed in an experiment that involved the optical recording of a monkey's IT using a series of dame doll faces, that an optical spot moves gradually as the face turns. This indicates the topological mapping of object views to the position of the cells in the cortex.

On the other hand, using a computational approach, several self-organization models were proposed to represent maps of the early visual cortex (Kohonen, 1982; Tanaka, 1990). However, no apparent relation to higher-level tasks such as pattern recognition has been shown. In pattern recognition tasks, the feature columns can be regarded as a set of exclusive columns each of which responds only to input patterns belonging to the same class. Although Weinshall et al. (1990) proposed a self-organization model for object classification, their model requires separate training for each object class.

We propose a computational model for forming feature columns for pattern recognition tasks in a self-organizing manner. The important point to note is that our model, unlike Weinshall's model, can form feature columns without any pattern class information. The proposed model is based on a modular structure (Jordan & Jacobs, 1992). That is, given input patterns for some classes, the same number of modules are trained so that only one module responds to the patterns of the same class. The training is based on the competitive learning scheme previously proposed by one of the authors (Suzuki & Ando, 1995; Fujita et al., 1996).

2. THE MODEL

We detail the proposed network model in this section. The proposed network can find out the continuity among various input patterns and can classify the input patterns belonging to the same continuity into a category. During the learning, the network acquires several representative patterns in each module corresponding to a category.

2.1. The Architecture

The network model, as shown in Fig. 1, consists of three layers: the input layer, the competition layer and the representation layer. The representation layer consists of M modules each of which is composed of U Gaussian units. Suppose that an input pattern is represented by a vector $\mathbf{x}$, the input vector $\mathbf{x}$ is presented to each Gaussian unit of all M modules through the input layer. The output of the jth unit in the ith module for the input $\mathbf{x}$ is defined by

$$\mathbf{y}_{j|i} = \mathbf{c}_{j|i}\, h_{j|i}\ . \tag{1}$$

Here, $\mathbf{c}_{j|i}$ is a "center" parameter and $h_{j|i}$ is a softmax function of the squared error $\left\|\mathbf{x} - \mathbf{c}_{j|i}\right\|^2$ for the jth Gaussian unit

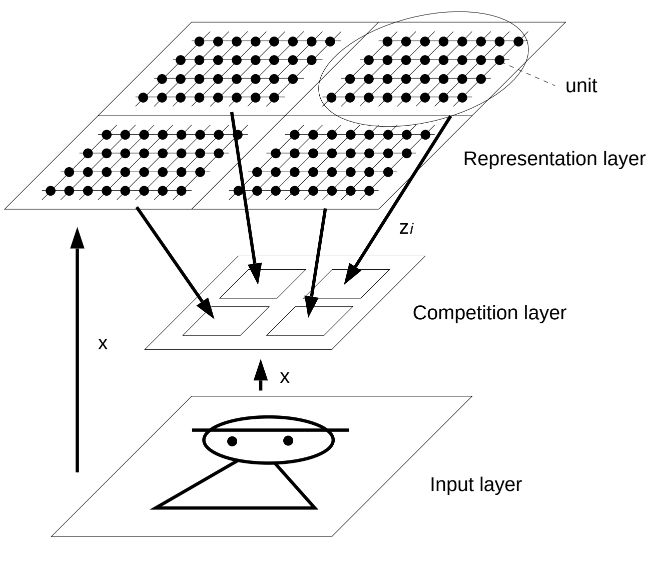

Figure 1: The network model

in the ith module. That is,

$$h_{j|i} = \frac{\exp\left[-\left\|\mathbf{x} - \mathbf{c}_{j|i}\right\|^2 \Big/ 2\sigma^2\right]}{\sum_{j=1}^{U} \exp\left[-\left\|\mathbf{x} - \mathbf{c}_{j|i}\right\|^2 \Big/ 2\sigma^2\right]} . \tag{2}$$

The output of the ith module is given by

$$\mathbf{z}_i = \sum_{j=1}^{U} \mathbf{y}_{j|i} . \tag{3}$$

Clearly, $\mathbf{y}_{j|i}$ is the nonlinear weighted average of $\mathbf{c}_{j|i}$, for $j = 1, \cdots, U$ and $\mathbf{z}_i$ represents a reconstructed pattern for $\mathbf{x}$ by ith module.

Once the reconstructed pattern for each module is obtained, the classification rule is given by

$$\text{Decide} \quad \text{Class } k \quad \text{if } \|\mathbf{x} - \mathbf{z}_k\| < \|\mathbf{x} - \mathbf{z}_l\|, \forall l \neq k.$$

2.2. Learning

During the learning, the network is given input patterns in random order. With such random input patterns, the smoothness constraints force successive representations to gather in a module; on the contrary, competition between units within a module strengthens the difference between the representations. This combination encourages similar but slightly different representations to gather in a module. On the other hand, competition between modules strengthens the differences of the representations among the modules. The representations in the ith module are shown by $\mathbf{c}_{j|i}$, for $j = 1, \cdots, U$.

To realize the above competitions, we maximize the following log likelihood function (Suzuki & Ando, 1995; Jordan et al., 1992):

$$\ln \sum_{i=1}^{M} g_i \sum_{j=1}^{U} h_{j|i} \frac{1}{\sigma} \exp\left[-\left\|\mathbf{x} - \mathbf{c}_{j|i}\right\|^2 \Big/ 2\sigma^2\right] - \delta \sum_{i=1}^{M} S_i, \tag{4}$$

where

$$g_i = \frac{\exp\left[-\|\mathbf{x} - \mathbf{z}_i\|^2 / 2\sigma^2\right]}{\sum_{i=1}^{M} \exp\left[-\|\mathbf{x} - \mathbf{z}_i\|^2 / 2\sigma^2\right]} , \tag{5}$$

and S_i denotes the smoothness constraint in the ith module and δ is its coefficient. S_i is computed by

$$S_i = \sum_{j=1}^{U} \left\| \frac{\mathbf{c}_{j-1|i} - \mathbf{c}_{j|i}}{\left\|\mathbf{c}_{j-1|i} - \mathbf{c}_{j|i}\right\|} - \frac{\mathbf{c}_{j|i} - \mathbf{c}_{j+1|i}}{\left\|\mathbf{c}_{j|i} - \mathbf{c}_{j+1|i}\right\|} \right\|^2 . \tag{6}$$

This smoothness constraint corresponds to computing the second derivative of $\mathbf{c}_{j|i}$ by using $\mathbf{c}_{j-1|i}$ and $\mathbf{c}_{j+1|i}$. That is, this constraint forces three successive representative vectors to be smooth as much as possible.

3. SIMULATIONS

3.1. Self-organization of Object representations

First, we show a simulation result in which representative images of three-dimensional objects are made to self-organize, and are topologically mapped into different modules.

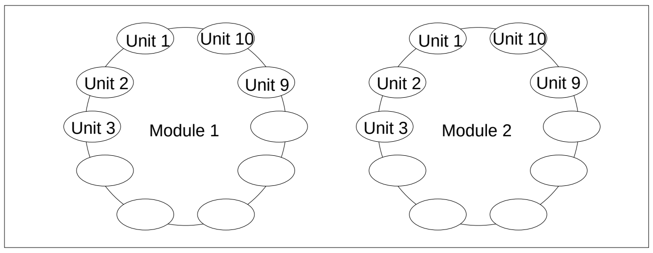

Figure 2: The representation layer used in the simulations

Learning We prepared two 3D objects, i.e., two kinds of dolls, for input and produced 360 different images from each of the objects, each image differing by one degree of rotation. Each image was compressed into 16×15 dimensional image with a Gaussian filter and then was converted into a vector as an input vector $\mathbf{x} \in R^{16\times15}$. These compressed images were employed as input to train the network.

The network was arranged into two modules to correspond to the number of object classes. Ten units were set on a circle in each module, where we assumed the first unit and the last unit to be adjacent (Fig. 2).

As the learning proceeded, we gradually decreased the standard deviation of the units to a threshold, to enhance the competition within the modules.

Results After training, we fixed the weights of the network and tested it with all images. Figure 3 plots g in Eq. (5) as a function of the view direction. Each value of g for an input image means the relative accuracy of image recovered by each module, and $g_1 > g_2$ and $g_1 < g_2$ indicate that the input image is classified as class 1 and class 2, respectively. Almost all winners of recovered images for input images of the same class were the same module, which means that most images of the same object were classified into the same class.

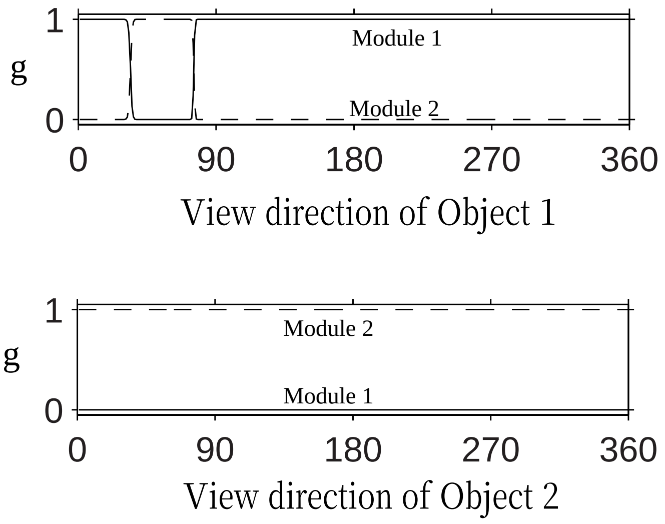

Figure 3: The results of classification

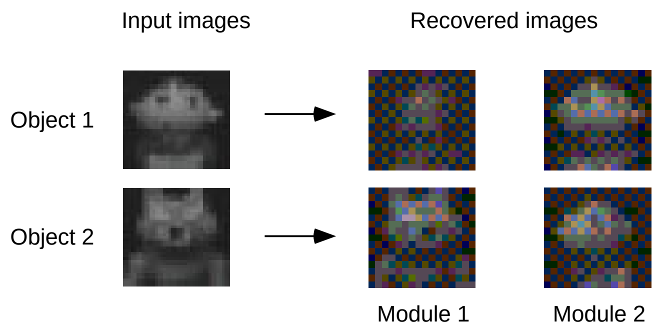

Figure 4: Input and recovered images

Figure 4 shows the input and recovered images by the modules. We can see that the input images of Object 1 and Object 2 were successfully recovered by module 2 and module 1, respectively.

Figure 5 shows images acquired in the units of each module. These images can be seen as images corresponding to representative view angles. The representations obtained in the modules differ in their view directions according to the positions of the units. These results suggest that the network has an ability of acquiring representations of 3D objects in different modules and topological mapping of the representations.

3.2. Extension to Hierarchical Model

We also implemented a hierarchical model by combining the network models described above. The self-organization of partial features and classification using the feature columns are shown through a simple simulation.

Learning Figure 6 shows the hierarchical network model composed by piling up the above-mentioned model: a higher network is put on four lower networks. Each network is given the same conditions as the simulation described above, that is, arranged into two modules and there are ten units on a circle in each module.

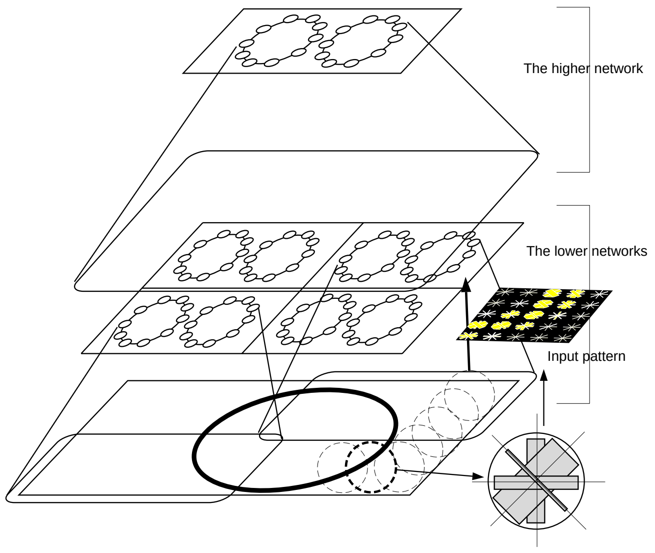

Figure 6: The hierarchical model

We prepared two line drawings: Circle and Rectangle. During the training, these drawings were sequentially presented to the network by changing their positions within a region. The region is divided into 2×2 subregions and each subregion corresponds to an input layer of the lower networks. Thus, we used four lower networks as shown in Fig. 6.

Each subregion is further divided into 5×5 areas which is shown as broken circles. For each area, 4 dimensional vector is produced according to the orientation of the drawing across the area: input vectors to the network are created by filtering the line across the area with four orientation filters. That is, four directions are represented by vertical, horizontal and two oblique directions bisecting them, and the strength of each direction is represented by its thickness. After all, a 5×5×4-dimensional vector is being presented to the lower network as one input pattern. On the other hand, the higher network gets the activities of all units in the lower networks, i.e., $\exp\left[-\left\|\mathbf{x}-\mathbf{c}_{j|i}\right\|^2/2\sigma^2\right]$, $\forall i, j$, as input vector.

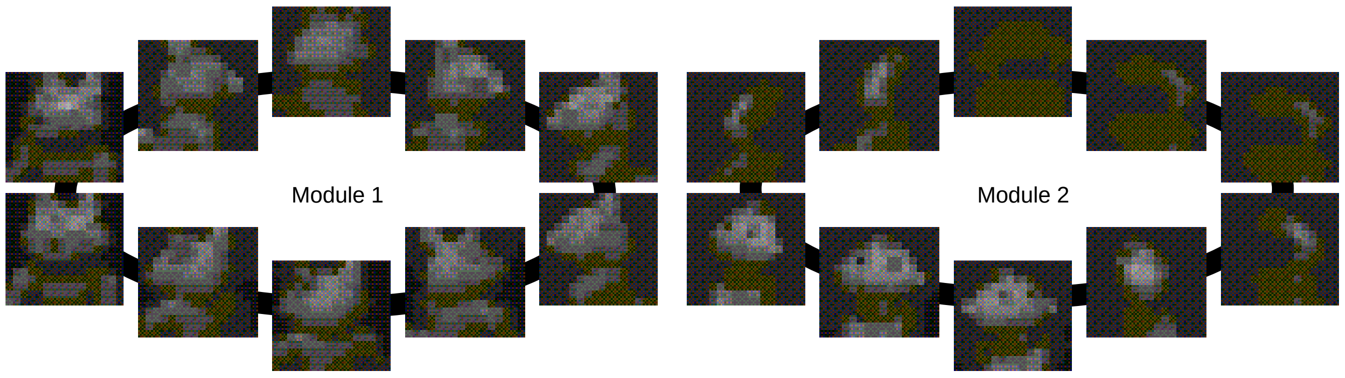

Figure 5: Representations obtained in modules

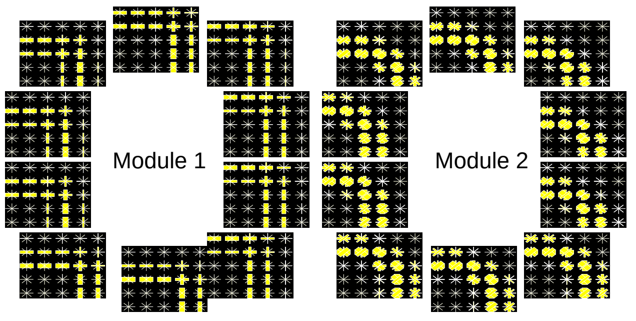

Figure 7: The representations obtained in a lower network

During training, we gradually decreased the standard deviation of the units like in the previous section. Each network was trained independently and the higher network was trained after the lower networks had been trained.

Results Figure 7 shows obtained representative patterns of all units for each module. Each representative pattern consists of 5×5 points each of which corresponds to one divided area of input image. The symbol at each point indicates four directions and their strength. Note that four kinds of symbols are superimposed at each point. As we expected, desired representative patterns for each module were obtained: parts of Rectangle and parts of Circle were gathered in module 1 and module 2, respectively.

Figure 8 shows values of g (Eq. (5)) over input region obtained in the higher network. One can see that $g_1 < g_2$ and $g_1 > g_2$ holds over the input region for Rectangle and Circle, respectively. This means that two line drawings were successfully classified.

Clearly, the hierarchical representation realizes more detailed feature columns that might get characteristic features for each pattern class.

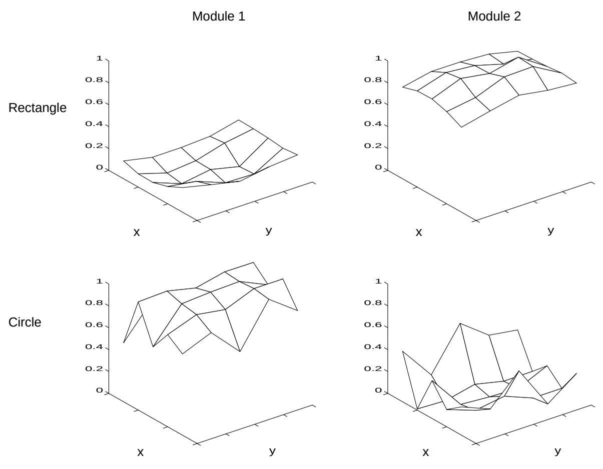

Figure 8: Classification results by hierarchical model

4. CONCLUSION

We proposed a network which can automatically form the feature columns for pattern recognition tasks. The within-module and between-modules competitions with a smoothness constraint enable the network to acquire several representative patterns for each class.

The simulation results with a single network have validated these abilities. On the other hand, a simulation with a hierarchical model showed that the networks can gather similar features even from partial images and can also classify objects by using these feature columns. The simulation that we showed was just a simple example. Nevertheless, this model may be hints on more general type of self-organization of feature columns and object recognition using feature columns.

References

[1] Fujita, I. Tanaka, K. et al. (1992). Columns for visual features of objects in monkey inferotemporal cortex. *Nature*, 360, 343-346.

[2] Fujita, T. Suzuki, S. & Ando, H. (1996). 3D Object Recognition by Coupling Mixtures of Autoencoders and Dynamic Matching. *ICONIP'96*, Hong Kong.

[3] Jordan, M. I. and Jacobs, R. A. (1992). Hierarchies of adaptive experts. In Moody, J. E., Hanson, S. J. & Lippmann, R. P., (eds), *Advances in Neural Information Processing Systems 4*. Morgan Kaufmann Publishers, San Mateo, CA. 985-992.

[4] Kohonen, T. (1988). Self-organization and associative memory. Springer, New York Berlin Heidelberg.

[5] Suzuki, S. and Ando, H. (1995). Unsupervised Classification of 3D Objects from 2D Views. *Advances in Neural Information Processing Systems 7*. The MIT Press, Cambridge, MA.

[6] Tanaka, S. (1990). Theory of Self-Organization of Cortical Maps: Mathematical Framework. *Neural Networks*, 3, 625-640.

[7] Wang, G., Tanaka, K., & Tanifuji, M. (1996). OPTICAL IMAGING OF FUNCTIONAL ORGANIZATION IN THE MONKEY INFEROTEMPORAL CORTEX. *science*, 272 (pp. 1665-1668).

[8] Weinshall, D., Edelman, S. and Bülthoff, H. H. (1990). A self-organizing multiple-view representation of 3D objects. In Touretzky, D. S., (eds), *Advances in Neural Information Processing Systems 2*. Morgan Kaufmann Publishers, San Mateo, CA. 274-281.