PAPER

Self-Organizing Map with False-Neighbor Degree between Neurons for Effective Self-Organization

Haruna MATSUSHITA[†a)], *Student Member and* Yoshifumi NISHIO[†b)], *Member*

SUMMARY In the real world, it is not always true that neighboring houses are physically adjacent or close to each other. in other words, "neighbors" are not always "true neighbors." In this study, we propose a new Self-Organizing Map (SOM) algorithm, SOM with False-Neighbor degree between neurons (called FN-SOM). The behavior of FN-SOM is investigated with learning for various input data. We confirm that FN-SOM can obtain a more effective map reflecting the distribution state of input data than the conventional SOM and Growing Grid.

key words: *self-organizing maps, clustering, feature extraction, visualization*

1. Introduction

Since we can accumulate a huge amount of data in recent years, it is important to investigate various clustering methods [1]. The Self-Organizing Map (SOM) has attracted attention for its clustering properties. SOM is an unsupervised neural network introduced by Kohonen in 1982 [2] and is a simplified model of the self-organization process of the brain. SOM obtains statistical features of input data and applies them to a wide field of data classifications. We can obtain the map reflecting the distribution state of input data using SOM. In the learning algorithm of SOM, a winner, which is a neuron with the weight vector closest to the input vector, and its neighboring neuron are updated, regardless of the distance between the input vector and the neighboring neuron. For this reason, if we apply SOM to clustering of the input data which include some clusters located at distant locations, there are some inactive neurons between clusters. Because inactive neurons are on a part without the input data, we are misled into thinking that there are some input data between clusters.

Meanwhile, in the real world, it is not always true that neighboring houses are physically adjacent or close to each other. For example, a case that the next-door house is at the top of a mountain whereas my house is at the foot (as Fig. 1(a)), and another case that there is a river, which does not have a bridge, between my house and my next-door house (as Fig. 1(b)). This means that "neighbors" are not always "true neighbors." In addition, the relationship between neighborhoods is not fixed, but keeps changing with time. It is important to change the neighborhood relationship flexibly according to the situation.

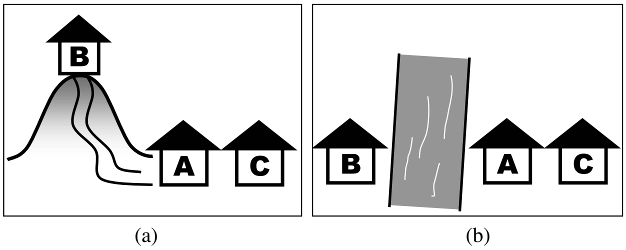


Fig. 1 What are the "neighbors"? The houses B and C is A's next-door neighbors on the left and on the right, respectively. (a) The house B is at the top of a mountain. (b) The river between A and B does not have a bridge.

On the other side, the synaptic strength is not constant in the brain. So far, the Growing Grid network was proposed in 1985 [3]. Growing Grid increases the neighborhood distance between neurons by increasing the number of neurons. However, there is not much research changing the synaptic strength even though there are algorithms which increase the number of neurons or consider rival neurons [4], [5].

In this study, we propose a new SOM algorithm, SOM with False-Neighbor degree between neurons (called FN-SOM). False-neighbor degrees are allocated between adjacent rows and adjacent columns of FN-SOM. We find the neuron q which has never become the winner, and the neurons, which is the most distant from q in a set of direct topological neighbors of q, are said to be "false neighbors" of q. The initial values of all of the false-neighbor degrees are set to zero, however, they are increased with learning, and the false-neighbor degrees act as a burden of the distance between map nodes when the weight vectors of neurons are updated. FN-SOM changes the neighborhood relationship more flexibly according to the situation and the shape of data.

We explain the learning algorithm of FN-SOM in detail in Sect. 4. The learning behaviors of FN-SOM for 2-dimensional input data and 3-dimensional data, which have some clustering problem, are investigated in Sect. 5. In addition, we apply FN-SOM to a real world data set, Iris data. Learning performance is evaluated both visually and quantitatively using three measurements. Furthermore, the results are compared with those obtained by the conventional SOM and Growing Grid. We can confirm that there are few inactive neurons using FN-SOM, and FN-SOM can obtain the most effective map reflecting the distribution state of input

Manuscript received October 26, 2007.
Manuscript revised February 8, 2008.

[†]The authors are with the Department of Electrical and Electronic Engineering, Tokushima University, Tokushima-shi, 770–8506 Japan.
a) E-mail: haruna@ee.tokushima-u.ac.jp
b) E-mail: nishio@ee.tokushima-u.ac.jp

DOI: 10.1093/ietfec/e91–a.6.1463

Copyright © 2008 The Institute of Electronics, Information and Communication Engineers

data in the three algorithms. In addition, we investigate the cluster structure of the learned map by calculating distances between neurons in order to confirm the clustering ability of FN-SOM. As the results, we confirm that FN-SOM can obtain more effective clustering map than SOM. In Sect. 6, in order to utilize the false-neighbor degree effectively, we propose a simple visualization method using the false-neighbor degrees. We apply this method to the result of the iris data, and we confirm that the method using false-neighbor degrees can obtain more effective clustering result than SOM.

2. Self-Organizing Map

First, we explain the learning algorithm of the conventional Self-Organizing Map (SOM). SOM consists of $n \times m$ neurons located at a regular low-dimensional grid, usually a 2-D $n \times m$ rectangular grid. The basic SOM algorithm is iterative. Each neuron i has a d-dimensional weight vector $\boldsymbol{w}_i = (w_{i1}, w_{i2}, \cdots, w_{id})$ $(i = 1, 2, \cdots, nm)$. The initial values of all the weight vectors are given over the input space at random. The range of the elements of d-dimensional input data $\boldsymbol{x}_j = (x_{j1}, x_{j2}, \cdots, x_{jd})$ $(j = 1, 2, \cdots, N)$ are assumed to be from 0 to 1.

(SOM1) An input vector $\boldsymbol{x}_j$ is inputted to all the neurons at the same time in parallel.
(SOM2) Distances between $\boldsymbol{x}_j$ and all the weight vectors are calculated. A winner, denoted by c, is the neuron with the weight vector closest to the input vector $\boldsymbol{x}_j$;

$$c = \arg\min_i \{\|\boldsymbol{w}_i - \boldsymbol{x}_j\|\}, \tag{1}$$

where $\|\cdot\|$ is the distance measure, Euclidean distance.
(SOM3) The weight vectors of the neurons are updated as

$$\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + h_{c,i}(t)(\boldsymbol{x}_j - \boldsymbol{w}_i(t)), \tag{2}$$

where t is the learning step. $h_{c,i}(t)$ is called the neighborhood function and is described as a Gaussian function;

$$h_{c,i}(t) = \alpha(t) \exp\left(-\frac{\|\boldsymbol{r}_i - \boldsymbol{r}_c\|^2}{2\sigma^2(t)}\right), \tag{3}$$

where $\|\boldsymbol{r}_i - \boldsymbol{r}_c\|$ is the distance between map nodes c and i on the map grid, $\alpha(t)$ is the learning rate, and $\sigma(t)$ corresponds to the width of the neighborhood function. Both $\alpha(t)$ and $\sigma(t)$ decrease with time, in this study, we use following equations;

$$\alpha(t) = \alpha_0(1 - t/t_{\max}), \quad \sigma(t) = \sigma_0(1 - t/t_{\max}), \tag{4}$$

where α_0 and σ_0 are the initial values of α and σ, respectively, and t_{max} is the maximum number of the learning.
(SOM4) The steps from (SOM1) to (SOM3) are repeated for all the input data.

3. Growing Grid

We explain an overview of the Growing Grid. The network of Growing Grid consists of nm neurons located at a rectangular $n \times m$ grid. Each neuron has a d-dimensional weight vector $\boldsymbol{w}_i$ as the conventional SOM. A winning frequency γ_i is associated with each neuron and is set to zero initially.

An input vector $\boldsymbol{x}_j$ is inputted to all the neurons, and a winner c is found according to Eq. (1). The weight vectors of the neurons are updated according to

$$\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + h_{Gc,i}(t)(\boldsymbol{x}_j - \boldsymbol{w}_i(t)), \tag{5}$$

where $h_{Gc,i}(t)$ is the neighborhood function of Growing Grid;

$$h_{Gc,i}(t) = \alpha_0 \exp\left(-\frac{{d_g}^2(c,i)}{2{\sigma_0}^2}\right), \tag{6}$$

where α_0 is a constant learning rate, and σ_0 is a constant width parameter. $d_g(c,i)$ is the distance on the grid between a winner c and each neuron i and is calculated by city-block distance (which is also known as L_1-norm). At each learning step, the winning frequency of c is incremented by ${\gamma_c}^{\text{new}} = {\gamma_c}^{\text{old}} + 1$.

After $n \times m \times \lambda_g$ number of learning steps have been performed, we determine the neuron q which has become the winner most frequently;

$$q = \arg\max_i \{\gamma_i\}. \tag{7}$$

We find the neuron f which is with the most different weight vector in 1-neighbor of q. We insert a new row (or column) between q and f. The weight vectors of the new neurons are interpolated from their neighbors which does increase the density of weight vectors in the vicinity of $\boldsymbol{w}_q$. The number n of rows (or m of columns) are increased, then all the winning frequencies are reset. We continue with the next round of learning unless $nm \geq nm_{\max}$ is fulfilled.

After the growth process is finished, we fine-tune the weight vectors using a decreasing learning rate. We perform $t'_{\max} = n \times m \times \lambda_f$ steps according to Eq. (5) using $\alpha(t') = \alpha_0(\alpha_1/\alpha_0)^{t'/t'_{\max}}$. t' denotes the learning step in the fine-tuning phase.

4. SOM with False-Neighbor Degree (FN-SOM)

We explain a new SOM algorithm, SOM with False-Neighbor Degree between neurons (FN-SOM). False-neighbor degrees of rows R_r $(1 \leq r \leq n-1)$ are allocated between adjacent rows of FN-SOM with the size of $n \times m$ grid (as Fig. 2). Likewise, false-neighbor degrees of columns C_k $(1 \leq k \leq m-1)$ are allocated between adjacent columns of FN-SOM. In other words, R_1 means the false-neighbor degree between neurons of the 1st row and the 2nd row, and C_4 is the false-neighbor degree between neurons of the 4th column and the 5th column. The initial values of all of the false-neighbor degrees are set to zero, and the initial values of all the weight vectors are given over the input space at random. Moreover, a winning frequency γ_i is associated with each neuron and is set to zero initially.

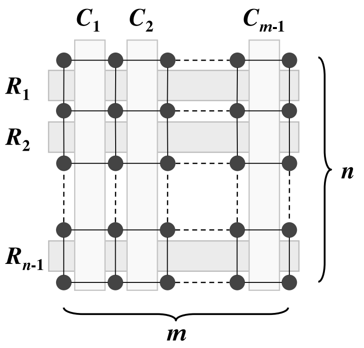


Fig. 2 A false-neighbor degree of row R_r ($1 \leq r \leq n-1$) and column C_k ($1 \leq k \leq m-1$). Neurons of FN-SOM are located at a $n \times m$ rectangular grid.

Learning Step

(FN-SOM1) An input vector $\boldsymbol{x}_j$ is inputted to all the neurons at the same time in parallel.

(FN-SOM2) Distances between $\boldsymbol{x}_j$ and all the weight vectors are calculated, and the winner c is found according to Eq. (1).

(FN-SOM3) Increment of the winning frequency of winner c by $\gamma_c{}^{\text{new}} = \gamma_c{}^{\text{old}} + 1$.

(FN-SOM4) The neighboring distances between the winner c and the other neurons are calculated. For instance, for two neurons s_1, which is located at r_1-th row and k_1-th column, and s_2, which is located at r_2-th row and k_2-th column, the neighboring distance is defined as the following measure;

$$d_f(s_1, s_2) = \left(|r_1 - r_2| + \sum_{r=r_1}^{r_2-1} R_r\right)^2 + \left(|k_1 - k_2| + \sum_{k=k_1}^{k_2-1} C_k\right)^2, \quad (8)$$

where $r_1 < r_2$, $k_1 < k_2$, namely, $\sum_{r=r_1}^{r_2-1} R_r$ means the sum of the false-neighbor degrees between the rows r_1 and r_2, and $\sum_{k=k_1}^{k_2-1} C_k$ means the sum of the false-neighbor degrees between the column k_1 and k_2.

(FN-SOM5) The weight vectors of the neurons are updated as

$$\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + h_{Fc,i}(t)(\boldsymbol{x}_j - \boldsymbol{w}_i(t)), \quad (9)$$

where $h_{Fc,i}(t)$ is the neighborhood function of FN-SOM:

$$h_{Fc,i}(t) = \alpha(t) \exp\left(-\frac{d_f(c,i)}{2\sigma^2(t)}\right). \quad (10)$$

(FN-SOM6) If $\sum_{i=1}^{nm} \gamma_i \geq \lambda$ is satisfied, we find the false-neighbors and increase the false-neighboring degree, according to steps from (FN-SOM7) to (FN-SOM10). If not, we perform step (FN-SOM11). In other words, we consider the false-neighbors every time when the learning steps are performed for λ input data.

Considering False-Neighbors

(FN-SOM7) We find a set of neurons S which have never become the winner:

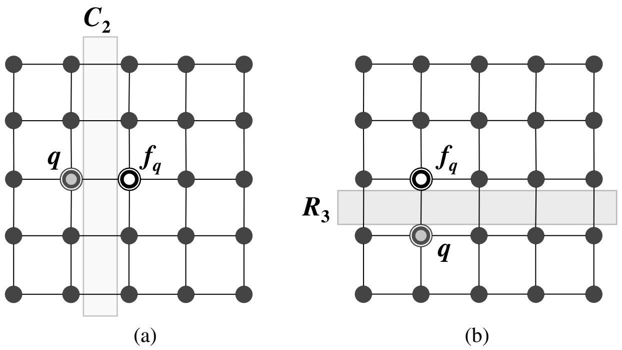


Fig. 3 Increment the false-neighbor degree. (a) q and its false-neighbor f_q are in the 3rd row and in the 2nd and 3rd column, respectively. Then, the false-neighbor degree C_2 between columns 2 and 3 is increased by Eq. (13). (b) q and f_q are in the 2nd column and in the 4th and 3rd row, respectively. Then, the false-neighbor degree R_3 between rows 3 and 4 is increased by Eq. (14).

$$S = \{i \mid \gamma_i = 0\}. \quad (11)$$

If the neurons, which have never become the winner, do not exist, namely $S = \emptyset$, we return to (FN-SOM1) without considering the false-neighbors.

(FN-SOM8) A false-neighbor f_q of each neuron q in S is chosen from the set of direct topological neighbors of q denoted as N_{q1}. f_q is the neuron whose weight vector is most distant from q:

$$f_q = \arg\max_i\{\|\boldsymbol{w}_i - \boldsymbol{w}_q\|\}, \quad q \in S,\ i \in N_{q1}. \quad (12)$$

(FN-SOM9) A false-neighbor degree between each q and its false-neighbor f_q, R_r or C_k, is increased. If q and f_q are in the r-th row and in the k-th and $(k+1)$-th column (as Fig. 3(a)), the false-neighbor degree C_k between columns k and $k+1$ is increased according to

$$C_k{}^{\text{new}} = C_k{}^{\text{old}} + \frac{n+m}{2nm}. \quad (13)$$

In the same way, if q and f_q are in the k-th column and in the $(r+1)$-th and r-th row (as Fig. 3(b)), the false-neighbor degree R_r between rows r and $r+1$ is also increased according to

$$R_r{}^{\text{new}} = R_r{}^{\text{old}} + \frac{n+m}{2nm}. \quad (14)$$

The increment $(n+m)/2nm$ is the average of $1/n$ and $1/m$. This is because Eqs. (13) and (14) are performed for all q in the set S, therefore, C_k or R_r is increased at most n or m times for every λ inputs.

(FN-SOM10) The winning frequency of all the neurons are reset to zero: $\gamma_i = 0$.

(FN-SOM11) The steps from (FN-SOM1) to (FN-SOM10) are repeated for all the input data.

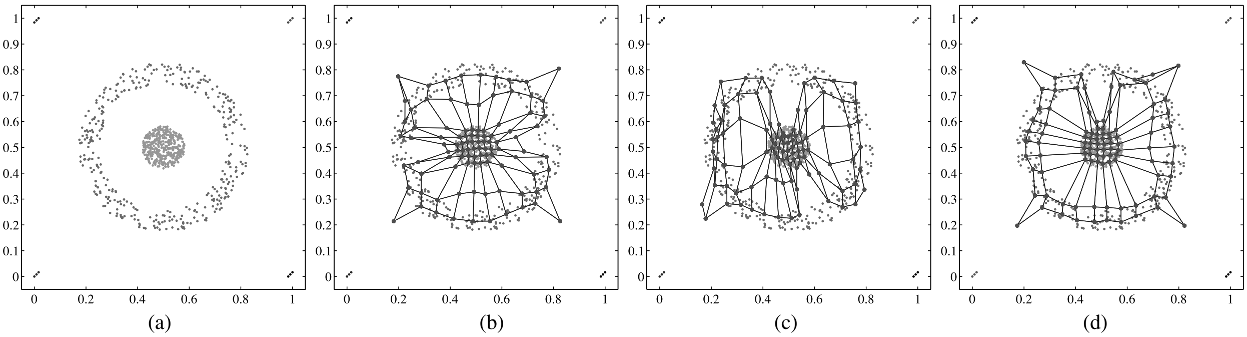


Fig. 4 Learning results of three algorithms for Target data. (a) Input data. (b) Conventional SOM. (c) Growing Grid. (d) FN-SOM.

5. Experimental Results

We apply FN-SOM to various input data and compare FN-SOM with the conventional SOM and Growing Grid.

5.1 For 2-dimensional Data

First, we consider 2-dimensional input data as shown in Fig. 4(a). The input data is Target data set, which has a clustering problem of outliers [6]. The total number of the input data N is 770, and the input data has six clusters which include 4 outliers. All the input data are sorted at random.

Both the conventional SOM and FN-SOM have $nm = 100$ neurons (10×10). Growing Grid starts learning with a 2×2 neurons, and new rows and columns are inserted as long as the number of neurons is less than $nm_{\max} = 100$. We repeat the learning 20 times for all input data, namely $t_{\max} = 15400$. The parameters of the learning are chosen as follows;
(For SOM)

$$\alpha_0 = 0.3,\ \sigma_0 = 4,$$

(For Growing Grid)

$$\alpha_0 = 0.1,\ \sigma_0 = 0.9,\ \lambda_g = 30,\ \alpha_1 = 0.005,\ \lambda_f = 100,$$

(For FN-SOM)

$$\alpha_0 = 0.3,\ \sigma_0 = 4,\ \lambda = 3000,$$

where we use the same α_0 and σ_0 to SOM and FN-SOM for the comparison and the confirmation of the false-neighbor degree effect.

The learning results of the conventional SOM and the Growing Grid are shown in Figs. 4(b) and (c), respectively. We can see that there are some inactive neurons between clusters. The other side, the result of FN-SOM is shown in Fig. 4(d). We can see from this figure that there are just a few inactive neurons between clusters, and FN-SOM can obtain the more effective map reflecting the distribution state of input data than SOM and the Growing Grid.

Furthermore, in order to evaluate the learning performance of FN-SOM in comparison with the conventional SOM and Growing Grid, we use the following three measurements to evaluate the training performance of the three algorithms.

Table 1 Quantization error Qe, Topographic error Te and Neuron utilization U for Target data.

	SOM	Growing Grid	FN-SOM
Qe	0.0207	0.0237	0.0191
Te	0.0740	0.2455	0.0442
U	0.8100	0.8137	0.9100

Quantization Error Qe**:** This measures the average distance between each input vector and its winner [2];

$$Qe = \frac{1}{N}\sum_{j=1}^{N} \|\boldsymbol{x}_j - \bar{\boldsymbol{w}}_j\|, \tag{15}$$

where $\bar{\boldsymbol{w}}_j$ is the weight vector of the corresponding winner of the input vector $\boldsymbol{x}_j$. Therefore, the small value Qe is more desirable.

Topographic Error Te**:** This describes how well the SOM preserves the topology of the studied data set [7];

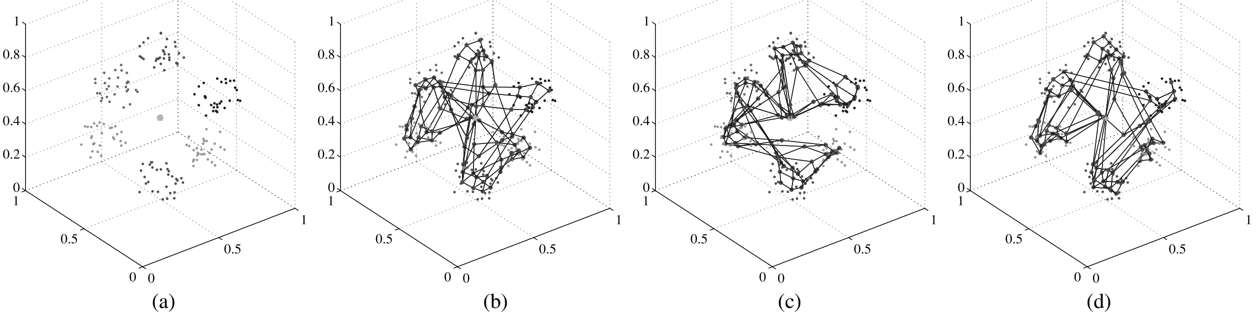
$$Te = \frac{1}{N}\sum_{j=1}^{N} u(\boldsymbol{x}_j), \tag{16}$$

where N is the total number of input data, $u(\boldsymbol{x}_j)$ is 1 if the winner and 2nd winner of $\boldsymbol{x}_j$ are NOT 1-neighbors each other, otherwise $u(\boldsymbol{x}_j)$ is 0. The small value Te is more desirable. Unlike the quantization error, it considers the structure of the map. For a strangely twisted map, the topographic error is big even if the quantization error is small.

Neuron Utilization U**:** This measures the percentage of neurons that are the winner of one or more input vector in the map [5];

$$U = \frac{1}{nm}\sum_{i=1}^{nm} u_i, \tag{17}$$

where $u_i = 1$ if the neuron i is the winner of one or more input data. Otherwise, $u_i = 0$. Thus, U nearer 1.0 is more desirable and larger value of U means fewer inactive neurons.



(a) (b) (c) (d)

Fig. 5 Learning results of three algorithms for Hepta data. (a) Input data. (b) Conventional SOM. (c) Growing Grid. (d) FN-SOM.

The calculated three measurements are shown in Table 1. The quantization error *Qe* of FN-SOM is the smallest value in the three algorithms, and by using FN-SOM, *Qe* has improved 7.7% from using the conventional SOM. This means that there are few errors between the input data and the neurons of the learned map, in other words, FN-SOM can be most reflecting the distribution state of the input data. The neuron utilization *U* of FN-SOM is also the best value in the three algorithms. It means that 90% of the neurons of FN-SOM are the winner of one or more input data, namely, there are few inactive neurons. On the other hand, the topographic error *Te* of FN-SOM is the smallest value although *Qe* and *U* are the best values. It means that FN-SOM self-organizes most effectively with maintenance of top quality topology. We consider these obtained results. The inactive neurons are on a part without the input data. Therefore, if the inactive neurons are few, more neurons can self-organize the input data of clusters. In consequence, the distance between each input data and its neurons will be small, namely the quantization error *Qe* will be a small value. Furthermore, the neurons of FN-SOM are not affected by the false-neighbors, so, the neurons can learn more distant for the distant input data, than SOM learning. For these reasons, FN-SOM obtains the best results.

5.2 For 3-dimensional Data

Next, we apply the three algorithms to 3-dimensional input data, Hepta data set [6], as shown in Fig. 5(a). The input data has a clustering problem of different variances. The total number of the input data *N* is 212, and the input data has seven clusters. All the input data are sorted at random.

We repeat the learning 70 times for all input data, namely $t_{\max} = 14840$. The input data are normalized and are sorted at random. The learning conditions are the same used in Fig. 4.

The learning results of three algorithms are shown in Figs. 5(b), (c) and (d), respectively. We can see that FN-SOM has the fewest inactive neurons between clusters. Three map quality measures are shown in Table 2. Results of FN-SOM are the best values in all map quality measures.

Table 2 Quantization error *Qe*, Topographic error *Te* and Neuron utilization *U* for Hepta data.

	SOM	Growing Grid	FN-SOM
Qe	0.0360	0.0409	0.0306
Te	0.1698	0.1509	0.1462
U	0.6500	0.6863	0.8000

Table 3 Quantization error *Qe*, Topographic error *Te* and Neuron utilization *U* for Iris data.

	SOM	Growing Grid	FN-SOM
Qe	0.0374	0.0452	0.0323
Te	0.2400	0.2267	0.1667
U	0.7200	0.7048	0.8200

The quantization error *Qe* of FN-SOM is the smallest value and has improved 15.0% from using the conventional SOM. Nonetheless, the topographic error *Te* of FN-SOM is also the smallest value, and it has improved 13.9% from using the conventional SOM and 3.1% from using Growing Grid. The neuron utilization *U* of FN-SOM is also the best value in three SOMs, and it has improved 23.1% from using the conventional SOM and 16.6% from using Growing Grid. From these results, we can say that FN-SOM is the most effective.

5.3 For Iris Data

Furthermore, we apply FN-SOM to the real world clustering problem. We use the Iris plant data [8] as real data. This data is one of the best known databases to be found in the pattern recognition literature [9]. The data set contains three clusters of 50 instances respectively, where each class refers to a type of iris plant. The number of attributes is four as the sepal length, the sepal width, the petal length and the petal width, namely, the input data are 4-dimension. The three classes correspond to *Iris setosa*, *Iris versicolor* and *Iris virginica*, respectively. *Iris setosa* is linearly separable from the other two, however *Iris versicolor* and *Iris virginica* are not linearly separable from each other.

We repeat the learning 100 times for all input data, namely $t_{\max} = 15000$. The input data are normalized and are sorted at random. The learning conditions are the same

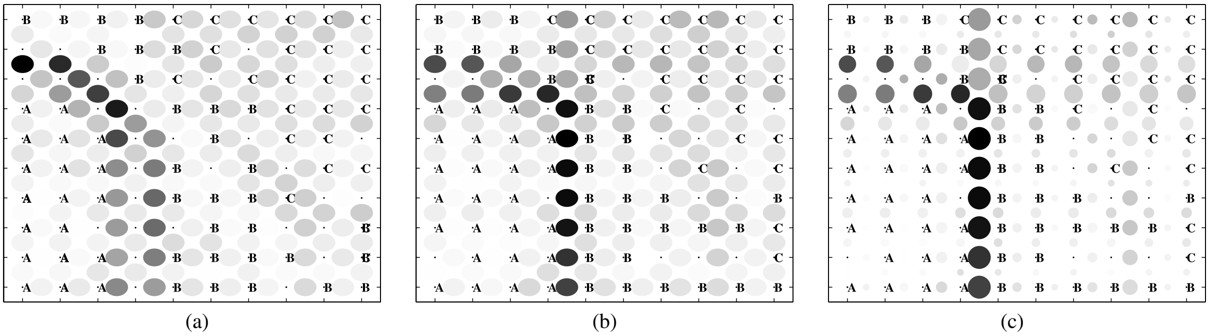


Fig. 6 Visualization results of Iris data. Label A, B and C correspond to *Iris setosa, Iris versicolor* and *Iris virginica*, respectively. (a) Conventional SOM. (b) FN-SOM. (c) FN-SOM using false-neighbor degree.

used in Fig. 4.

The calculated quantization error *Qe*, the topographic error *Te* and the neuron utilization *U* are shown in Table 3. We confirm that the quantization error *Qe* and the topographic error *Te* of FN-SOM are the smallest value in the three algorithms. *Qe* of FN-SOM has improved 13.6% from using the conventional SOM and *Te* of FN-SOM has also improved 30.5% from using the conventional SOM. This is because the result of FN-SOM hardly has inactive neurons between *Iris setosa* and the other two, therefore, the more neurons can self-organize the data of *Iris versicolor* and *Iris virginica*. Furthermore, *U* of FN-SOM is also the best value. From these results, we can confirm the efficiency of FN-SOM.

Furthermore, we investigate the cluster structure of the learned map in order to confirm the clustering ability of FN-SOM. Figures 6(a) and (b) show distances between adjacent neurons of the learning results of the conventional SOM and FN-SOM for Iris data, respectively. This figure thus visualizes the cluster structure of the map. Black circles on this figure mean large distance between neighboring map nodes. Clusters are typically uniform areas of white circles. We can see that the boundary lines of FN-SOM are clearer than the conventional SOM because FN-SOM has few inactive neurons. From these results, we can say that FN-SOM can be applied to clustering and can obtain more effective clustering result than SOM.

6. Visualization Method Using False-Neighbor Degrees

FN-SOM learns not only input vectors but also the false-neighbor degree between the neurons. In order to utilize the false-neighbor degree effectively, we propose a simple visualization method using the false-neighbor degrees.

Figure 6(c) shows not only the distances but also the false-neighbor degrees. The false-neighbor degree reflects the size of circles. The bigger the false-neighbor degree between the neurons, the bigger the circle is shown. From this figure, we can see the boundary line and cluster relation more clearly.

It is a very simple visualization method using the false-neighbor degrees. The FN-SOM holds the possibility of realizing a more effective visualization method.

7. Conclusions

In this study, we have proposed a new SOM algorithm, SOM with False-Neighbor degree between neurons (called FN-SOM). False-neighbor degrees are allocated between adjacent rows and adjacent columns of FN-SOM. The initial values of all of the false-neighbor degrees are set to zero, however, they are increased with learning, and the false-neighbor degrees act as a burden of the distance between map nodes when the weight vectors of neurons are updated.

We have applied FN-SOM to 2-dimensional data, 3-dimensional data and Iris data, and we have investigated the learning behaviors of FN-SOM. Furthermore, the results were compared with those obtained by the conventional SOM and Growing Grid. We have confirmed that the quantization error and the topographic error of FN-SOM were the smallest value among the three algorithms. Moreover, the neuron utilization of FN-SOM was the largest value among the three algorithms. From these results, we have confirmed the efficiency of FN-SOM.

Besides, in order to confirm the clustering ability of FN-SOM, we have investigated the cluster structure of the map by calculating the distances between neurons. As the results, FN-SOM has obtained more effective clustering map than SOM, although investigating the clustering ability quantitatively is our future work.

In addition, in order to utilize the false-neighbor degree effectively, we have proposed a simple visualization method using the false-neighbor degrees. We have confirmed the effectiveness of the proposed method and that FN-SOM holds the possibility of realizing a more effective visualization method.

Acknowledgment

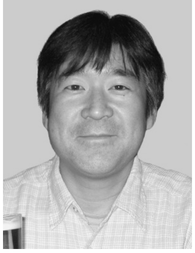
This work was supported by Grant-Aid for JSPS Fellows (19-7384).

References

[1] J. Vesanto and E. Alhoniemi, “Clustering of the self-organizing map,”

IEEE Trans. Neural Netw., vol.11, no.3, pp.586–600, 2002.
[2] T. Kohonen, Self-organizing Maps, Berlin, Springer, 1995.
[3] B. Fritzke, "Growing grid — A self-organizing network with constant neighborhood range and adaptation strength," Neural Processing Letters, vol.2, no.5, pp.9–13, 1995.
[4] L. Xu, A. Krzyzak, and E. Oja, "Rival penalized competitive learning for clustering analysis, RBF net, and curve detection," IEEE Trans. Neural Netw., vol.4, no.4, pp.636–649, 1993.
[5] Y. Cheung and L. Law, "Rival-model penalized self-organizing map," IEEE Trans. Neural Netw., vol.18, no.1, pp.289–295, 2007.
[6] A. Ultsch, "Clustering with SOM: U*C," Proc. Workshop on Self-Organizing Maps, pp.75–82, 2005.
[7] K. Kiviluoto, "Topology preservation in self-organizing maps," Proc. International Conference on Neural Networks, pp.294–299, 1996.
[8] D.J. Newman, S. Hettich, C.L. Blake, and C.J. Merz, UCI Repository of Machine Learning Database, 1998, http://www.ics.uci.edu/˜mlearn/MLRepository.html
[9] R.A. Fisher, "The use of multiple measurements in taxonomic problems," Annual Eugenics, no.7, part II, pp.179–188, 1936.

Haruna Matsushita was born in Tokushima, Japan, in 1982. She received the B.E., M.E. degrees from Tokushima University, Tokushima, Japan, in 2005 and 2007. She is currently working towards Ph.D. degree at the same university. Her research interests include theory and application of Self-Organizing Maps. She is a student member of the IEEE.



Yoshifumi Nishio received B.E., M.E., and Ph.D. degrees in electrical engineering from Keio University, Yokohama Japan, in 1988, 1990, and 1993, respectively. In 1993, he joined the Department of Electrical and Electronic Engineering at Tokushima University, Tokushima Japan, where he is currently an Associate Professor. From May 2000 he spent a year in the Laboratory of Nonlinear Systems (LANOS) at the Swiss Federal Institute of Technology Lausanne (EPFL) as a Visiting Professor. His research interests include analysis and application of chaos in electrical circuits, analysis of synchronization in coupled oscillatory circuits, development of analyzing methods for nonlinear circuits, theory and application of cellular neural networks, and neural network architecture. He was the Chair of the IEEE CASS Technical Committee on Nonlinear Circuits and Systems (NCAS) during 2004–2005, the Steering Committee Secretary of the IEICE Research Society of Nonlinear Theory and its Applications (NOLTA) during 2004–2007, and is currently the Secretary/Treasurer of the IEEE CASS Shikoku Chapter. He is serving as an Associate Editor for the IEEE CAS Magazine, the IEEE CASS Newsletter, and the RISP Journal of Signal Processing, and the Editor for the IEICE Fundamentals Review. He is a senior member of the IEEE and the RISP.