Batch-Learning Self-Organizing Map with Weighted Connections Avoiding False-Neighbor Effects

Haruna Matsushita and Yoshifumi Nishio

Abstract— **This study proposes a Batch-Learning Self-Organizing Map with Weighted Connections avoiding false-neighbor effects (BL-WCSOM). We apply BL-WCSOM to several high-dimensional datasets. From results measured in terms of the quantization error, inactive neurons, the topographic error and the computation time, we confirm that BL-WCSOM obtain the effective map reflecting the distribution state of the input data using fewer neurons in less time.**

I. Introduction

The Self-Organizing Map (SOM) is an unsupervised neural network [1] and has attracted attention for its clustering properties [2]. SOM consists of neurons arranged on 2 or 3 dimensional grid, called "map". In the learning algorithm of SOM, the winning neuron, the one closest to the input data, and its neighboring neurons on the map are updated, regardless of the distance between the input data and the neighboring neurons. For this reason, if we apply SOM to a clustering of the input data including some clusters located at distant locations, there are some inactive neurons between clusters. Because the inactive neurons are on a part without the input data, we are misled into thinking that there are some input data between clusters. Furthermore, because the simulation time depends on the number of neurons, it is important to utilize the neurons effectively by reducing the inactive neurons.

In the real world, it is not always true that neighboring houses are physically adjacent or close to each other. 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. Meanwhile, the synaptic strength is not constant in the brain. So far, the Growing Grid network was proposed in 1985 [3]. Growing Grid is one kind of the SOM and increases the neighborhood distance between neurons by increasing the number of neurons. However, there is not much research of SOM changing the synaptic strength itself even though there are algorithms which increase the number of neurons or consider rival neurons [4], [5].

In our past study, we proposed the SOM with False-Neighbor degree between neurons (FN-SOM) [6]. False-neighbor degrees (FNDs) are allocated between adjacent rows and adjacent columns of FN-SOM. FN-SOM can greatly reduce the inactive neurons by changing FNDs with learning, however, the algorithm has the following problem. All the FNDs between the neurons on same line are increased simultaneously and forcibly. It often produces the increase of FNDs between correct-neighboring neurons, namely, FALSE false-neighbor!

To solve this problem, we proposed the SOM with Weighted Connections (WC-SOM) [7]. In WC-SOM, unlike FN-SOM, all the connections between adjacent neurons are weighted to avoid false-neighbor effects. This weights are called as false-neighbor weights (FNWs). WC-SOM changes the neighborhood relationship more flexibly than FN-SOM according to the situation and the shape of data.

By the way, there are two well-known learning algorithms for SOM: sequential learning and batch learning. In the sequential learning, the winner for an input is found and the weight vectors are updated immediately. However, the sequential learning has some problems as a large calculation amount and a dependence on input order of the data. In the batch learning, the updates are deferred to the presentation of the whole dataset. Batch-Learning SOM (BL-SOM) [1] is used to speed up the calculation time and to remove the dependence on input order of the data.

In this study, we propose a Batch-Learning Self-Organizing Map with Weighted Connections avoiding false-neighbor effects (BL-WCSOM). We improve the previously proposed WC-SOM and apply WC-SOM to the batch learning. In the algorithm, we find winless neurons and its "false neighbors", and FNWs between these neurons are increased. The initial values of all of FNWs are set to zero, however, they are increased with learning, and FNWs act as a burden of the distance between map nodes when the weight vectors of neurons are updated. BL-WCSOM changes the neighborhood relationship flexibly according to the situation and the shape of data although using batch learning.

We compare BL-WCSOM with the conventional BL-SOM, the batch-learning FN-SOM [8] and the sequential-learning WC-SOM by measuring in terms of the quantization error, the number of inactive neurons, the topographic error and the computation time. The effectiveness of BL-WCSOM is investigated by applying it to various input dataset. From results, we confirm that BL-WCSOM can obtain the most effective map reflecting the distribution state of the input data using fewer neurons in less time.

II. Batch-Learning Self-Organizing Map (BL-SOM)

We explain a batch learning algorithm of SOM (BL-SOM) in detail. SOM consists of $n \times m$ neurons located at 2-

Haruna Matsushita is with the Department of Electrical and Electronic Engineering, Hosei University, Tokyo, 184–8584, Japan (phone: +81 42 387 6354; email: haruna.matsushita.pd@k.hosei.ac.jp).

Yoshifumi Nishio is with the Department of Electrical and Electronic Engineering, Tokushima University, Tokushima, 770–8506, Japan (phone: +81 88 656 7470; email: nishio@ee.tokushima-u.ac.jp).

978-1-4244-8126-2/10/$26.00

dimensional rectangular grid. Each neuron i is represented by a d-dimensional weight vector $\boldsymbol{w}_i = (w_{i1}, w_{i2}, \cdots, w_{id})$ $(i = 1, 2, \cdots, nm)$, where d is equal to the dimension of the input vector $\boldsymbol{x}_j = (x_{j1}, x_{j2}, \cdots, x_{jd})$ $(j = 1, 2, \cdots, N)$. The range of the elements of the input data $\boldsymbol{x}$ are assumed to be from 0 to 1. Also the batch learning algorithm is iterative, but instead of using a single data vector at a time, the whole dataset is presented to the map before any adjustments are made. The initial weight vectors of BL-SOM are given at random.

(BLSOM1) Input an input vector $\boldsymbol{x}_j$ to all the neurons at the same time in parallel.
(BLSOM2) Calculate distances between $\boldsymbol{x}_j$ and all the weight vectors. Relate the input vector $\boldsymbol{x}_j$ to a winning neuron which is closest to $\boldsymbol{x}_j$.

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

denotes the index of the winner of the input vector $\boldsymbol{x}_j$, where $\|\cdot\|$ is the distance measure, in this study, Euclidean distance. Let a training step $t = t + 1$.
(BLSOM3) After repeating the steps (BLSOM1) and (BLSOM2) for all the input dataset, update all the weight vectors according to

$$\boldsymbol{w}_i^{\text{new}} = \frac{\sum_{j=1}^{N} h_{c_j,i} \boldsymbol{x}_j}{\sum_{j=1}^{N} h_{c_j,i}}, \tag{2}$$

where $h_{c_j,i}$ is the neighborhood function described by

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

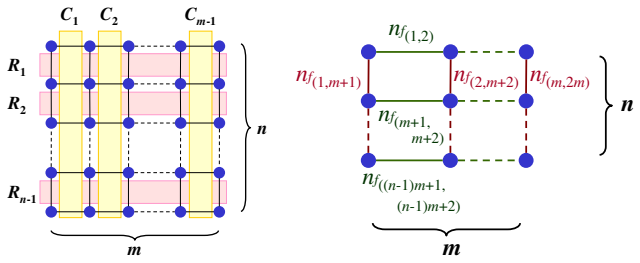
where $\boldsymbol{r}_i$ and $\boldsymbol{r}_{c_j}$ are coordinates of the i-th and the winning c_j neuron on the competitive layer, namely, $\|\boldsymbol{r}_i - \boldsymbol{r}_{c_j}\|$ is the distance between map nodes c_j and i on the map grid. $\sigma(t)$ decreases with time, in this study, we use following equation;

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

where σ_0 is the initial value of σ, and t_{max} is the maximum number of the learning.
(BLSOM4) Repeat the steps from (BLSOM1) to (BLSOM3) sufficient time.

III. BL-SOM with Weighted Connections (BL-WCSOM)

We explain a Batch-Learning SOM with Weighted Connections avoiding false-neighbor effects (BL-WCSOM) in detail. In the conventional method, FN-SOM, has false-neighbor degrees (FNDs) between neurons as shown in Fig. 1(a). On the other hand, in BL-WCSOM, a false-neighbor weight (FWN) denoted by $n_{f_{(i,k)}}$ is allocated between directly-connected neurons i and k as shown in Fig. 1(b), and we calculate the neighborhood distance considering FNWs. The initial values of all FNWs 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 i and is set to zero initially.



(a) (b)

Fig. 1. Differences between false-neighbor degree of FN-SOM and false-neighbor weights of WC-SOM. Neurons of both SOMs are located at a $n \times m$ rectangular grid. (a) A false-neighbor degree of row R_r $(1 \le r \le n-1)$ and column C_k $(1 \le k \le m-1)$. (b) False-neighbor weights $n_{f_{(i,k)}}$ are allocated between all the directly-connected neurons i and k. $k \in N_{i1}$

The learning algorithm of BL-FNSOM consists of two steps; *Learning steps* and *Consideration of False-Neighbor*. A flowchart of the learning algorithm is shown in Fig. 2.

Learning Steps
(BL-WCSOM1) Input an input vector $\boldsymbol{x}_j$ to all the neurons.
(BL-WCSOM2) Calculate distances between $\boldsymbol{x}_j$ and all the weight vectors, and find the winner c_j according to Eq. (1). Increase the winning frequency of the winner c_j by ${\gamma_{c_j}}^{\text{new}} = {\gamma_{c_j}}^{\text{old}} + 1$. Let $t = t + 1$.
(BL-WCSOM3) After repeating the steps (BL-WCSOM1) and (BL-WCSOM2) for all the input dataset, update all the weight vectors according to

$$\boldsymbol{w}_i^{\text{new}} = \frac{\sum_{j=1}^{N} h_{Fc_j,i} \boldsymbol{x}_j}{\sum_{j=1}^{N} h_{Fc_j,i}}, \tag{5}$$

where $h_{Fc_j,i}$ is the neighborhood function of BL-WCSOM;

$$h_{Fc_j,i}(t) = \alpha(t) \exp\left(-\frac{dis(c_j,i)}{2\sigma^2(t)}\right). \tag{6}$$

where $dis(c_j, i)$ is the neighborhood distance between the winner c_j and each neuron i by considering FNWs n_f as the following measure;

$$dis(c_j,i) = \|\boldsymbol{r}_i - \boldsymbol{r}_c\|^2 + d_{f(c_j,i)}, \tag{7}$$

where $d_{f(c_j,i)}$ is the connection weight between c and i, and it is defined as the minimum of sum-of-FNWs on the shortest-path from c_j to i (as shown in Fig. 3).
(BL-WCSOM4) If $\sum_{i=1}^{nm} \gamma_i \ge \lambda$ is satisfied, find the false-neighbor and increase FNWs n_f, according to steps from (BL-WCSOM5) to (BL-WCSOM8). If not, perform step (BL-WCSOM9).

Consideration of False-Neighbor
(BL-WCSOM5) Find a set of neurons S which have never become the winner; $S = \{i \mid \gamma_i = 0\}$. If the winless neuron does not exist, namely $S = \emptyset$, return to (BL-WCSOM1) without considering the false-neighbor.
(BL-WCSOM6) Choose a false-neighbor f_q of each neuron q in S from the set of direct topological neighbors of q

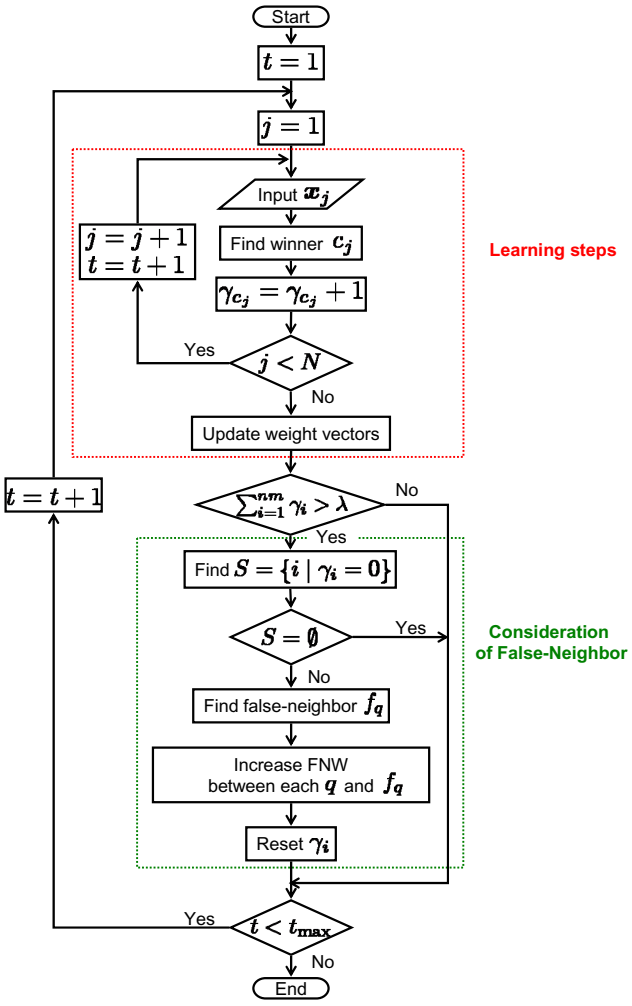
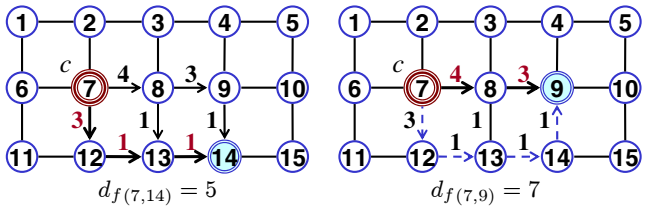


Fig. 2. Flowchart of BL-WCSOM



(a) (b)

Fig. 3. The connection weight $d_{f(c,i)}$ between the winner $c(=7)$ and the neuron i. $d_{f(c,i)}$ is the minimum of sum-of-FNWs on the shortest-path from c to i. Values in circles mean respective neuron numbers. Values between neurons mean FNWs n_f between directly-connected neurons. (a) Let $i = 14$. The possible shortest path routes from c to i are [7 8 9 14], [7 8 13 14] and [7 12 13 14], and its sum of FNWs are 8, 6 and 5, respectively. Then, $d_{f(7,14)} = 5$ which is a minimum value. (b) Let $i = 9$. Then, $d_{f(7,9)} = 7$ because the possible shortest-path route is only [7 8 9]. Note that a route [7 12 13 14 9] is NOT adopted because it is not the shortest-path although the sum of FNWs is 6 which is smaller than 7.

denoted as N_{q_1}. f_q is a neuron which is the most distant from q;

$$f_q = \arg\max_i \{\|\boldsymbol{w}_i - \boldsymbol{w}_q\|\}, \quad q \in S, \ i \in N_{q_1}. \tag{8}$$

(BL-WCSOM7) Increase FNW $n_{f(q,f_q)}$ between each q and its false-neighbor f_q as

$$n_{f(q,f_q)} = n_{f(q,f_q)} + \Delta f, \tag{9}$$

where $n_{f(q,f_q)} = n_{f(f_q,q)}$ and Δf is a fixed value which is the amount of false-neighbor.

(BL-WCSOM8) Reset the winning frequencies of all the neurons to zero; $\gamma_i = 0$.

(BL-WCSOM9) Repeat the steps from (BL-WCSOM1) to (BL-WCSOM8) for all the input data.

IV. Experimental Results

We applied BL-WCSOM to complex clustering problems and compared it with the conventional Batch-Learning SOM (BL-SOM), the Batch-Learning FN-SOM (BL-FNSOM) and the sequential-learning WC-SOM (WC-SOM).

TABLE I

Parameters of four SOMs.

Algorithm	α_0^*	σ_0	λ^{**}	Δf^{***}
BL-SOM	-	4.5	-	-
BL-FNSOM	-	4.5	3000	0.1
WC-SOM	0.3	3.0	3000	1.0
BL-WCSOM	-	4.5	5000	1.0

*Initial learning rate,
**Frequency of consideration of false-neighbor.
***Increased amount of FND or FNW.

For the experiment, all the SOMs had $nm = 100$ neurons (size 10×10). The parameters of the learning were chosen as Table I.

A. For Target dataset

We considered Target dataset [9] which has a clustering problem of outliers. The input data is 2-dimension and has six clusters including four outliers, and the total number of the input data N is 770. We repeat the learning 26 times for all the input data, namely $t_{\max} = 20020$.

A learning result of the conventional BL-SOM is shown in Fig. 4(a). We can see that there are a lot of inactive neurons between clusters. The other side, it is clear that BL-FNSOM, WC-SOM and BL-WCSOM can greatly reduce the number of inactive neurons as shown in Figs. 4(b)–(d).

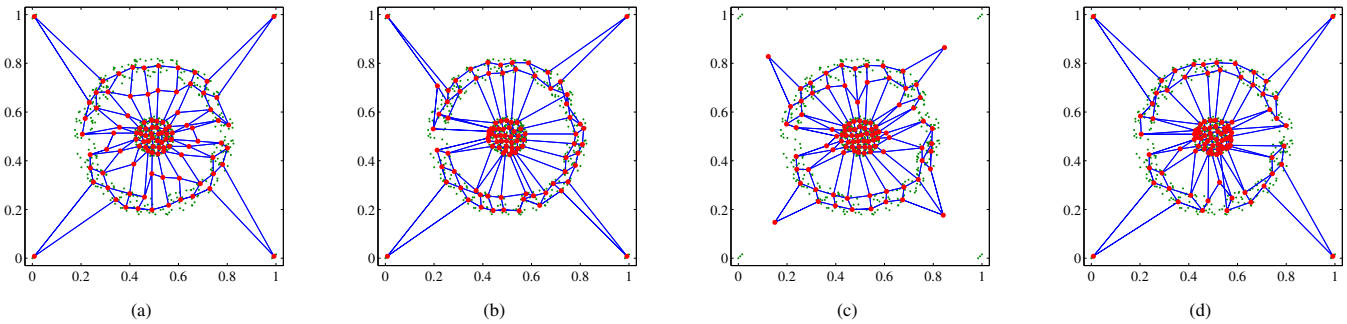
In order to evaluate the learning performance of four SOMs numerically, we use the following two measurements.

Quantization Error Qe**:** This measures the average distance between each input vector and its winner [1];

$$Qe = \frac{1}{N} \sum_{j=1}^{N} \|\boldsymbol{x}_j - \bar{\boldsymbol{w}}_j\|, \tag{10}$$

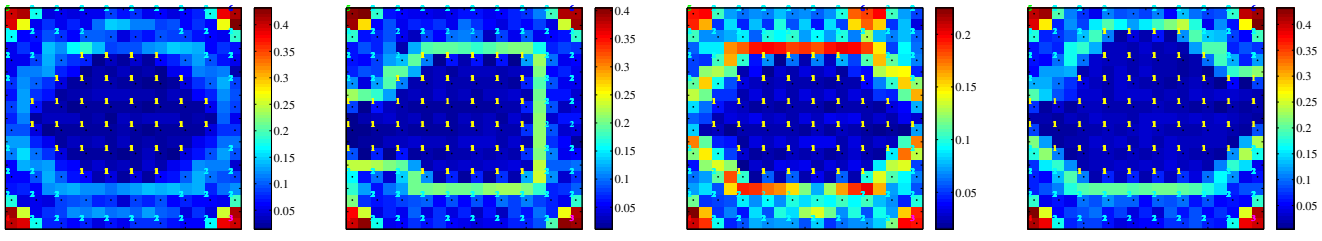
where $\bar{\boldsymbol{w}}_j$ is the weight vector of the corresponding winner of the input vector $\boldsymbol{x}_j$. Therefore, the small value Qe is more desirable.

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



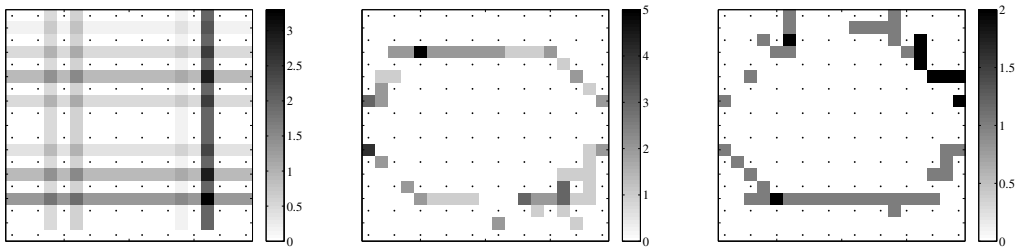
(a) (b) (c) (d)

Fig. 4. Learning results of four algorithms for Target data. (a) Conventional BL-SOM. (b) BL-FNSOM (c) WC-SOM. (d) BL-WCSOM.



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

Fig. 5. Data visualization by U-Matrices of simulation results for Target data. The winner of each input vector is found, and numbers by neurons show cluster numbers to which the input data belongs. (a) Conventional BL-SOM. (b) BL-FNSOM (c) WC-SOM. (d) BL-WCSOM.



(a) (b) (c)

Fig. 6. Gray scale display of false-neighbor degrees (FNDs) and false-neighbor weights (FNWs) after learning for Target data. (a) FNDs of BL-FNSOM. (b) FNWs of WC-SOM. (c) FNWs of BL-WCSOM.

the map [5];

$$U = \frac{1}{nm}\sum_{i=1}^{nm} u_i, \tag{11}$$

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.

TABLE II

QUANTIZATION ERROR Qe AND NEURON UTILIZATION U FOR TARGET DATA.

	BL-SOM	BL-FNSOM	WC-SOM	BL-WCSOM
Qe	0.0140	0.0134	0.01777	0.01374
U	0.8457	0.993	0.887	0.95333

We carried out 30 simulations with different initial weight vectors and different order of inputting even though three batch-learning SOMs do not depend on the order of input. The averages of two measurements over the 30 independent runs are listed in Table II. The quantization errors Qe of the three batch-learning SOMs are better than the sequential-learning, WC-SOM. This is because, compared with WC-SOM, the batch-learning SOMs converge more quickly and self-organized the data in every corner. Meanwhile, the neuron utilization U of the conventional BL-SOM is the worst value among the four algorithms. It means that the three SOMs considering the false-neighbor have fewer inactive neurons than the standard BL-SOM. Furthermore, because BL-WCSOM could obtain better Qe and U than the WC-SOM, we can say that BL-WCSOM can carry out the self-organization with few inactive neurons and with quick convergence as well or better than sequential-learning WC-SOM.

In order to evaluate how well SOM preserves the topology of the dataset, we calculated U-Matrix [9] which visualizes the cluster structure of the map by showing distances between neighboring neurons. Figure 5 shows U-Matrices of the four

TABLE III

COMPUTING TIMES FOR TARGET DATA. 30 SIMULATIONS.

BL-SOM	BL-FNSOM	WC-SOM	BL-WCSOM
10.73 [sec]	17.58 [sec]	52.16 [sec]	18.78 [sec]

algorithms. We can see that the boundary lines of BL-FNSOM, WC-SOM and BL-WCSOM, which consider the false-neighbor, We can see that the boundary lines of BL-FNSOM, WC-SOM and BL-WCSOM, which consider the false-neighbor, are clearer than the conventional BL-SOM shown in Fig. 5(a) and it is easy to distinguish between light areas (cluster) and dark areas (no input data) because there are few inactive neurons in the results of these SOMs. However, it should be noted that it is clear that U-Matrix of BL-FNSOM does not visualize the cluster structure correctly. It means that BL-FNSOM can not preserve the topology of the dataset by increasing its FNDs forcibly.

This can be confirmed by differences between FNDs of FN-SOM and FNWs of WC-SOM and BL-WCSOM after learning. Figure 6 shows FNDs and FNWs between the neurons displayed by gray-scale. From FNDs of BL-FNSOM as Fig. 6(a), we can clearly see that even FNDs between the correct-neighboring neurons were increased. This is because that all the FNDs between the neurons on same line are increased simultaneously and forcibly, in BL-FNSOM algorithm. On the other hand, in Fig. 6(b), WC-SOM can adapt its FNWs to the shape of the input data most flexibly among the three SOMs considering the false-neighbor. Additionally, we can see that also BL-WCSOM shown in Fig. 6(c) can adapt its FNWs flexibly, if not the same as the sequential-learning.

Furthermore, we investigated computing time which is one of the most important issues to evaluate the learning performance of SOM. The tests were run in a machine with 4GB of memory and 3.16GHz Intel Core 2 Duo 32-bit CPUs running Windows Vista operating system. The programs were run by Matlab whose version was 7.6 (R2008a).

Table III shows the total time of 30 independent runs. The sequential-learning WC-SOM is the worst among the four SOMs, and the proposed BL-WCSOM reduced the learning time 64% compared to that of WC-SOM. The conventional BL-SOM was the fastest, and second-fastest was BL-FNSOM. However, BL-SOM will never obtain the result, which includes few inactive neurons, as BL-FNSOM even if the number of learning are increased, and BL-FNSOM will never increase its FNDs flexibly as BL-FNSOM.

From these results, we can say that the proposed BL-WCSOM obtain effective results including few inactive neurons by varying its FNWs flexibly as the sequential-learning WC-SOM in less time.

B. For real world dataset

We applied BL-WCSOM to a real-world clustering problem, Wine dataset [10]. This dataset contains three clusters of 59, 71 and 48 instances respectively, namely $N = 178$, and its dimension d is 13 which is high dimension. We repeated the learning 113 times for all the input data, namely $t_{\max} = 20114$. The input data are normalized and are sorted at random.

TABLE IV

QUANTIZATION ERROR Qe AND NEURON UTILIZATION U FOR WINE DATASET.

	BL-SOM	BL-FNSOM	WC-SOM	BL-WCSOM
Qe	0.2078	0.1808	0.2362	0.2032
U	0.8367	0.949	0.8547	0.8533

TABLE V

COMPUTING TIMES FOR WINE DATASET. 30 SIMULATIONS.

BL-SOM	BL-FNSOM	WC-SOM	BL-WCSOM
20.01 [sec]	33.62 [sec]	55.43 [sec]	28.27 [sec]

Table IV summarizes the calculated two measurements of the results of the four SOMs. Because the quantization error Qe of the sequential-learning WC-SOM is the worst among the four SOMs, we can say that WC-SOM could not self-organize the data in every hole and corner, compared to other three batch-learning SOMs. However, the neuron utilization U of the conventional BL-SOM is the worst value. It means that BL-SOM included more inactive neurons than other SOMs considering the false-neighbor. BL-WCSOM achieved a satisfactory level of results as BL-FNSOM and WC-SOM.

The U-Matrix calculated from each learning result is shown in Fig. 7. We can confirmed that the boundary lines of WC-SOM and BL-WCSOM are clear. Although BL-FNSOM has few inactive neurons, its boundary lines are not clear as BL-SOM without consideration of the false-neighbor.

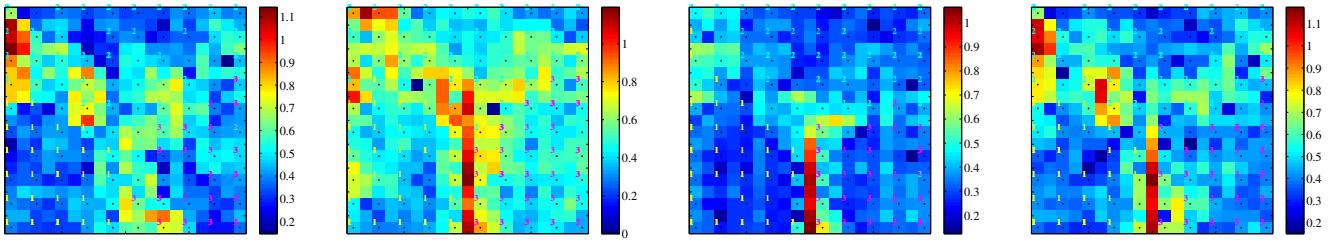
This is backed by FNDs and FNWs after learning shown in Fig. 8. FNWs of WC-SOM and BL-WCSOM were increased flexibly depending on the shape of the input data in comparison with BL-FNSOM. It means that in contrast to BL-FNSOM which could not preserve the topology of the data, BL-WCSOM enabled flexibly self-organization with varying its FNWs correctly as WC-SOM.

The computing time of 30 simulations are shown in Table V. The three batch-learning SOMs are faster than the sequential-learning WC-SOM. The proposed BL-WCSOM reduced the computation time 49% compared to that of WC-SOM.

From these results measured in terms of the quantization error, the inactive neurons, the topographic error and the computing time, we can say that the proposed BL-WCSOM can carry out effective self-organization for the real-world high dimensional dataset in less time.

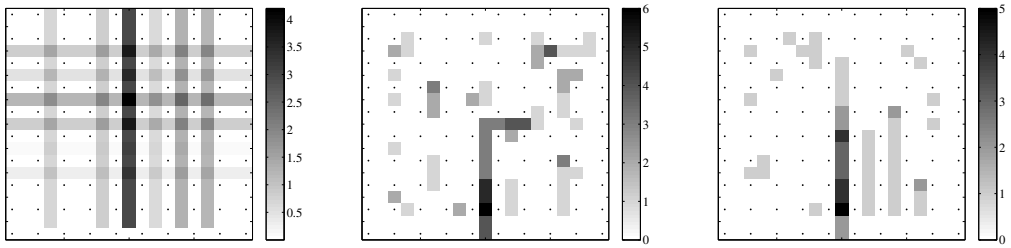
V. CONCLUSIONS

This study has proposed the Batch-Learning Self-Organizing Map with Weighted Connections (BL-WCSOM). BL-WCSOM has the false-neighbor weights (FNWs) allocated between connections of adjacent neurons to avoid the false-neighbor effects. In the algorithm, the calculation method of the neighborhood distance has been proposed in accordance with FNWs. We have compared WC-SOM



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

Fig. 7. Data visualization by U-Matrices of simulation results for Wine dataset. (a) Conventional BL-SOM. (b) BL-FNSOM (c) WC-SOM. (d) BL-WCSOM.



(a) (b) (c)

Fig. 8. Gray scale display of false-neighbor degrees (FNDs) and false-neighbor weights (FNWs) after learning for Wine dataset. (a) FNDs of BL-FNSOM. (b) FNWs of WC-SOM. (c) FNWs of BL-WCSOM.

with the conventional SOM, Growing Grid and FN-SOM and have confirmed that WC-SOM enabled the most flexible self-organization by increasing FNWs with learning. Furthermore, we have confirmed visually and numerically that BL-WCSOM could carry out effective self-organization for the real-world high dimensional dataset in less time.

REFERENCES

[1] T. Kohonen, *Self-organizing Maps*, Berlin, Springer, 1995.

[2] J. Vesanto and E. Alhoniemi, "Clustering of the Self-Organizing Map," *IEEE Trans. Neural Networks*, vol. 11, no. 3, pp. 586–600, 2002.

[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 Networks*, vol. 4, no. 4, pp. 636–649, 1993.

[5] Y. Cheung and L. Law, "Rival-Model Penalized Self-Organizing Map," *IEEE Trans. Neural Networks*, vol. 18, no. 1, pp. 289–295, 2007.

[6] H. Matsushita and Y. Nishio, "Self-Organizing Map with False-Neighbor Degree between Neurons for Effective Self-Organization," *IEICE Trans. on Fundamentals*, vol. E91-A, no. 6, pp. 1463–1469, 2008.

[7] H. Matsushita and Y. Nishio, "Self-Organizing Map with Weighted Connections Avoiding False-Neighbor Effects," *Proc. of IEEE International Symposium on Circuits and Systems*, 2009. (Accepted)

[8] H. Matsushita and Y. Nishio, "Batch-Learning Self-Organizing Map with False-Neighbor Degree between Neurons," *Proc. of International Joint Conference on Neural Networks*, pp. 2260–2267, 2008.

[9] A. Ultsch, "Clustering with SOM: U*C," *Proc. of International Workshop on Self-Organizing Maps*, pp. 75–82, 2005.

[10] D. J. Newman, S. Hettich, C. L. Blake and C. J. Merz, UCI Repository of Machine Learning Database, 1998.