Adaptive Learning Algorithm in Tree-Structured Self-Organizing Feature Map

Takashi Yamaguchi[*1], Takumi Ichimura[*2] & Kenneth James Mackin[*1]
[*1]Department of Information Systems, Tokyo University of Information Sciences
4-1 Onaridai, Wakaba-ku, Chiba, 265-8501 Japan
Email: tyamagu@edu.tuis.ac.jp, mackin@rsch.tuis.ac.jp
[*2]Faculty of Management and Information Systems, Prefectural University of Hiroshima
1-1-71, Ujina-Higashi, Minami-ku, Hiroshima, 734-8559, Japan
Email: ichimura@pu-hiroshima.ac.jp

***Abstract*— Self-Organizing Feature Map is a layered neural network consisting of an input layer and a competitive layer for the data visualization and vector quantization. The accuracy of SOM vector quantization depends on the number of competitive layer's neurons. Therefore, when an unknown data set is given, it is difficult to decide the sufficient competitive layer size. In this paper, we propose a hierarchical competitive layer adaptation method in order to find out the sufficient number of neurons. The proposed method adds and deletes neurons using the means error and frequency in use among neighboring neurons.**

I. INTRODUCTION

Self-organizing feature map (SOM) is a well known artificial neural network for the data visualization and the vector quantization [1]. SOM have been applied to various problems such as sound recognitions, image recognitions, and clustering etc [1]-[3]. SOM consists of an input layer and a competitive layer that neurons are arranged to lattice structure. SOM quantizes and visualizes given data set by the learning using competitive layer. The accuracy of SOM vector quantization depends on the number of competitive layer's neurons because the codebook vectors correspond to the neurons in competitive layer. Therefore, it is necessary to decide the sufficient competitive layer size for given data set however it is difficult to decide when an unknown data set is given.

For automatically finding out a competitive layer size, the growing type SOMs that add and delete neurons have been researched [4]-[7]. The approach of competitive layer adaptation is that increase neurons at high probability density regions in input data space for improving quantization accuracy.

In this paper, we proposed a variant of growing type SOM that put the neurons in the regions that has high gradient in probability density estimation. The proposed growing type SOM is based on Tree Structured SOM [8]. TS-SOM is a faster SOM variant applying tree search algorithm. We applied the pruning of neurons and the layer creation to a tree structure of TS-SOM by using the means error among neighboring neurons' weight vector and frequency in use for winner.

II. SELF-ORGANIZING FEATURE MAP

A. Basic Self-Organizing Feature Map

SOM is a layered neural network consisting of an input layer and a competitive layer proposed by T. Kohonen [1]. SOM is trained using unsupervised learning to produce low dimensional representation for the training samples while preserving the topological information of the input vectors. SOM characteristics such as low computational cost, unsupervised learning and visualization is effective for the explore analysis in data-mining. Hence SOM has been applied to wide range of complex problems, such as speech recognition, optimal character recognition, text classification etc [1]-[3].

SOM can be expressed as a 2 layered neural network consists of an input layer and a competitive layer. When the input data set $S = \{ \mathbf{x}_i ; \mathbf{x}_i \in \Re^n \}$, $i = 1, 2, \ldots, imax$ is a set of n dimensional real vector $\mathbf{x}_i = \{x_1, x_2, \ldots x_n\}$ where $imax$ is the number of samples in input data set, the input layer is constructed with n input neurons. The competitive layer is constructed with arranged m neurons to 2 or 3 dimensional lattice structure. The jth neuron on competitive layer has the weight vector $\mathbf{w}_j = \{w_1, w_2, \ldots w_n\}$, $j = 1, 2, \ldots, m$, corresponding to the elements of input vector.

The SOM training consists of following 4 steps; (1) weight initialization, (2) selecting input sample, (3) determining winner, and (4) updating weight vector. The step (2) to (4) are repeated until satisfying the termination criterion based on the parameter for maximum training times or the energy function. The basic algorithm for online training using Euclidian distance for the distance measure is described as follows. In the step of weight initialization, the values of weight vectors are initialized by random initialization or liner initialization before training.

In the first step of iteration at training step $t = 1, 2, \ldots, T$ where T is the maximum number of training steps, an input sample $\mathbf{x}_i(t)$ is selected randomly from data set S. Next, a closest b_ith neuron to an input sample $\mathbf{x}_i(t)$ is selected from the neurons on competitive layer for a winner neuron b_i by equation (1)

$$b_i = \arg\min_j \left\{ \left\| \mathbf{x}_i(t) - \mathbf{w}_j(t) \right\| \right\} \quad (1)$$

where $\mathbf{w}_j(t)$ is jth weight vector at the training step t, $\| \mathbf{x}_i(t) - \mathbf{w}_j(t) \|$ is the distance between input vector $\mathbf{x}_i(t)$ and weight vector $\mathbf{w}_j(t)$, and distance measure is Euclidian distance. After the determining winner neuron b_i, the weight vectors of winner neuron and its neighbors are updated to close to input vector $\mathbf{x}_i(t)$ using equation (2)

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \alpha(t) \cdot h_{b_i j}(t) \cdot \left[\mathbf{x}_i(t) - \mathbf{w}_j(t) \right] \quad (2)$$

where $\alpha(t)$ is learning rate factor and $h_{b_i j}(t)$ is the neighborhood function that defines the form of neighborhood. The Gaussian type neighborhood function that commonly used is defined as equation (3)

$$h_{b_i j}(t) = \exp\left(-\frac{\left\| \mathbf{r}_{b_i} - \mathbf{r}_j \right\|^2}{2\sigma^2(t)} \right) \quad (3)$$

where $\left\| \mathbf{r}_{b_i} - \mathbf{r}_j \right\|$ is the distance between winner neuron b_i and jth neuron on the map of competitive layer and $\sigma^2(t)$ is a parameter that defines the width of Gaussian distribution. This weight updating method is called neighborhood learning. The monotonic decreasing function with training step t is used for the learning rate factor $\alpha(t)$ and the parameter $\sigma^2(t)$. In the SOM training process, the weight vectors close to input vectors by repeating these procedures. Finally, a map to the weight vectors from the input vectors is constricted by the neighborhood learning.

SOM partitions the input data set into m Voronoi regions that center on weight vectors [1], [3]. Therefore the accuracy of SOM vector quantization depends on the competitive layer size so that the sufficient competitive layer size should be determined for given input data set. The finding out proper competitive layer size is a fundamental problem in SOM. As an approach for solving this problem, adaptively change the competitive layer size adding and deleting the neurons have been researched [4]-[7]. In this paper, we proposed a method for adaptively change the competitive layer size in the process of training based on TS-SOM.

B. Growing Type Self-Organizing Feature Maps

The adaptation of the competitive layer size is an important problem in SOM because SOM vector quantization accuracy depends on the number of neurons in competitive layer. An approach for the finding out proper number of neurons is adaptively change the competitive layer size adding and deleting the neurons such as Growing Grid (GG) [4], Growing Cell Structure (GCS) [5], Growing Hierarchical SOM (GH-SOM) [6], and Growing Hierarchical Tree SOM (GHT-SOM) [7]. This approach can find out the proper number of neurons in the training process so that these methods improve the flexibility for the application to unknown data set.

GG and GCS add neurons at high probability density regions in input data space similar to another vector quantization method Growing Neural Gus [9]. This growing approach is effective for improving vector quantization accuracy. GH-SOM and GHT-SOM add neurons at high probability density regions similar to GG and GCS. In addition, these methods have a hierarchical structure in the competitive layer such as TS-SOM [8] and Hierarchical SOM [10]. An aim for the approach of using hierarchical structure in competitive layer is extract a hierarchical relationship from input data set in order to improve the clustering capability of SOM.

The applying to cluster analysis is the popular application of SOM due to the intuitive cluster visualization of SOM map. Generally, the clusters can be visually confirmed in a map that visualizes distance between the neighboring weight vectors [1]-[3]. On the other hand the boundary of cluster is depends on visual decision by human or other clustering method such as agglomerative hierarchical clustering. Moreover, the region size of cluster is depends on the distribution in input dataset such that the dense cluster gave large region and the sparse cluster gave small region in the map [1]. This characteristic can be considered to robustness for the noise in input data set but the important sparse cluster can not be discovered such as rare case in medical diagnostics. For solving this problem, it is necessary to use more neurons in the boundary of cluster in order to improve the approximation accuracy of discriminant function for each cluster. Furthermore, the architecture for separating approximation of the vectors and the probability density is necessity in the learning process in order to discover the sparse cluster.

In this paper, we proposed a variant of growing type SOM that put the neurons in the regions that has high gradient in probability density estimation. The proposed growing type SOM is based on TS-SOM. TS-SOM is a faster SOM variant applying tree search algorithm. We applied the pruning of neurons and the layer creation to a tree structure of TS-SOM by using the means error among neighboring neurons' weight vector and frequency in use for winner.

III. TREE-STRUCTURED SELF-ORGANIZING FEATURE MAP

TS-SOM is a faster SOM method applying tree search algorithm proposed by Koikkalainen [8]. TS-SOM uses hierarchical neural network structure consisted of multiple competitive layers arranged from small sized layer to large sized layer like as shown in Fig. 1. Fig. 1 shows a case of 2 dimensional grid typed map. Each layer correspond to the competitive layer in basic SOM, the white colored points are neurons. and the bold lines are connection between parent and child neurons. The parent neuron in upper layer connects to 4 child neurons in bottom layer. The child neurons arranged at the position which divides parent's region into quarters. TS-SOM can be the determining winner neuron by which the parent neurons lead the determining winner neuron from child neurons in bottom layer. Each layer is trained individually by basic SOM training method described in section II from the top layer

to bottom layer. Before the layer is trained, the weight vectors are initialized from the parent neuron and the neighbors of initialized neuron. As a result, higher resolution maps can be yield preserving learned topological information in the upper layer to the bottom layer.

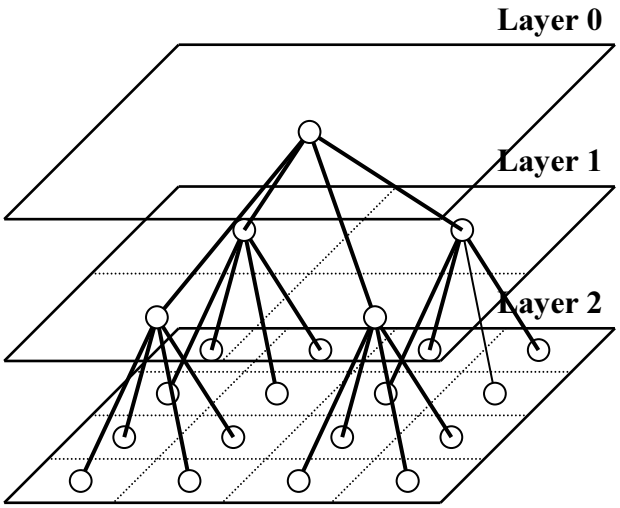

Fig. 1. 2 Dimensional Neural Network Structure of TS-SOM.

The detailed algorithm of TS-SOM is described as follows. Let be the weight vector in multiple layers of TS-SOM by $W^l = \{\mathbf{w}_{j^l}\}, j^l = 1, 2, \ldots, m^l$ with the index for layer $l = 0, 1, 2, \ldots,$ *lmax* where 0th layer is a root or top layer that has only 1 neuron. The j^lth neuron in *l*th layer connected with some child neurons in bottom *l*+1th layer as shown in Fig. 1. Let C_{j^l} be set of child neurons for the parent neuron j^l. In the case of 2 dimensional grid typed map, the number of child neurons is that $|C_{j^l}| = 4$ and number of neurons in *l*th layer is that $|W^l| = 4^l$. The TS-SOM learning method using this neural network structure for the determining winner neuron, weight initialization and updating weight vectors are described as follows.

The determining winner neuron in TS-SOM is based on a tree search algorithm. The winner neuron is recursively determined from a root to the child neuron in next layer. The winner neuron b_i^l in layer l is selected from the child nodes $C_{b_i^{l-1}}$ of previous layer's winner neuron b_i^{l-1} using following equation (4)

$$b_i^l = \arg\min_{j^l \in C_{b_i^{l-1}}} \left\{ \left\| \mathbf{x}_i(t) - \mathbf{w}_{j^l}(t) \right\| \right\} \tag{4}$$

where $l \neq 0$ since 0th layer is a root.

In TS-SOM, the weight vector is updated individually in each layer when the updating weight in previous upper layer is finished. The weight vector is updated using following equation (5) similar to basic SOM weight updating shown in equation (2)

$$\mathbf{w}_{j^l}(t+1) = \mathbf{w}_{j^l}(t) + \alpha(t) \cdot h_{b_i^l j^l} \cdot \left[\mathbf{x}_i(t) - \mathbf{w}_{j^l}(t) \right]. \tag{5}$$

where $\alpha(t)$ is learning rate factor and $h_{b_i^l j^l}$ is the neighborhood function. Note that the fixed grid type neighborhood function can be used in TS-SOM as shown in Fig. 2. The gray colored area is the neighborhood and a black colored point is a winner neuron in Fig. 2. The Gaussian type neighborhood function that the width is decreasing with training step *t* is commonly used in basic SOM in order to yields the fine map adjustment in latter training steps. However it is not necessity in TS-SOM because the detailed training is carried over to the training in lower layer [8].

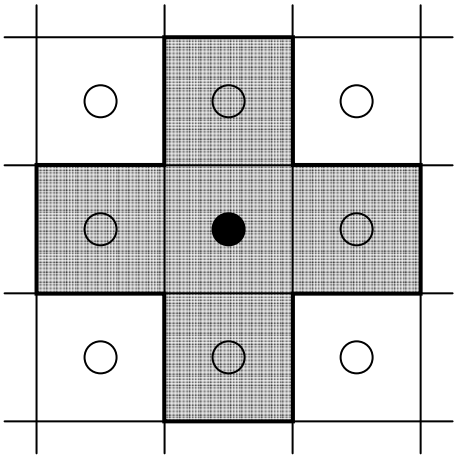

Fig. 2. Fixed Grid Type Neighborhood Function of TS-SOM.

Before the training in each layer, the weight vectors are initialized from the parent neuron and the neighbor neurons of target neuron. The weight vector is initialized by 2 steps of following equation (6) and equation (7)

$$\mathbf{w}_{j^l} \leftarrow \mathbf{w}_{k^{l-1}}, \tag{6}$$

$$\mathbf{w}_{j^l} \leftarrow \mathbf{w}_{j^l} + \beta \cdot \overline{\mathbf{w}}_{j^l} \tag{7}$$

where $l \neq 0$, k^{l-1} is the parent neuron for j^lth neuron, $\overline{\mathbf{w}}_{j^l}$ is centroid of the set of neurons in the neighborhood for j^lth neuron that denoted by N_{j^l} and is gray colored points in Fig. 2, and β ($0 < \beta < 1$) is a parameter that defines the modification rate for the neurons in neighborhood.

TS-SOM can continue the learning adding new layer in bottom using above mentioned weight initialization method if the training is finished in upper layer. However the number of neurons is greatly increased whenever new layer is added since the number of neurons in *l*th layer is 4^l. For the preventing the increasing number of neurons at the adding new layer, we investigated the adaptive learning method that is optimize the hierarchical neural network structure adding and deleting the units that consists of a parent neuron and 4 child neurons in the process of training. Additionally, the methods for the updating weight vectors were investigated because the updating method in TS-SOM can not be used for dynamically changed neural network structure.

IV. Adaptive Learning Algorithm in TS-SOM

A. Neural Network Structure Adaptation

In this paper, we proposed a variant of growing type SOM that put the neurons in the regions that has high gradient in probability density estimation in order to adapt a neural network structure of SOM for clustering. In the proposed growing type SOM, we applied the pruning of neurons and the layer creation to a tree structure of TS-SOM by using the means error among neighboring neurons' weight vector and frequency in use of winner.

In the TS-SOM, the new layer is added in bottom layer if the training of previous layer is completed. In the proposed method, the new layer can be added partially in a pyramidal unit with a parent neuron and 4 child neurons. Fig. 3 shows the process of growing neural network structure. The proposed method neural network structure is similar to TS-SOM such that the child neurons arranged at the position which divides parent's region into quarters. Fig. 4 shows the coordinate mapping for each partial layer in the case of a neural network structure shown in Fig. 3. The proposed hierarchical competitive layer can be mapped to a layer using the neurons at leaf in tree structure. And so, it can be consider a competitive layer to have partially high resolution region. Unlike TS-SOM, because the progress of training is different in the region on each layer, the adaptation of weight updating method should be considered for proper training. The weight updating methods is discussed in section IV-B. The detailed algorithm for adapting neural network structure is described as follows.

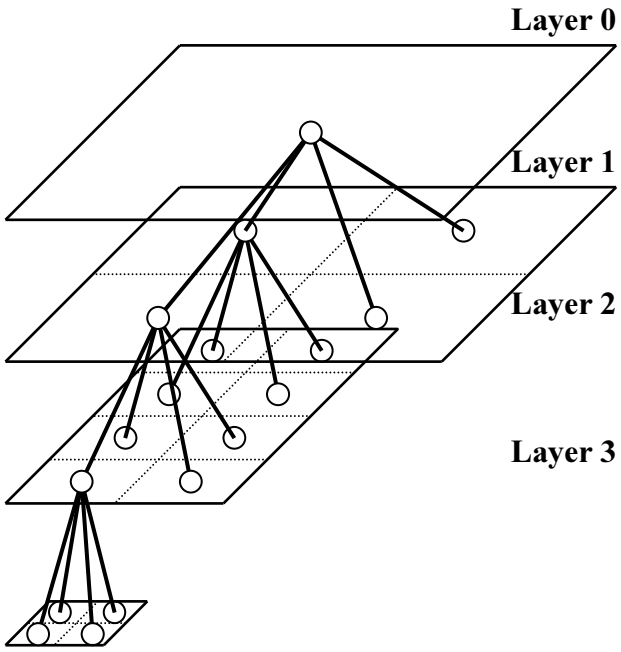

Fig. 3. Neural Network Structure of Proposed Growing Type SOM.

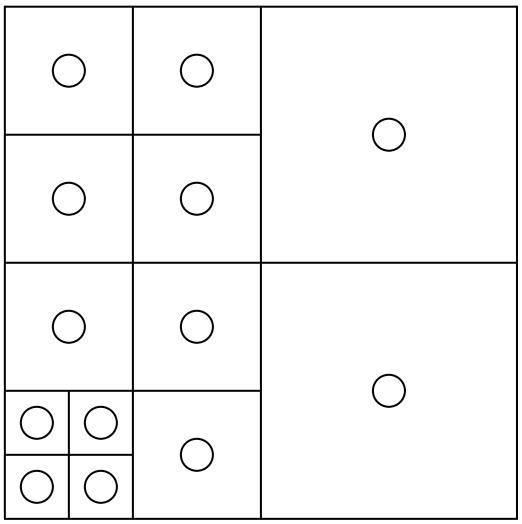

Fig. 4. The Coordinate Mapping for Multiple Competitive Layers.

In the proposed neural net work structure adaptation method, the procedure of adding unit or delete unit is performed when satisfy the target criterion in each training step τ. τ is a user defined parameter, and is $\tau = 500 - 1000$ is used. It is correspond to the training step for completing rough fitting weight vectors to input vectors in basic SOM.

The proposed method adds the pyramidal units in early stage for the covering whole input vector space in detail, and deletes the unnecessary pyramidal units for clustering in latter stage, because it is not possible to learn appropriately if the neuron is deleted when the weight vectors are not yet fitting to input vectors. The condition for distinction between the early stage and latter stage is defined by follows using a maximum training step T and a user defined parameter γ ($0 < \gamma < 1$). It is defined to be early stage when $1 < t < \gamma T$ and it is defined to be latter stage when $\gamma T \le t \le T$.

The initial neural network structure is minimum tree structure consists of a root unit (1 parent neuron and 4 child neuron). The weight vectors in a root unit are initialized by random values.

In the early stage of training, the new child neurons are added under a neuron in the leaf unit when the error between target neuron's weight vector and its neighbor's weight vector is larger than threshold. Let L be the set of child neurons in all leaf unit. The child neurons are added under the j^lth neuron such that $j^l \in L$ when satisfy equation (8) and equation (9)

$$\max_{k^l \in N_{j^l}} \left\| \mathbf{w}_{j^l} - \mathbf{w}_{k^l} \right\| > \varepsilon \tag{8}$$

$$\frac{p_{j^l}(S)}{|S|} > \rho \tag{9}$$

where $j^l \neq k^l$, k^l is a neuron in neighborhood N_j for j^lth neuron, ε ($0 < \varepsilon < 1$) is a parameter defines the allowable error in the training, $p_j(S)$ is the frequency in use for winner neuron when input data set S is given, and ρ ($0 < \rho < 1$) is parameter defines

the minimum value for frequency in use. The definition of neighborhood N_j is same as the neighborhood of TS-SOM shown in Fig. 2. A lot of unused neurons are yield because the neurons are added regardless of distribution in input data set if the equation (9) is not considered. For the preventing unused neurons, equation (9) was applied so that the neurons are not added to the target neuron that the frequency in use is extremely low. The weight vectors of new child neurons are initialized using equation (6) and equation (7).

In the latter stage of training, the child neurons in leaf units are deleted if the variance of frequency in use for winner neuron is smaller than threshold. Consider the deleting child nodes procedure in a leaf unit has k^{l-1}th neuron as parent where $l \neq 0$. The 4 child neurons such that $j^l \in C_{k^{l-1}}$ and $j^l \in L$ are deleted when satisfy equation (10) and equation (11)

$$\frac{v_{k^{l-1}}(S)}{|S|} > \theta \tag{10}$$

$$\frac{p_{k^{l-1}}(S)}{|S|} \leq \rho \tag{11}$$

where $v_{k^{l-1}}(S)$ is a variance of frequency in use for winner neuron in the set of child neurons $C_{k^{l-1}}$, and θ ($0 < \theta < 1$) is the user defined parameter. When 2 dimensional grid type competitive layer is used, the variance of frequency $v_{k^{l-1}}(S)$ can be calculated by equation (12)

$$v_{k^{l-1}}(S) = \frac{1}{4} \sum_{j^l \in C_{k^{l-1}}} \left(\frac{p_{k^{l-1}}(S)}{4} - p_{j^l}(S) \right)^2 . \tag{12}$$

The variance of frequency $v_{k^{l-1}}(S)$ is the estimator for the gradient of probability density. Thus it can be considered that the proposed adaptation method keep the neurons arranged in which the probability density is changed. Equation (11) is for the deleting the neuron that the frequency in use is extremely low as well as the equation (9).

B. Gaussian Type Neighborhood Learning Model

The weight updating method of TS-SOM can not be used in dynamic and partial neural network structure because the training in upper layer should be completed before the training in lower layer for the preserving topological relation between each layer. For resolving this, we proposed a grid type neighborhood leaning model that recursively updates the weight vector from root to leaf in each winner neuron and its neighbor [11]. From the computer simulation, the proposed method yielded slightly worse classification accuracy than TS-SOM. It is considered that this worsening accuracy caused by the error in the fixed grid type neighborhood because TS-SOM also yielded worse accuracy than basic SOM. In this paper, we proposed Gaussian type neighborhood leaning model in order to improve classification accuracy.

The proposed Gaussian type neighborhood leaning model update the weight vector using Gaussian kernel without recursive procedure. The position of neurons is calculated on a global coordinate system corresponding to map like as Fig. 4.

The determining winner neuron is same as TS-SOM. The winner neuron recursively determined from a root to the child neuron in next layer. This tree search determines the winner neuron b_i^l in each layer by equation (4) and eventually determines the global winner neuron b_i^g such that $b_i^g \in L$ in layer g. After the global winner neuron b_i^g is determined, All weight vector in all layer is updated using following equation (13)

$$\mathbf{w}_{j^l}(t+1) = \mathbf{w}_{j^l}(t) + \alpha(t) \cdot h_{b_i^g j^l}(g) \cdot \left[\mathbf{x}_i(t) - \mathbf{w}_{j^l}(t) \right] \tag{13}$$

where $h_{b_i^g j^l}(g)$ is Gaussian type neighbor hood function that the width is defined by the depth of global winner neuron g. When 2 dimensional grid type competitive layer is used, $h_{b_i^g j^l}(g)$ can be defined by following equation (14)

$$h_{b_i^g j^l}(g) = \exp\left(- \frac{\left\| \mathbf{r}_{b_i^g} - \mathbf{r}_{j^l} \right\|^2}{2\sigma^2(g)} \right) \tag{14}$$

where $\left\| \mathbf{r}_{b_i^g} - \mathbf{r}_{j^l} \right\|$ is the distance between b_i^gth neuron and j^lth neuron on the coordinate of competitive layer, and $\sigma^2(g)$ is a parameter that defines the width of Gaussian distribution. The coordinate of j^lth neuron $\mathbf{r}_{j^l} = \{ r_{j^l q} \}$, $q = 1, 2$ is calculated by equation (15)

$$r_{j^l q} = r_{k^{l-1} q} + \frac{0.5}{2^l} \tag{15}$$

where $l \neq 0$, and $r_{k^{l-1} q}$ is the parent neurons coordinate value. The coordinate of neurons can be calculated when the neuron is newly added. For fitting the width of Gaussian distribution to grid size of gth layer, $\sigma^2(g)$ is defined by equation (16)

$$\sigma^2(g) = \frac{\sigma^2(0)}{2^g} \tag{16}$$

where $\sigma^2(g)$ is a parameter defines a initial width of Gaussian distribution.

V. EXPERIMENT

For evaluating the classification performance, we applied the proposed growing type SOM to iris and wine data set. Iris and wine data set are most popular benchmark data set provided by UCI Machine Learning Repository. The descriptions of these data are shown in Table 1. Iris data set contains 3 classes of 50 samples each. Wine data set contains 3 classes with different class ratio. Each value was normalized to be [0, 1].

TABLE 1 NUMBER OF CLASSES AND ATTRIBUTES IN DATA SETS

Name	Classes	Samples	Ratio	Attributes
Iris	3	150	1:1:1	4
Wine	3	178	59:71:48	13

The proposed SOM with Gaussian neighborhood was compared in iris data set to following 3 methods; previous proposed SOM with fixed grid neighborhood, basic SOM with Gaussian neighborhood, and TS-SOM with fixed grid neighborhood. For the evaluating the performance in other data set, proposed method with Gaussian neighborhood and basic SOM was compared in wine data set. The parameter for proposed methods are follows; $\tau = 500$, $T = 50000$, $\gamma = 0.5$, $\varepsilon = 0.01$, $\sigma = 0.01$, $\theta = 0.01$, $\alpha(t) = \alpha_0 \cdot \alpha_{dec}{}^t$, $\alpha_0 = 0.5$, $\alpha_{dec} = 0.99995$. The competitive layer size of basic SOM and TS-SOM are determined by prior experiments so that the size is similar to proposed SOM.

TABLE 2 COMPARISONS FOR CLASSIFICATION PERFORMANCE

Method	Classification Accuracy	Quantization Error	Size
Iris Data Set			
Proposed SOM (Gaussian)	0.941	0.051	17.8
Proposed SOM (Fixed Grid)	0.886	0.143	13.0
Basic SOM (Gaussian)	0.959	0.022	25.0
TS-SOM (Fixed Grid)	0.901	0.033	21.0
Wine Data Set			
Proposed SOM (Gaussian)	0.952	0.267	18.6
Basic SOM (Gaussian)	0.961	0.204	25.0

These methods are evaluated by the correctly classification rate, the quantization error, and the number of neurons in competitive layer. For the calculating correctly classification rate, each neuron was labeled by actual class label that has maximum frequency. Table 2 shows the result of average in 100 times separate runs. In the result of iris data set, it can be confirmed that the classification accuracy of proposed SOM with Gaussian neighborhood was greatly improved from previous SOM with fixed grid type and yields very close accuracy with basic SOM. The quantization error of proposed method tends to be increased compared with basic SOM and TS-SOM because the proposed method delete the neurons in dense point. This tendency was confirmed in the result show in Table 2 but the quantization error is greatly improved as well as classification accuracy.

VI. CONCLUSION

In this paper, we proposed a variant of growing type TS-SOM that put the neurons in the regions that has high gradient in probability density estimation in order to adapt the competitive layer size and structure for the clustering input data space. From the computer simulation, we shows that the proposed SOM can yield very close accuracy with basic SOM that trained by sufficient competitive layer size.

In the training of proposed SOM, the computational time is tend to increase compared with basic SOM and TS-SOM. The reasons for this is considered to the calculation of frequency $p_j(S)$ depends on input data size N, the increase of updating neurons by Gaussian neighborhood, and the increase of training step due to the iterative neuron addition. The increase of training is important point because it is related to optimize the maximum training step T and parameter τ defines the interval of neural network structure modification. The parameter τ should be determined considering the complexity of data set such that the number of classes and the class ratio. And so, we plan to investigate the method for optimizing parameter τ in the training process in the future.

REFERENCES

[1] T. Kohonen, "Self-Organizing Maps", Springer-Verlag, 1984.

[2] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas, "Engineering applications of the self-organizing map", in Proc. IEEE, Oct. 1996, vol.84, pp. 1358-1384.

[3] J. Vesanto, E. Alhoniemi, "Clustering of the Self-Organizing Map", IEEE Transactions on Neural Networks, 11(3), pp 586-600, 2000.

[4] B.Fritzke, "Growing Grid - a self-organizing network with constant neighborhood range and adaptation strength", NeuralProcessing Letters, vol.2, no.5, pp.9-13, 1995.

[5] B Fritzke, "Growing cell structures - A self-organizing network for unsupervised and supervised learning", Neural Networks, vol. 7, no. 9, pp. 1441-1460, 1994.

[6] M. Dittenbach, D. Merkl, A. Rauber. "The Growing Hierarchical Self-Organizing Map". In Proc. Int. Joint Conf. on Neural Networks, Vol. 6, pp. 15-19, 2000.

[7] A. Forti, G.L. Foresti, "Growing Hierarchical Tree SOM : An Unsupervised Neural Network with Dynamic Topology", Neural Networks, Vol. 19, No. 10, pp. 1568-1580, 2006.

[8] P. Koikkalaien, E. Oja, "Self-Organizing Hierarchical Feature Maps", International Joint Conference on Neural Networks, IEEE Neural Networks Council, vol.2, pp.279-284, Jun. 1990.

[9] B Fritzke, "A growing neural gas network learns topologies", Advances in Neural Information Processing Systems, vol.7, pp.625-632, 1995.

[10] J. Lampinen and E. Oja, "Clustering properties of hierarchical self-organizing maps", J. Math. Imag. Vis., vol. 2, no. 2–3, pp. 261–272, Nov. 1992.

[11] T. Yamaguchi, T. Ichimura, K. J. Mackin, "Adaptive Neural Network Topology in Tree-Structured Self-Organizing Feature Map", Proceedings of 26th Fuzzy System Symposium, 2010 (in Japanese).

[12] C. L. Blake and C. J. Merz. UCI repository of machine learning databases. Technical report, University of California, Department of Information and Computer Science, Irvine, CA, 1998. Available at: http://archive.ics.uci.edu/ml/.