Visualization using Multi-Layered U-Matrix in Growing Tree-Structured Self-Organizing Feature Map

Takashi Yamaguchi
Department of Information Systems
Tokyo University of Information Sciences
4-1 Onaridai, Wakaba-ku,
Chiba, 265-8501, Japan
tyamagu@rsch.tuis.ac.jp

Takumi Ichimura
Faculty of Management and Information Systems
Prefectural University of Hiroshima
1-1-71, Ujina-Higashi, Minami-ku,
Hiroshima, 734-8559, Japan
ichimura@pu-hiroshima.ac.jp

***Abstract*—Self-organizing feature map (SOM) is well known artificial neural network using unsupervised learning for the data visualization and vector quantization. SOM has been used for cluster analysis. On the other hand, SOM cannot produce clarified clusters. And so SOM clustering capability is depends on visualization method. We proposed a variant of SOM that construct hierarchical neural network structure to clarify cluster boundaries in previous research. In this paper, we proposed a visualization method for this growing Tree-Structured SOM and discuss the computational result of Iris data.**

Keywords-component; Self-Organizing Feature Map, Tree-Structure, Data Visualization, U-Matrix

I. INTRODUCTION

Self-organizing feature map (SOM) is an artificial neural network for the 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 unsupervised learning.

A unique characteristic of SOM is visualization. SOM can represent the features of date set from various visualization methods [1], [10], [11]. U-Matrix [11] is commonly used SOM visualization method to show the cluster boundaries in data set. However, U-Matrix method cannot clarify cluster boundaries when the data set includes the sparse clusters. This problem is occurred from insufficient map resolution and SOM cannot learn the features of samples in sparse clusters.

In order to clarify boundaries of cluster and learn sparse clusters, we proposed a variant of growing type SOM based on Tree Structured SOM [9] in previous research [8]. Our growing SOM put the neurons in the regions that has high gradient in probability density estimation for clarify boundaries of clusters, and generates hierarchical competitive layers to learn the features of samples in sparse clusters. In this paper, we proposed a cluster visualization method based on U-Matrix for our growing TS-SOM in order to discover the sparse cluster. The proposed visualization method presents a map that is merged from multiple competitive layers of growing TS-SOM. This map similar to basic SOM U-Matrix maps, but it has partially high resolution regions.

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 samples while preserving the topological information. SOM characteristics such as low computational cost, unsupervised learning and visualization is effective for the explore analysis in data-mining.

SOM can be expressed as two layered neural network consists of an input layer and a competitive layer shown as Figure 1. When the input data set $S = \{\mathbf{x}_i; \mathbf{x}_i \in \Re^n\}, i = 1,2,\dots,imax$ is a set of n dimensional real vector $\mathbf{x}_i = \{x_1, x_2, \dots, x_n\}$ where $imax$ is the size of input data set S. The input layer is constructed with n input neurons. The competitive layer is constructed with arranged m neurons to two or three dimensional lattice structure. The jth neuron on competitive layer has the weight vector $\mathbf{w}_j = \{w_{j1}, w_{j2}, \dots, w_{jn}\}$, $j = 1, 2, \dots, m$, corresponding to the elements of input vector.

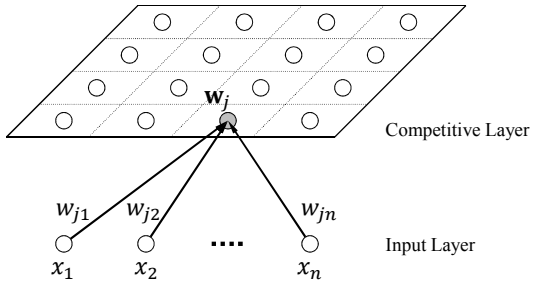

Figure 1. An Overview of SOM

The SOM training consists of following 4 steps; (1) weight initialization, (2) input sample selection, (3) determining winner neuron, and (4) updating weight vector. The step (2) to (4) are repeated until satisfying the termination criterion. The basic algorithm is described as follows.

In the weight initialization step, the weight vectors are initialized by random initialization or liner initialization before training. In the first step of iteration at training step $t = 1, 2, \dots, T$ where T is the maximum training steps, an input sample $\mathbf{x}_i(t)$ is selected randomly from data set S. Next, a

closest b_i th neuron to a sample $\mathbf{x}_i(t)$ is selected from the competitive layer for a winner neuron b_i by equation (1)

$$b_i = \underset{j}{\operatorname{argmin}}\{\|\mathbf{x}_i(t) - \mathbf{w}_i(t)\|\} \quad (1)$$

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

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

where $\alpha(t)$ is learning rate factor and $h_{b_i j}(t)$ is the neighborhood function that defines the shape and the size of region for neighborhood on the map. The Gaussian neighborhood function is defined as equation (3)

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

where $\|\mathbf{r}_{b_i} - \mathbf{r}_j\|$ is the distance between winner neuron b_i and jth neuron on the map 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 time step t is used for the leaning 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 constructed by the neighborhood learning.

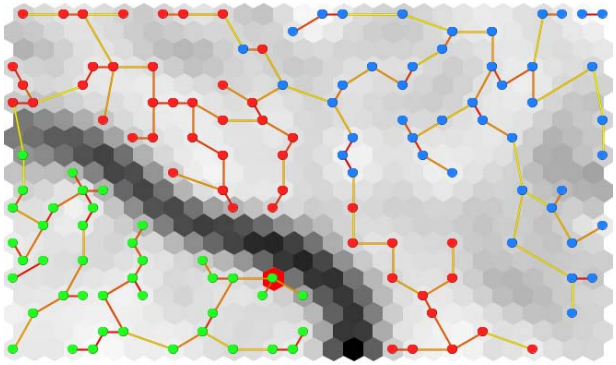

Figure 2. An Example of SOM Visualization with Class Visualization, Minimum Spanning Tree, and U-Matrix

B. *Visualization in Self-Organizing Feature Maps*

A unique characteristic of SOM is visualization. An example of SOM visualization using iris data set [12] is shown in Figure 2. This map is constructed by three different methods; class visualization [1], minimum spanning tree [1], [10] and U-Matrix [11]. The colored points are class visualization that shows the distribution and location of classes from input data set. The yellow colored lines are minimum spanning tree that generated from the hierarchical clustering method with winner neurons [10]. The gray colored area is U-Matrix visualization.

The U-Matrix is commonly used visualization method for the cluster analysis using SOM. The U-Matrix method visualizes based on the distance in input space between a weight vector and its neighbors on map. Let N_j be index set of neurons that are neighboring j th neuron. U-Matrix method calculates the U-height u_j for each neuron on competitive layer by equation (4).

$$u_j = \frac{1}{|N_j|} \sum_{k \in N_j} \|\mathbf{w}_j - \mathbf{w}_k\| \quad (4)$$

The shape of neighborhood function for N_j is different from the neighborhood function $h_{b_i j}(t)$ for SOM training. The simple neighborhood function that includes 4 or 8 neurons is commonly used. Figure 3 is a grid type neighborhood function with 4 neurons. A black colored point is jth neuron and the gray colored area is neighborhood. j th neuron has 4 neighboring neurons shown in gray colored point.

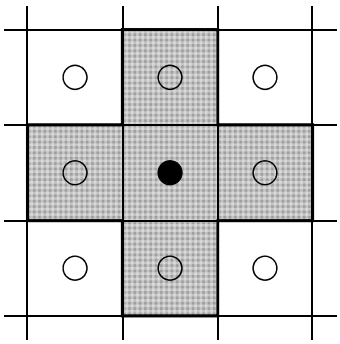

Figure 3. Grid Type Neighborhood

The darker gray colored area has high U-height in Figure 2. It can be seen U-Matrix method clarifies a cluster boundary between the green colored class and the other. On the other hand, the other cluster boundaries cannot be seen in Figure 2. It is known that U-Matrix method cannot clarify cluster boundaries when data set includes sparse clusters or a lot of sub cluster. A.Ultsch proposed U*-Matrix[13] for handling data set with sub clusters .

In this paper, we focus on discovering sparse cluster. 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 cannot be discovered such as rare case in medical diagnostics.

Considering U-Matrix based visualization, the approaches that applies the hierarchical structure to SOM is effective for discovering the sparse cluster because the cluster boundaries derived by U-height u_j is different when the map resolution is changed. Figure 4 is a comparison with four different resolution U-Matrix visualizations for iris data set in basic SOM. The different cluster boundaries can be seen in green and blue colored area. In this paper, we investigated a visualization method that aggregate these different resolution U-Matrix maps in order to clarify cluster boundaries.

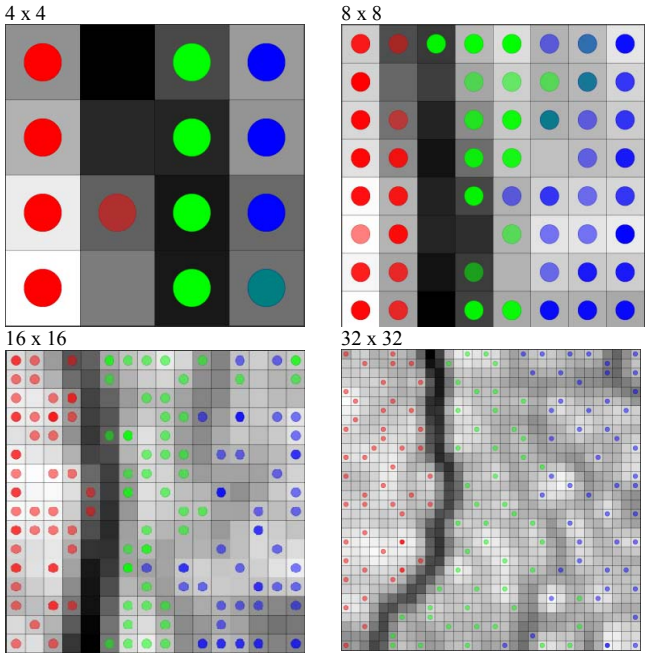

Figure 4. Comparison with Different Resolution U-Matrix Visualizations

III. TREE-STRUCTURED SELF-ORGANIZING FEATURE MAP

A lot of SOM variants that has hierarchical competitive layer such as Growing Hierarchical SOM [6], and Growing Hierarchical Tree SOM [7], Hierarchical SOM [14], etc. An aim for using hierarchical structure in competitive layer is extract a hierarchical relationship from input data set in order to improve the clustering capability of SOM.

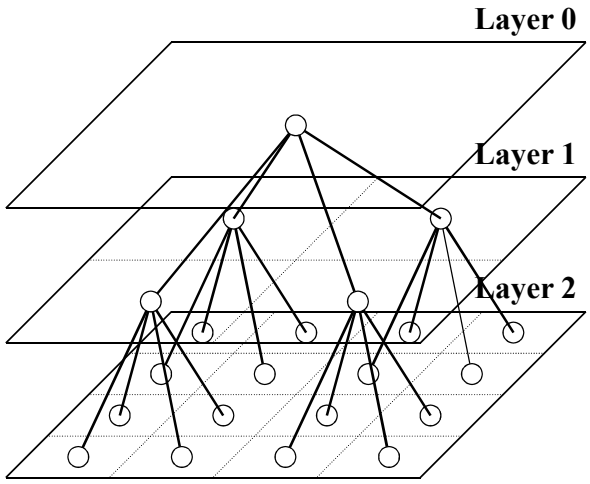

Figure 5. A Hierarchical Structure of TS-SOM

An aim of TS-SOM is different from these hierarchical SOM variants. TS-SOM is a faster SOM method applying tree search algorithm proposed by Koikkalainen [9]. 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 Figure 5. Each layer corresponds 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 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.

The detailed algorithm of TS-SOM is described as follows. Let be the weight vector in multiple layers of TS-SOM by $W^l = \{\mathbf{w}_{j^l}\}, j^l = 1, 2, \dots, m^l$ with the index for layer $l = 1, 2, \dots, lmax$ where 0th layer is a root or top layer that has only one neuron. The j^lth neuron in lth layer connected with some child neurons in bottom $l+1$th layer. Let C_{j^l} be set of child neurons for the parent j^lth neuron. In the case of two dimensional grid typed map, the number of child neurons is that $|C_{j^l}| = 4$ and neurons size in lth layer is that $|W^l| = 4^l$. The TS-SOM learning using this neural network structure for the weight initialization, determining winner neuron, 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 lth layer is selected from the child nodes $C_{b_i^{l-1}}$ of previous layer's winner neuron b_i^{l-1} using following equation (5)

$$b_i^l = \underset{j^l \in C_{b_i^{l-1}}}{\operatorname{argmin}} \{\|\mathbf{x}_i(t) - \mathbf{w}_{j^l}(t)\|\} \tag{5}$$

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

In standard TS-SOM, the weight vector is updated individually in each layer when the updating weight in previous upper layer is finished. Before the training in each layer, weight vectors are initialized from the parent neuron and the neighbor neurons of target neuron. he TS-SOM weight updating method is similar to basic SOM but the fixed grid type neighborhood function is used.

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 neuron size is greatly increased whenever new layer is added since the neuron size in lth layer is 4^l. For the preventing the increasing neuron size at the adding new layer, we investigated the adaptive learning method that is optimize the hierarchical neural network structure by the addition and elimination of unit 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 cannot be used for dynamically changed neural network structure.

A. Neural Network Structure Adaptation

In previous research, 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.

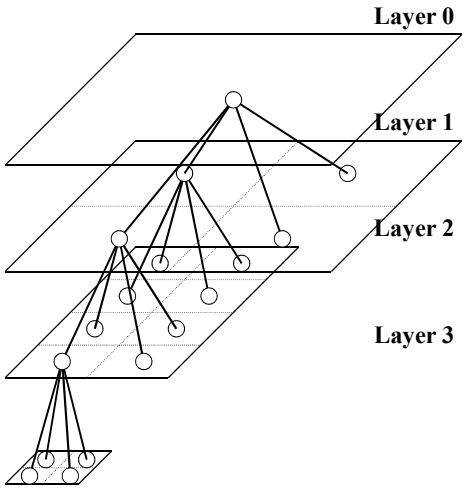

Figure 6. An Overview of Adapted Neural Network Structure

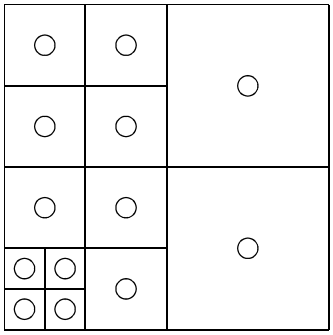

Figure 7. An Overview of Coodinate Mapping for Multiple Layers

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. Figure 6 shows an overview 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. Figure 7 shows the coordinate mapping for each partial layer in the case of a neural network structure shown in Figure 6. 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 III-B. The detailed algorithm for adapting neural network structure is described as follows.

In the proposed neural network structure adaptation method, the procedure of unit addition or elimination is performed when satisfy the target criterion in each training step τ. τ is a user defined parameter, and is $\tau = 500$ to 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 eliminates the unnecessary pyramidal units for clustering in latter stage, because it is not possible to learn appropriately if the neuron is eliminated when the weight vectors are not yet fitting to input vectors. The condition for distinction between the early stage and later stage is defined by follows using a maximum training step T and a user defined parameter $\gamma(0 < \gamma < 1)$. It is defined to be early stage when $1 < t < \gamma T$ and it is defined to be latter stage when $\gamma T \leq t \leq T$.

The initial neural network structure is minimum tree structure consists of a root unit (one parent neuron and four 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^l th neuron such that $j^l \in L$ when satisfy equation (6) and equation (7)

$$\max_{k^l \in N_{j^l}} \left\| \mathbf{w}_{j^l} - \mathbf{w}_{k^l} \right\| > \varepsilon \tag{6}$$

$$p_{j^l}(S) > \rho \cdot |S| \tag{7}$$

where $j^l \neq k^l$, k^l is a neuron in neighborhood N_{j^l} for j^l th neuron, $\varepsilon(0 < \varepsilon < 1)$ is a parameter defines the allowable error in the training, $P_{j^l}(S)$ is the frequency in use for winner neuron when input data set S is given, and $\rho(0 < \rho < 1)$ is parameter defines the minimum value for frequency in use. The definition of neighborhood N_{j^l} is same as the fixed grid neighborhood shown in Figure 3. A lot of unused neurons are yield because the neurons are added regardless of distribution in input data set if the equation (7) is not considered. For the preventing unused neurons, equation (7) 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 unit are initialized from the parent neuron and the neighbor neurons of target neuron. The weight vector is initialized by two steps of following equation (8) and equation (9)

$$\mathbf{w}_{j^l} \leftarrow \mathbf{w}_{k^{l-1}} \tag{8}$$

$$\mathbf{w}_{j^l} \leftarrow \mathbf{w}_{j^l} + \beta \cdot \overline{\mathbf{w}}_{j^l} \tag{9}$$

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 N_{j^l}, and

$\beta(0 < \beta < 1)$is a parameter that defines the modification rate for the neurons in neighborhood.

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

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

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

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 $\theta(0 < \theta < 1)$ is the user defined parameter. When two dimensional grid type competitive layer is used, the variance of frequency $v_{k^{l-1}}(S)$ can be calculated by equation (12)

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

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 unit elimination that the frequency in use is extremely low as well as the equation (7).

B. Gaussian Type Neighborhood Learning Model

The weight updating method of TS-SOM cannot 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. Moreover, TS-SOM yielded worse accuracy than basic SOM in prior experiment. It is considered that this worsening accuracy occurred by the error in the grid type neighborhood. Thus we proposed Gaussian neighborhood leaning model. The proposed Gaussian neighborhood leaning model updates the weight vector using Gaussian kernel without recursive procedure. The position of neurons is calculated on a global coordinate system corresponding to bottom layer.

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 lth layer by equation (4) and eventually determines the global winner neuron b_i^g such that $b_i^g \in L$ in gth layer. After the global winner neuron b_i^g is determined, all weight vectors in all layer is updated by 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 neighborhood function that the width is defined by the depth of global winner neuron g. When two 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 global coordinate system of competitive layers, 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)

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

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)

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

where $\sigma^2(0)$ is a parameter defines a initial width of Gaussian distribution.

C. Cluster Visualization for Growing TS-SOM

The growing TS-SOM has multiple competitive layer and partially high resolution region. In this paper, we proposed a visualization method using U-Matrix for growing TS-SOM. U-height of growing TS-SOM is calculated by two steps on global coordinate system defined in section III-B. First step is the U-height calculation for each layer. After that, the sum of all U-height at each global coordinate is calculated. Moreover, we investigated a density visualization method over the U-Matrix. The density information is important on SOM because SOM training gave high influence from probability density in input data set. Therefore the density information is visualized by circle that has different size correspond to density.

The global U-Matrix calculation is described as follows. The l th layer U-height u_{j^l} is calculated by equation (17) similar to U-height calculation on basic SOM.

$$u_{j^l} = \frac{1}{|N_{j^l}|} \sum_{k \in N_{j^l}} \left\| \mathbf{w}_{j^l} - \mathbf{w}_{k^l} \right\| \tag{17}$$

After the calculating U-height for each layer, calculate global U-height u_{j^g} by equation (18)

$$u_{j^g} = \sum_{l=1}^{g} \beta(l) \cdot u_{k^l} \tag{18}$$

where $\beta(l)$ is a parameter function to emphasize cluster boundaries on lth layer resolution. If $\beta(l) > 1$, then the cluster boundaries on lth layer are emphasized. The optimal $\beta(l)$ is different by input data set but the monotonic increasing function is a better function because U-height tends to decrease in lower layer.

The density information is visualized by circles over the U-Matrix visualization. The size of circle at j^l th neuron is calculated by equation (19).

$$d = \mu \cdot \frac{P_{j^l}(S)}{|S|} \tag{19}$$

where $j^l \in L$, and μ is a parameter to define the scale of circles.

IV. Experiment

For evaluating the visualization, we applied the proposed growing type SOM to iris data set. Iris data set are most popular benchmark data set provided by UCI Machine Learning Repository [12]. Iris data set contains 3 classes of 50 samples each. The proposed method parameters are follows; $\tau = 500$, $T = 50000$, $\gamma = 0.5$, $\varepsilon = 0.01$, $\rho = 0.01$, $\theta = 0.01$, $\sigma^2(0) = 0.5$, $\alpha(t) = \alpha_0 \cdot \alpha_{dect}$, $\alpha_0 = 0.5$, $\alpha_{dec} = 0.9995$, $\beta(l) = 1.25^l$.

A result for iris data is shown in Figure 8. In this result, the growing TS-SOM yielded 4 layers network structure consisted by 37 neurons. The bottom left map is aggregated U-matrix by proposed method and the others are U-Matrix visualization for each layer. The colored circle is density visualization. The color of circle is defined by class ratio from input data set.

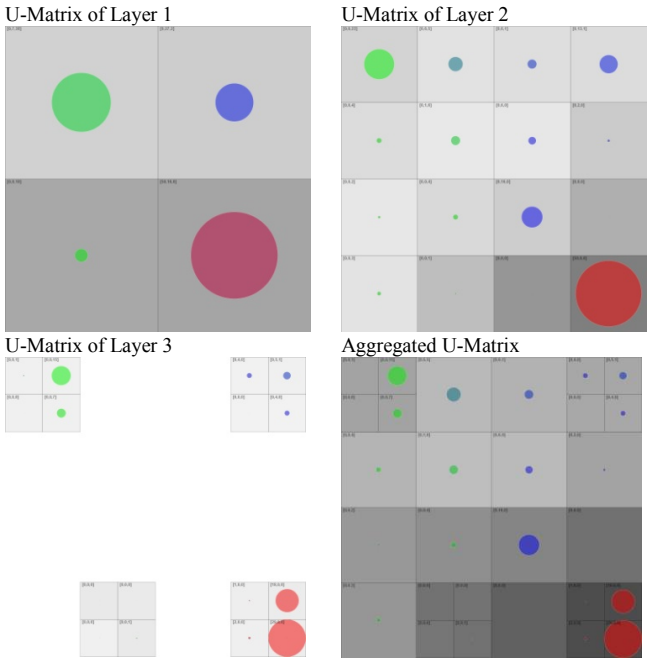

Figure 8. A Result of U-Matrix Visualization on growing TS-SOM

From the result of aggregated U-Matrix, it is difficult to distinguish the cluster boundaries. It was considered to be occurred from the insufficient resolution for cluster boundary visualization. The growing TS-SOM can represent the input data set with smaller neuron size compared with basic SOM. However, the cluster boundaries tend to be not clarified like as 4x4 map in Figure 4. For solving this, it is necessary to adapt U-Matrix method for small maps. On the other hand, the density visualization is good for intuitively representation over the U-Matrix visualization.

V. Conclusion

In this paper, we proposed a visualization method for U-matrix and density in the growing TS-SOM. From the experiment, proposed density visualization obtains good result. On the other hand, proposed U-Matrix based visualization cannot clarify cluster boundaries. It was caused by the insufficient resolution for cluster visualization occurred from the unit elimination of growing TS-SOM. For solving this problem, we plan to reconsider the U-height calculation and the parameter and criteria of unit elimination in order to clarify cluster boundaries.

For future works, we plan to investigate the effectiveness of the emphasize function $\beta(l)$ on hierarchical SOM that has smaller change in resolution. We expected that interactively configurable emphasize function $\beta(l)$ helps the knowledge acquisition on complex data analysis.

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] T. Yamaguchi, T. Ichimura, K. J. Mackin, "Adaptive Learning Algorithm in Tree-Structured Self-Organizing Feature Map", Proceedings of Joint 5th International Conference on Soft Computing and Intelligent Systems and 11th International Symposium on Advanced Intelligent Systems, 2010.

[9] 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.

[10] T. Yamaguchi, T. Ichimura, K.J.Mackin, Adaptive Tree Structured Clustering Method using Self-Organizing Map, Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International

[11] Alfred Ultsch and H. Peter Siemon, Kohonen's self-organizing feature maps for exploratory data analysis. In Proceedings of the International Neural Network Conference (INNC'90), pages 305-308. Kluwer, 1990.

[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/.

[13] A. Ultsch. U*-Matrix: a Tool to visualize Clusters in high dimensional Data. Technical Report 36, CS Department, Philipps-University Marburg, Germany, 2004.

[14] 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.